OCTOBER 1981
VOL. 60, NO. 8

THE BELL SYSTEM TECHNICAL JOURNAL

(Contents continued on back cover)

THE BELL SYSTEM TECHNICAL JOURNAL

THE BELL SYSTEM TECHNICAL JOURNAL is published monthly, except for the May–June and July–August combined issues, by the American Telephone and Telegraph Company, C. L. Brown, Chairman and Chief Executive Officer; W. M. Ellinghaus, President; V. A. Dwyer, Vice President and Treasurer; F. A. Hutson, Jr., Secretary. Editorial inquiries should be addressed to the Editor, The Bell System Technical Journal, Bell Laboratories, Room WB 1L-336, Crawfords Corner Road, Holmdel, N.J. 07733. Checks for subscriptions should be made payable to The Bell System Technical Journal and should be addressed to Bell Laboratories, Circulation Group, Bell Laboratories, 101 J. F. Kennedy Parkway, Short Hills, N.J. 07078. Subscriptions $20.00 per year; single copies $2.00 each. Foreign postage $1.00 per year; 15 cents per copy. Printed in U.S.A. Second-class postage paid at New Providence, New Jersey 07974 and additional mailing offices.

Comments on the technical content of any article or brief are welcome. These and other editorial inquiries should be addressed to the Editor, The Bell System Technical Journal, Bell Laboratories, Room WB 1L-336, Crawfords Corner Road, Holmdel, N.J. 07733. Comments and inquiries, whether or not published, shall not be regarded as confidential or otherwise restricted in use and will become the property of the American Telephone and Telegraph Company. Comments selected for publication may be edited for brevity, subject to author approval.

THE BELL SYSTEM
TECHNICAL JOURNAL

DEVOTED TO THE SCIENTIFIC AND ENGINEERING
ASPECTS OF ELECTRICAL COMMUNICATION

Volume 60 October 1981 Number 8

Transmission Studies of a Long Single-Mode Fiber—Measurements and Considerations for Bandwidth Optimization

By L. G. COHEN, W. L. MAMMEL, J. STONE,
and A. D. PEARSON

(Manuscript received March 12, 1981)

Loss and bandwidth spectra were measured in the longest length of single-mode MCVD *fiber drawn to date. The 21.7-km-long fiberguide has a 0.5 percent index difference between its 7-μm-diameter core and cladding. Chromatic dispersion effects resulted in a minimum dispersion at a wavelength near 1.35 μm. At 1.30 μm, the fiber loss and bandwidth were measured at 1 dB/km and 21 GHz-km (source linewidth = 4 nm), respectively. Potential system performance was estimated from calculations of dispersion power penalties and chromatic-dispersion-limited repeater spacings for 274- and 548-Mb/s data transmission rates. A new numerical parametric study was used to show how the bandwidth of a fiber can be optimized by properly choosing its core diameter and core-to-cladding index difference.*

I. INTRODUCTION

The high bandwidths and low losses of single-mode fibers make them leading contenders for use in future wideband undersea cable systems.[1] These systems are expected to transmit at 274 Mb/s between repeater stations that will be approximately 35 km apart.

In anticipation of future need for long lengths of single-mode fiber, the modified chemical vapor deposition (MCVD) preform fabrication process has been scaled up. By using 19- by 25-mm support tubes 4 ft long, and glass deposition rates of 0.5 g/min, very large preforms can be made in reasonable times. Each preform yields up to 40 km of fiber.[2]

The purpose of this paper is to present the results of transmission measurements made on a 21.7-km fiber—the longest continuous MCVD fiber drawn to date. Improved automated test set-ups were used to measure loss and dispersion spectra in the 1.06- to 1.7-μm wavelength region.[3,4] Group-delay measurements were used to determine the minimum dispersion wavelength of the fiber. Bandwidth spectra, calculated from group-delay measurements, were compared to direct measurements of pulse broadening due to light sources with 4-nm emission linewidths.[5] Potential system performance was estimated by using the baseband frequency response of the fiber to calculate dispersion power penalties and chromatic-dispersion-limited repeater spacings for 274- and 548-Mb/s data rates.[6,7] Finally, results from a numerical study were used to suggest more optimal waveguide parameters for future fibers that could have higher bandwidths in the vicinity of 1.3-μm wavelength.

II. FIBER PROPERTIES

Transmission characteristics were obtained from measurements of the fiber when it was wound on a foam-covered, 11-in.-diameter support drum. The 21.7-km-long fiber was overwound on the drum in layers of about 1 km/layer. This configuration may have introduced some external microbending and curvature effects in the fiber. As a result, the measured transmission loss may be slightly higher and the measured cut-off wavelength for the second propagating mode may be slightly shorter than if the fiber had been perfectly straight.

Figure 1 shows the fiber loss spectrum which was obtained by using an improved automated test system.[3] Curves are drawn on linear scales representing loss (in dB/km) versus wavelength (in μm). The dashed curve was drawn tangent to the measured curve to illustrate the region where loss decreases with a λ^{-4} wavelength dependence that is characteristic of intrinsic Rayleigh scattering. The rapidly increasing slope of the measured curve for wavelengths shorter than 1.1 μm indicates that the cut-off wavelength for the second propagating mode is near 1.1 μm. The 0.14 dB/km water-related loss peak at 1.24 μm is normally about 20 times lower than the water peak (approximately 2.8 dB/km) near $\lambda = 1.39$ μm.[8] The minimum fiber loss values are 1 dB/km in the 1.3-μm region and 0.78 dB/km in the 1.55-μm wavelength region. Intrinsic low loss values of 0.5 dB/km at $\lambda = 1.3$ μm and 0.2 dB/km at $\lambda = 1.55$ μm have been reported in the literature.[9]

Dispersion and bandwidth data were obtained with a measurement set that can automatically select narrow pulses from within the almost continuous 1.06- to 1.7-μm range of wavelengths emitted by a fiber

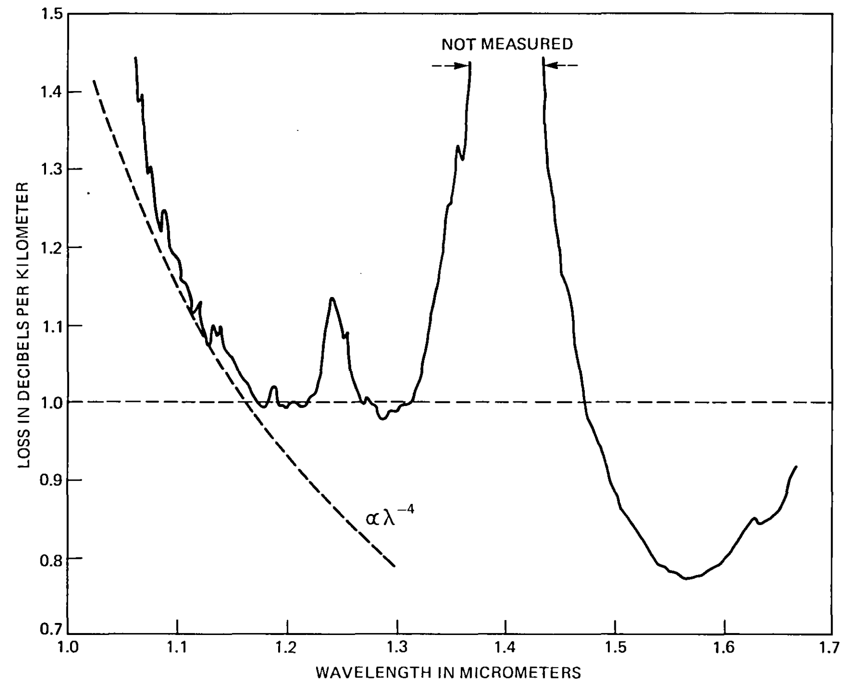


Fig. 1—Fiber loss spectrum plotted on linear scales. The dashed curve is drawn tangent to the measured curve to illustrate the region where loss decreases with a λ^{-4} wavelength dependence.

Raman laser souce.[4] Data are acquired, processed, and displayed by a microcomputer. The system uses an experimental InGaAs photodiode[10] (risetime < 80 ps, bandwidth > 4.25 GHz), which can resolve pulses narrower than the pulse emitted by the laser source (risetime ~ 120 ps, bandwidth ~ 3 GHz).

Figure 2a illustrates group delay spectral measurement results. They were used to calculate the chromatic dispersion spectrum in Fig. 2b, as well as the bandwidth spectrum (the solid line in Fig. 2c). Note, in Fig. 2a, that chromatic dispersion effects in this long fiber length cause large propagation delay changes between pulses at different wavelengths. For example, pulses near $\lambda = 1.35\ \mu m$ arrive almost 60 ns earlier than pulses near $\lambda = 1.12\ \mu m$. Minimum chromatic dispersion occurs at a wavelength near 1.35 μm. The bandwidth spectrum in Fig. 2c applies to a laser source with a 4-nm linewidth, propagating within a fiber with negligible polarization dispersion. However, a slightly elliptical core and or strain-induced birefringence effects could cause propagation delay differences between orthogonally polarized components of the LP(01) mode which would limit the maximum bandwidth in (c) to a value below 1000 GHz-km.[11]

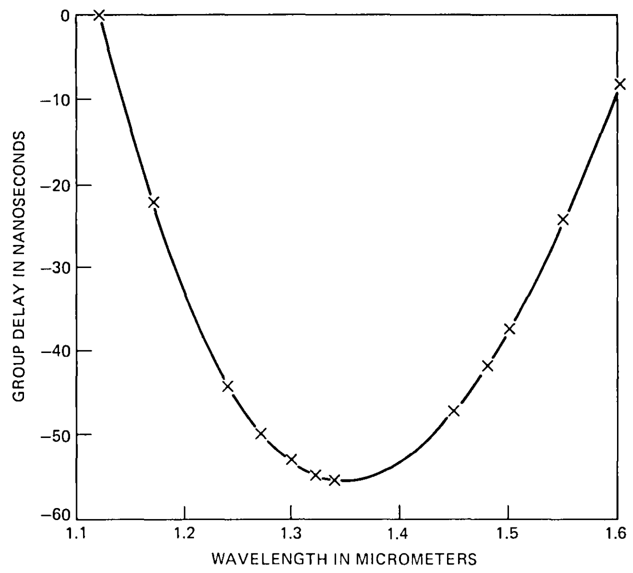


Fig. 2a—Group delay spectrum. The solid curve was fitted to the measured x data points by using a least-mean-square-fit procedure.

The procedure, by which group-delay measurements are used to calculate bandwidth spectra,[5] has proven to be a very convenient way of measuring the performance of short fiber lengths (i.e., as short as 0.5 km) that cannot be characterized from pulse broadening measurements. The 21.7-km, single-mode fiber described is long enough to cause significant pulse broadening which can be used to assess the validity of the bandwidth spectrum in Fig. 2c. The circular points represent bandwidth values that were obtained by transforming pulse

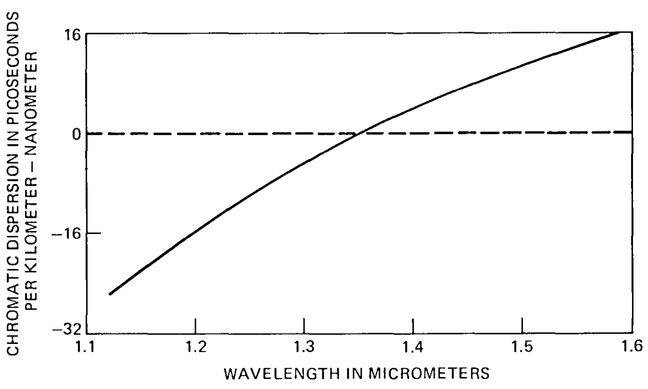


Fig. 2b—Chromatic dispersion spectrum calculated from Fig. 2a. The minimum dispersion wavelength is located at $\lambda = 1.35\ \mu$m.

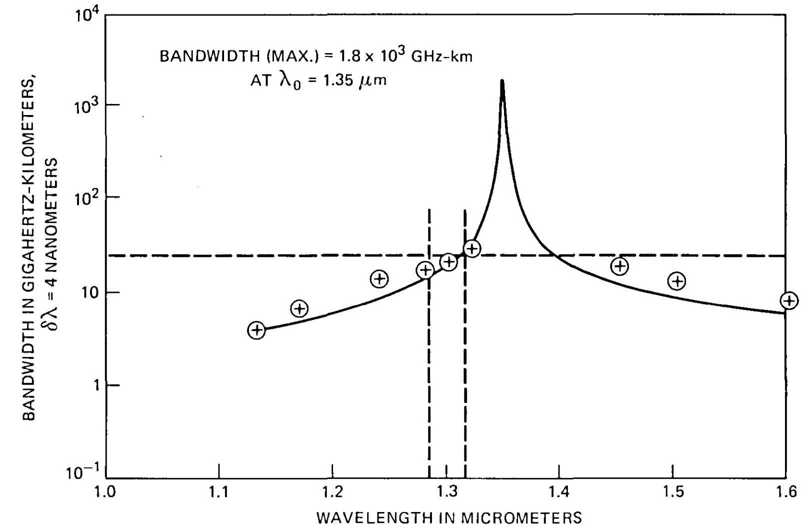


Fig. 2c—Bandwidth spectrum for a source with a 4-nm spectral linewidth. The solid curve was calculated from the values and slopes of the chromatic dispersion spectrum in Fig. 2b. The 0 data points were obtained from pulse broadening measurements. The dashed horizontal line represents the 25-GHz-km bandwidth level which is required to avoid equalization in a regenerator for 274-Mb/s data transmission. The dashed vertical lines represent the range of allowed laser wavelengths for a proposed undersea lightwave cable system.

broadening data. They are in excellent agreement with the solid bandwidth spectrum that was deduced from group delay measurements.

Figure 3 illustrates broadened output pulses at five different wavelengths when the Raman laser output light was filtered to have a 1-nm spectral width. The horizontal time scale is 1 ns/division for $\lambda = 1.06$-μm wavelength and 0.4 ns/division for the other wavelengths. The double-peaked pulse shape at $\lambda = 1.06$ μm is indicative of two-mode propagation.[12] The remaining pulses have only one peak which implies that the cut-off wavelength for the second mode lies between 1.06 and 1.12 μm. That result is consistent with the 1.1-μm-wavelength value which was deduced from the loss spectrum (Fig. 1). Note too, that the pulsewidth becomes narrower as the wavelength increases towards the minimum dispersion wavelength at $\lambda = 1.35$ μm. The pulsewidth at $\lambda = 1.6$ μm is broader than the one at $\lambda = 1.3$ μm because the former is displaced further from $\lambda = 1.35$ μm.

Figure 4 illustrates impulse responses obtained at $\lambda = 1.3$ μm by using 1-nm- and 5-nm-wide spectral filters for (b) and (c), respectively. The resolution of the output pulse shapes was limited by the 100-ps risetime and 3-GHz bandwidth of the input pulse in (a). Fiber bandwidths were obtained from output/input FFT (fast Fourier transform)

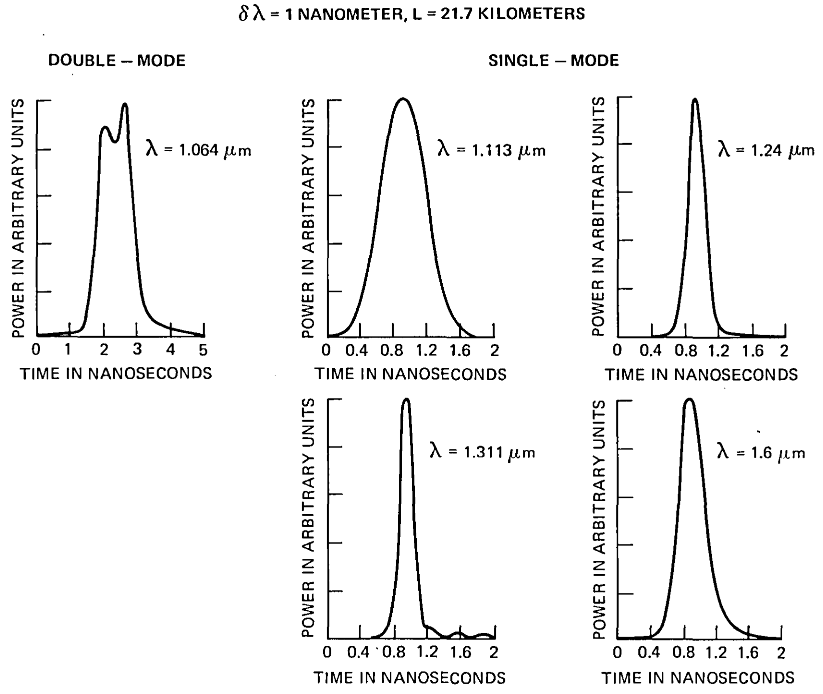


Fig. 3—Broadened output pulses at five different wavelengths. The horizontal time scale is 1 ns/division at $\lambda = 1.06\ \mu m$ and 0.4 ns/division at the other wavelengths. The twin-peaked pulse at $\lambda = 1.06\ \mu m$ indicates double-mode propagation.

ratios calculated from the pulse shapes. The inset graph plots bandwidth results versus the inverted spectral width of the filtered Raman laser source. They confirm that the fiber bandwidth increases linearly[5] with the inverse of the source linewidth, from 5 nm to 1 nm, because the 1.3-μm test wavelength is significantly different from the minimum chromatic dispersion wavelength at 1.35 μm. However, the linear extrapolation cannot be extended indefinitely to very narrow linewidths which would make the fiber bandwidth very large. In practical situations, the maximum bandwidth is limited by polarization dispersion effects caused by small propagation delay differences between orthogonally polarized components of the LP(01) mode.[11] The maximum bandwidth measured in this study was 90 GHz-km in 1.32-μm-wavelength light with a $\delta\lambda$ = 1-nm-rms linewidth. A bandwidth of 71 GHz-km was independently measured using an Nd:YAG laser which has a spectral linewidth $\delta\lambda < 0.05$ nm.[13] Therefore, bandwidths did not scale with source linewidths less than 1 nm, implying that the maximum bandwidth of this fiber is about 90 GHz-km. This limit may be imposed by polarization dispersion. Further investigation will be required to validate this conjecture and to determine whether polarization dispersion effects, if any, are caused by core ellipticity or by strain-induced birefringent effects at the core-cladding interface.

Semiconductor injection laser sources typically have 4-nm spectral linewidths in the 1.3-μm wavelength region. Lasers of this type are being proposed for use in undersea telecommunication systems whose repeater spacings will be approximately 35 km. Results in Fig. 2c indicated that the normalized fiber bandwidth is 21 GHz-km (actual bandwidth would be 600 MHz for a 35-km propagation length). The next section will show that those bandwidth characteristics should be suitable for use in systems transmitting at 274-Mb/s data rates.

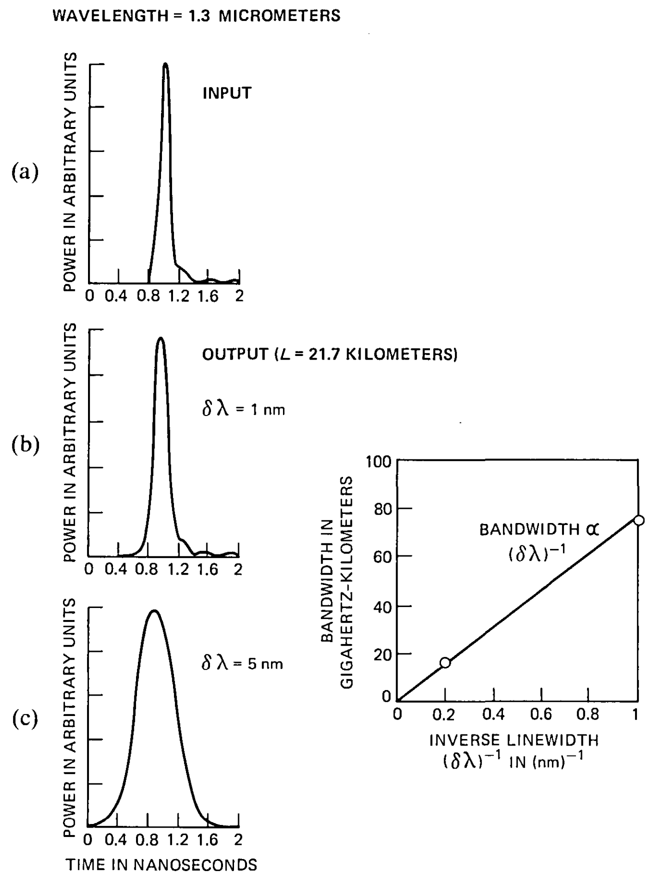


Fig. 4—Impulse response at λ = 1.3 μm. The horizontal time scale is 0.4 ns/division. (a) Input pulse (risetime ~120 ps, bandwidth ~3 GHz). (b) Output pulse measured within a 1-nm source spectral linewidth. (c) Output pulse measured within a 5-nm source spectral linewidth. The inset graph plots bandwidth versus reciprocal spectral width of the source.

III. ESTIMATES OF SYSTEM PERFORMANCE

In a pulse-code-modulation communication system, pulse spreading causes intersymbol interference in the form of overlapping pulses. In principle, those pulses can be separated by equalization or high-frequency enhancement in the receiver. However, that enhancement also increases receiver noise which reduces the receiver sensitivity relative to the dispersion-free case. Therefore, system degradation because of dispersion effects can be assigned noise penalties, in dB, which add to fiber transmission losses to give the total lightwave cable loss. Repeater spacings can then be calculated by comparing cable losses with the difference between the optical power levels available and the power levels required for a specified error probability at various data rates.

Optical power penalties, D_p, caused by dispersion are calculated as follows:[7]

$$D_p(\text{in dB}) = 5\log\left\{\frac{1}{J_3}\left[J_3 + \left[\frac{1}{b_2} + C_1\right]B^2J_4 + \left[\frac{C_1}{b^2} + C_2\right]B^4J_5 + \left[\frac{C_2}{b^2} + C_3\right]B^6J_6 + \left[\frac{C_3}{b^2}\right]B^8J_7\right\}, \quad (1)$$

where

B =	transmission bit rate,
$b = 1.2$ GHz =	electrical 3-dB modulation bandwidth of the laser source,
$C_1 - C_3$ =	coefficients that are used to approximate the fiber's baseband frequency response, $\lvert H_c\rvert$, with a polynomial as follows: $1/\lvert H_c\rvert^2 = 1 + C_1f^2 + C_2f^4 + C_3f^6$,
$J_3 - J_7$ =	tabulated coefficients for equalizing the receiver passband from a non-return to zero (nrz) input to a raised cosine signal spectrum.

Figure 5 illustrates how to relate fiber transmission bandwidth with the dispersion power penalty, D_p. The vertical axis on the left applies to the D_p versus wavelength curve, while the vertical axis on the right corresponds to the bandwidth spectrum. The magnitude of D_p increases approximately quadratically with L. The illustrated spectral curve is applicable to 274-Mb/s data-rate transmission within a 40-km cable length, which is within the range of proposed repeater spacings for future undersea systems. Comparisons between the two curves in Fig. 5 show that 274-Mb/s transmission rates require that the fiber bandwidth be greater than 274 MHz to keep the system dispersion power penalty below 1 dB, whereas the fiber bandwidth has to be greater than 750 MHz to keep the dispersion penalty below 0.2 dB. The former specification can be generalized in the following interesting

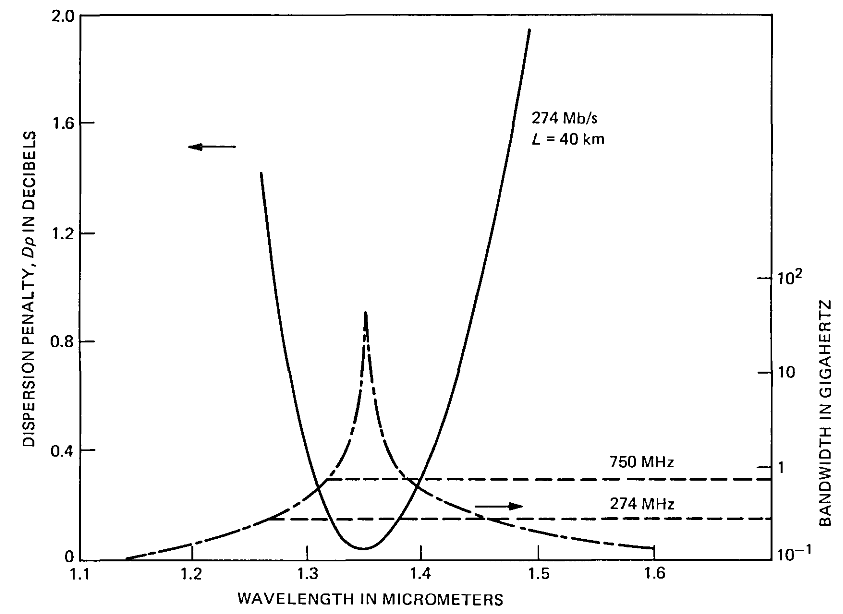


Fig. 5—Dispersion power penalty spectrum (solid curve) for 274-Mb/s systems (left-hand scale). Bandwidth spectrum (dashed curve, right-hand scale). The dashed horizontal lines show that the fiber bandwidth must be greater than 274 or 750 MHz to keep the corresponding dispersion penalty below 1 dB or 0.2 dB, respectively.

way. If the bandwidth of a fiber is equal to the bit rate of a system, then the resultant dispersion power penalty will be about 1 dB because of intersymbol interference. The $D_p = 0.2$ dB specification would be very desirable to meet because the penalty is small enough to ensure that no equalization would be necessary in any of the numerous regenerators that would be required for a long distance telecommunication system.

Results similar to those shown in Fig. 5 were generated for different fiber lengths so that chromatic-dispersion-limited repeater spacings could be calculated as a function of wavelength. Results shown in Fig. 6 indicate solid curves which apply for 274-Mb/s data rates, as well as dotted curves which apply for 548-Mb/s data rates. The vertical dashed lines indicate the wavelength limits, $1.3 \pm 0.015\ \mu$m, which give a margin for source wavelength deviations around a 1.3-μm nominal system wavelength. The outer solid and dotted curves were calculated to keep $D_p < 1$ dB, while the inner curves were calculated to restrict $D_p < 0.2$ dB. The results show that repeater spacings for $B = 274$ Mb/s could range between 24 km and 54 km, depending on the dispersion penalty allowed. By comparison, for $B = 548$ Mb/s, the repeater spacing could be 24 km if $D_p = 1$ dB, but would be much shorter if smaller dispersion penalties are required.

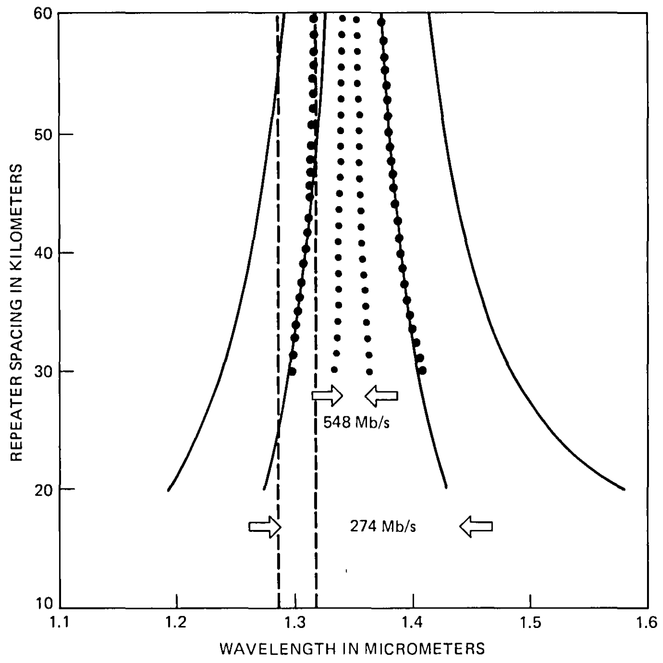


Fig. 6—Chromatic-dispersion-limited repeater spacings plotted versus wavelength. The solid curves apply to 274-Mb/s systems and the dashed curves apply to 548-Mb/s systems. The inner (solid and dotted) curves were calculated to maintain $D_p = 1$ dB; while the outer curves maintain $D_p = 0.2$ dB. The dashed vertical lines represent the allowed range of laser wavelengths.

The curves in Fig. 6 indicate that the 21.7-km fiber under study should meet the bandwidth requirements for 274-Mb/s systems with 35-km repeater spacings provided that each regenerator is individually equalized. Potential repeater spacings could be significantly lengthened if the minimum dispersion wavelength could be moved closer to the operating system wavelength at 1.3 μm.

IV. SUGGESTIONS FOR OPTIMIZATION

This section describes results of a numerical study to determine a more optimal structure that would make future fibers have higher bandwidths near $\lambda = 1.3$ μm. Results were generated from numerically exact solutions for the LP(01) propagating mode of the scalar wave equation as indicated in a companion publication.[5] Figure 7 summarizes the results with curves of bandwidth (source linewidth = 4 nm) at $\lambda = 1.3$ μm as a function of fiber core diameter, d. A step-index profile shape was assumed for various core-to-cladding index differences, Δ, which served as variable parameters for the curves. If future experimental fiber profiles are found to conform to a universal shape, other than step-index, then future parametric studies could be modified accordingly. Therefore, the curves in Fig. 7 should be viewed as qualitative. They indicate that fiber bandwidths at $\lambda = 1.3$ μm can be increased by increasing $V = (\pi d/\lambda)n\sqrt{2\Delta}$ as long as single-mode behavior is maintained. The range of allowed values lies below the diagonal dashed lines which correspond to constant V-values of 2.4 and 2.7 at $\lambda = 1.3$ μm. The $V = 2.4$ value is the theoretical limit for step-index, single-mode fibers. However, recent measurements indicate that $V = 2.7 = (\pi d/\lambda_{co})n\sqrt{2\Delta}$ is a more accurate value for calculating the cut-off wavelength, λ_{co}, for single-mode operation in experimental fibers.[14]

Fibers with relatively small-core diameters, d, and large index differences, Δ, offer good mode confinement and very good resistance to curvature-induced cabling losses. The 21.7-km fiber, described in this paper, has a nominal core diameter, $d \approx 7.2$ μm, an index difference, $\Delta \approx 0.0051$, and a cut-off wavelength, $\lambda_{co} \approx 1.2$ μm. Higher bandwidths should result if V-values for future fibers are increased by about 8 percent, which is the maximum allowed change that would still maintain single-mode behavior at system wavelengths near 1.3 μm. Results in Fig. 7 indicate that bandwidths will increase most if Δ is kept constant and the fiber diameter is increased. More optimal parameters for future fibers would then be $\Delta \approx 0.0051$ and $d = 7.8$ μm.

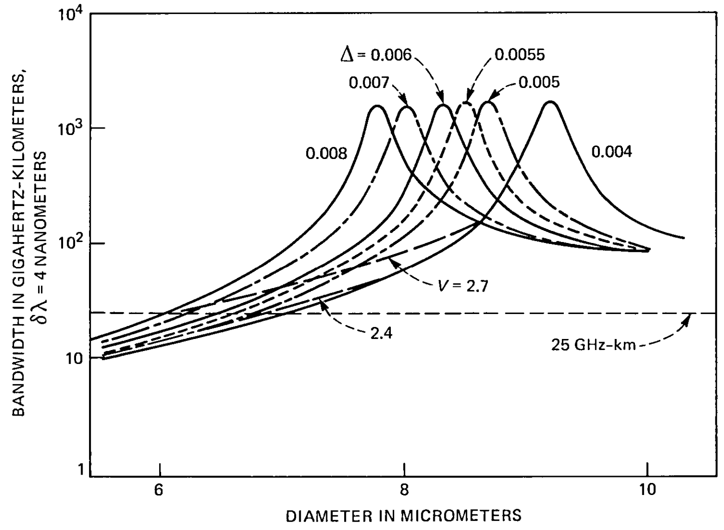


Fig. 7—Parametric study of step-index, single-mode fibers. Bandwidth (source linewidth = 4 nm) at $\lambda = 1.3$-μm wavelength is plotted as a function of fiber core diameter, d, with index difference, Δ, as the variable parameter. The dashed diagonal lines indicate constant V-values of 2.4 and 2.7.

V. CONCLUSIONS

The transmission characteristics of a 21.7-km-long, single-mode fiber have been extensively studied. Prototype automated measuring systems were used to determine the fiber properties across the 1.06- to 1.7-μm wavelength spectrum. All measurements were made when the coated fiber was overwound on an 11-in.-diameter, foam-covered drum. This method of supporting the fiber is not ideal because it may have induced external bending effects that could have raised fiber losses and reduced its cut-off wavelength.

The transmission loss spectrum showed minimum loss values of 1 dB/km and 0.78 dB/km at wavelengths near 1.3 and 1.55 μm, respectively. A small water-related loss peak was evident at $\lambda = 1.24$ μm and implies that the OH absorption peak at $\lambda = 1.39$ μm is approximately 2.8 dB/km. The shape of the fiber loss spectrum curve was used to infer that its cut-off wavelength is near 1.1 μm.

Dispersion and bandwidth characteristics were determined by using two independent measurement techniques. Group delay spectral measurements were used to determine the chromatic dispersion spectrum and locate the minimum dispersion wavelength at 1.35 μm. The bandwidth spectrum was calculated from the dispersion spectrum and was found to be in excellent agreement with pulse broadening effects that were measured at different wavelengths using a variety of spectral filters. Light filtered with a linewidth of 5 nm centered around 1.3 μm was used to closely approximate source characteristics proposed for undersea lightwave cable applications. The normalized fiber bandwidth was measured to be 21 GHz-km with a 4-nm linewidth source centered around 1.3-μm wavelength. A higher bandwidth was measured for a source linewidth of 1 nm. The resultant 90-GHz-km bandwidth was close to a value that was measured with a laser whose spectral linewidth was less than 0.05 nm. One possible explanation for the lack of linear dependence of bandwidth on inverse source linewidth might be that polarization dispersion effects caused by core ellipticity or strain-induced birefringence may have limited the maximum bandwidth of this fiber to about 90 GHz-km, independent of the spectral characteristics of the source. However, the limitation is academic because the measured bandwidth is still adequate for a 274-Mb/s system.

Estimates of system degradation because of intersymbol interference were made by calculating the dispersion penalty or, equivalently, the additional power required in the regenerator to equalize the receiver passband. These calculations were related to fiber bandwidth spectrum characteristics through their dependence on baseband frequency response. One interesting result is that the dispersion power penalty is approximately 1 dB when the 3-dB fiber bandwidth equals the bit rate of the system. By comparison, a 750-MHz fiber bandwidth is required to maintain the dispersion penalty below 0.2 dB for a 274-Mb/s system data rate. The latter penalty is small enough to ensure that no equalization would be required in any of the regenerators. The 21.7-km fiber described in this paper should be able to meet system requirements for transmission at a 274-Mb/s rate between repeaters separated by 35 km.

A numerical study was used to show how the bandwidth of a fiber depends on its core diameter, d, and core-to-cladding index difference, Δ. Results indicate that the bandwidth performances of future single-mode fibers could be significantly improved by increasing their core diameters ($d \approx 7.8\ \mu$m; $\Delta \approx 0.0051$). If that is done, potential repeater spacings might be increased to 75 km, which is the upper limit for 274-Mb/s systems with 0.5-dB/km cabled fiber losses and -38 dBm minimum detectable signals at the receivers. Additional limitations because of mode-partition-noise generated in laser sources have not been considered in this paper.

VI. ACKNOWLEDGMENT

The authors thank T. P. Lee for providing the detector, S. Lumish for computational assistance in obtaining numerical data, and P. D. Lazay for helpful discussions.

REFERENCES

1. P. K. Runge, "High Capacity Optical-Fiber Undersea Cable System," Tech. Dig, Conf. on Laser and Electro-Optical Syst., San Diego, California, February 26–8, 1980.
2. A. D. Pearson, "Fabrication of Single-Mode Fiber at High Rate in Very Long Lengths for Submarine Cable," Tech. Dig. Third Int. Conf. on Integ. Opt. and Opt. Fib. Commun., San Francisco, California, April 27–29, 1981.
3. J. Stone, H. E. Earl, and R. M. Derosier, "A Measurement Set for Optical Fibers," private communication.
4. L. G. Cohen, P. Kaiser, and C. Lin, "Experimental Techniques for Evaluation of Fiber Transmission Loss and Dispersion," *Proc. IEEE* (Special Issue on Optical-Fiber Communications), *68* (October 1980), pp. 1203–9.
5. L. G. Cohen, W. L. Mammel, and S. Lumish, unpublished work.
6. D. Gloge et al., "High-Speed Digital Lightwave Communication Using LEDs and PIN Photodiodes at 1.3 μm," B.S.T.J., *59,* No. 8 (October 1980), pp. 1365–82.
7. L. G. Cohen and S. Lumish, "Effects of Water Absorption Peaks on Transmission Characteristics of LED-Based Lightwave Systems Operating Near 1.3-μm Wavelength," Tech. Dig. Third Int. Conf. on Integ. Opt. and Opt. Fib. Commun., San Francisco, California, April 27–29, 1981.
8. H. Osanai et al., "Effect of Dopants on Transmission Loss of Low-OH Content Optical Fibers," Electron. Lett., *12* (October 1976), pp. 549–50.
9. T. Miya et al., "Ultimate Low-Loss Single-Mode Fibre at 1.55 μm," Electron. Lett., *15* (February 1979), pp. 106–8.
10. T. P. Lee et al., "Very-High-Speed Back-Illuminated InGaAs/InP PIN Punch-Through Photodiodes," Electron. Lett., *17* (June 1981), pp. 431–2.
11. H. Tsuchiya and N. Imoto, "Dispersion-Free Single-Mode Fibre in 1.5 μm Wavelength Region, "Electron. Lett., *15* (July 1979), pp. 476–8.
12. L. G. Cohen et al., "Propagation Characteristics of Double-Mode Fibers," B.S.T.J., *59,* No. 6 (July–August 1980), pp. 1061–72.
13. W. A. Reed et al., "Bandwidth Measurement of a 20 km Length of Single-Mode Fiber," private communication.
14. P. D. Lazay, private communication.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Electromagnetic Fields, Field Confinement, and Energy Flow in Dispersionless Single-Mode Lightguides With Graded-Index Profiles

By U. C. PAEK, G. E. PETERSON, and A. CARNEVALE

(Manuscript received February 13, 1981)

It is shown by numerical solution of Maxwell's equations that, for a given wavelength, the degree of confinement of the electromagnetic field to the core of a graded-index, single-mode, optical-fiber can be optimized by the proper choice of the radial variation of the index. Such confinement of the energy to the core helps alleviate loss. The fibers considered have zero total dispersion bandwidths in excess of 100 GHz·Km, at wavelengths between 1.3 μm and 1.55 μm.

I. INTRODUCTION

Our earlier work described a method of designing single-mode lightguides with zero total dispersion by varying the index profile in the core.[1] In the range of wavelengths between 1.3 μm and 1.55 μm, bandwidths in excess of 100 GHz·Km are attainable by balancing material dispersion with waveguide dispersion.[2,3] However, one of the serious difficulties with single-mode fibers is microbending loss. We know that the microbending loss for the case of the step-index, single-mode fiber is proportional to λ^2/Δ^3.[4,5,6] Here, λ refers to the operating wavelength and Δ to the relative index difference which is defined as:

$$\Delta = (N_{\text{core}} - N_{\text{clad}})/N_{\text{core}}. \tag{1}$$

Now the design of a single-mode fiber must be such that it prevents the field of the fundamental mode (HE_{11}) from extending well into the cladding. In other words, the electromagnetic field must be tightly confined to the core. To this end, two methods can be considered: one is to increase Δ, which is a common and prevailing method, and the other is to change the index profile in the core. This work will focus on the latter case by assuming an α index profile, where $N = N_1(1 - \Delta r^{\alpha})$. Reducing microbending by profile design might be advantageous because the TE and TM modes can be maintained well beyond the cutoff point of a step-index, single-mode fiber and the manufacturing tolerances are relaxed. Note that $N_1 = N_{\text{core}}$ and $N_2 = N_{\text{clad}}$.

For a single-mode lightguide having a radially inhomogeneous core, it is usually not possible to obtain analytical solutions of closed form for Maxwell's equations. Hence, to attain vector electromagnetic field distributions of the HE_{11} mode and to determine accurate propagation characteristics of a single-mode fiber, we used a numerical method to solve the governing equations.[7,8]

II. THEORY

Our method of solving Maxwell's equations for lightguides has been described in our earlier publications. However, we did not consider the cladding fields in much detail. For the work to be described in this paper, this is essential. Thus, we develop the necessary mathematical expressions.

In an optical fiber having a permittivity ϵ and permeability μ, we assume that the outer diameter D is much larger than the core

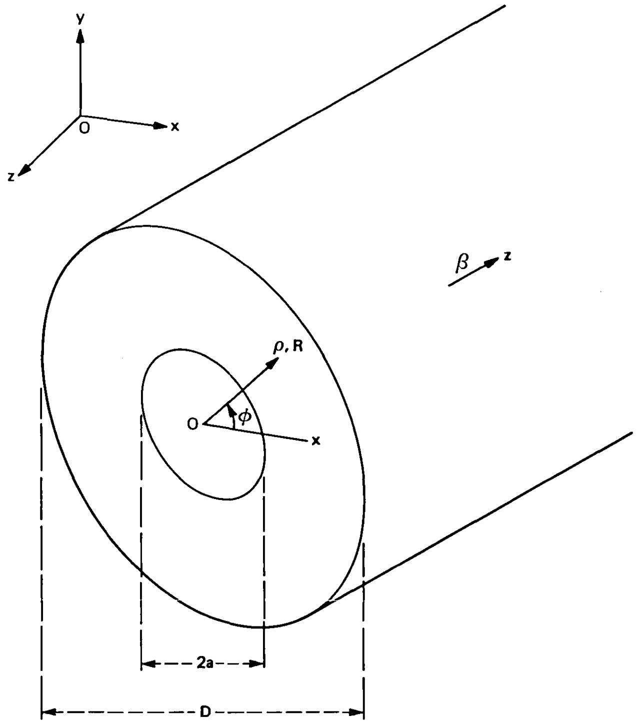


Fig. 1—A cross-section of a fiber and its cylindrical coordinate system.

diameter $2a$. In a cylindrical coordinate system, for a position vector $\boldsymbol{r}$ having as its components $\{R, \phi, z\}$, the corresponding components of electric and magnetic fields can be written as $E = \{E_R, E_\phi, E_z\}$ and $H = \{H_R, H_\phi, H_z\}$ (see Fig. 1). However, in obtaining a complete set of vector solutions, it is only necessary to find the tangential components $\{E_\phi, E_z\}$ and $\{H_\phi, H_z\}$ since the radial components E_R and H_R are linear combinations of the other components. In particular,

$$E_R = -\frac{Z_o}{\rho N^2} H_z + \frac{Z_o N_e}{N^2} H_\phi, \tag{2}$$

$$H_R = \frac{1}{Z_o} E_z - \frac{N_e}{Z_o} E_\phi . \tag{3}$$

In addition, the tangential field components are continuous through the core-cladding interface and this simplifies the mathematics of the boundary value problem.

In the above equation, Z_o is the wave impedance defined by $(\mu/\epsilon)^{1/2}$ and N is the index of refraction and a function of R. The effective refractive index N_e is defined by two quantities, β and k, where β is the propagation constant along the fiber axis and $k = 2\pi/\lambda$; ρ is a dimensionless quantity defined by Rk. The fundamental HE_{11} mode propagates in the fiber when the angular mode number M equals 1. Moreover, $V < V_c$ must be satisfied. Here V is the normalized frequency and V_c is the cutoff frequency for a single-mode propagation. The V value is defined by

$$V = \frac{2\pi a_{\mathrm{opt}}}{\lambda} \sqrt{N_1^2 - N_2^2} . \tag{4}$$

The input data, a_{opt}, is the optimum core radius that will give zero total dispersion for a given α, Δ, and λ.

In the most general case, there are two possible solutions to Maxwell's equations for a guided mode in a lightguide. A general solution will be a sum of these two vector solutions. We introduce variable Γ_i to establish the following relationship with the tangential field vectors $\{E_\phi, E_z\}$ and $\{H_\phi, H_z\}$.

$$\Gamma_i = \begin{Bmatrix} \Gamma_1 \\ \Gamma_2 \\ \Gamma_3 \\ \Gamma_4 \end{Bmatrix} = \begin{Bmatrix} E_z \\ -iZ_o\rho H_\phi \\ \rho E_\phi \\ -iZ_o H_z \end{Bmatrix} \tag{5}$$

From eq. (5) we denote the two solutions in cores Γ_{i1} and Γ_{i2}. Since our earlier work gave a detailed description of the computation of Γ_{i1} and Γ_{i2}, we will avoid repetition of the procedure for the solution in the core region.[1] In the cladding region, the two solutions designated by Γ_{i3} and Γ_{i4} are given by the following expressions:[7]

$$\Gamma_{i3} = W_1(\xi)\cdot\begin{bmatrix} N_e^2 - \kappa(c) \\ \kappa(c)\gamma_1(\xi) \\ N_e \\ 0 \end{bmatrix} \tag{6}$$

and

$$\Gamma_{i4} = W_1(\xi)\cdot\begin{bmatrix} 0 \\ N_e \\ \gamma_1(\xi) \\ N_e^2 - \kappa(c) \end{bmatrix}, \tag{7}$$

where $\kappa(c)$ is the dielectric constant in the cladding, and

$$\left.\begin{aligned} \xi &= [N_e^2 - \kappa(c)]^{1/2}\rho_i \\ W_1(\xi) &= K_1(\xi)/[N_e^2 - \kappa(c)] \\ \gamma_1(\xi) &= \xi\cdot K_1'(\xi)/K_1(\xi) \end{aligned}\right\}, \tag{8}$$

where K_1 is a modified Bessel function of the second kind and its prime denotes differentiation with respect to ξ. (Note that ρ_i is the value of ρ at the interface.)

The total solution Γ can be written in the core and cladding region separately.
In the core region, Γ is expressed by

$$\Gamma = A_1\Gamma_{i1} + A_2\Gamma_{i2}, \tag{9}$$

and in the cladding, Γ is expressed by

$$\Gamma = A_3\Gamma_{i3} + A_4\Gamma_{i4}, \tag{10}$$

where A_j is an arbitrary constant, $j = 1, 2, 3, 4$.

To calculate the field function Γ, we require basically four input data, namely, λ, the optimum core radius a_{opt}, N_e, and N. Among those parameters, calculation of N_e has been described in detail in Ref. 1. The material dispersion effect is incorporated with Maxwell's equations to achieve a high degree of accuracy for N_e. This is needed to acquire the precise eigenfunctions from eqs. 9 and 10.

In the design of a single-mode fiber, Δ is usually specified as an input data. It is rather small, ranging from 0.002 to 0.008, since the cladding of the fiber is generally made of a high-silica glass. The dispersive character of the cladding is well known.[9] Therefore, for a given Δ, the index of the core center N_1 can be expressed in terms of N_2 by

$$N_1 = \frac{N_2}{(1 - \Delta)}. \tag{11}$$

The dispersive properties of the N_2 in eq. (11) can be described by a modified Sellmeier formula.[1,7]

$$N_2 = C_0 + C_1\lambda^2 + C_2\lambda^4 + \frac{C_3}{(\lambda^2 - l)} + \frac{C_4}{(\lambda^2 - l)^2} + \frac{C_5}{(\lambda^2 - l)^3}, \quad (12)$$

where $l = 0.035$. The coefficients C_i are given in our previous work.[7]

For the index profile, we use a well-known formula that is particularly useful in fiber design.

$$N = N_1 \left[1 - \Delta \left(\frac{R}{a}\right)^{\alpha}\right]. \quad (13)$$

Finally, the dispersion of the index N will be determined by substituting N_1 in eq. (13) with eqs. (11) and (12).

At the core-cladding interface, Γ must satisfy the continuity condition of the tangential field components. Consequently, this yields a set of simultaneous equations

$$\left.\begin{aligned} A_1\Gamma_{11} + A_2\Gamma_{12} &= A_3\Gamma_{13} + A_4\Gamma_{14} \\ A_1\Gamma_{21} + A_2\Gamma_{22} &= A_3\Gamma_{23} + A_4\Gamma_{24} \\ A_1\Gamma_{31} + A_2\Gamma_{32} &= A_3\Gamma_{33} + A_4\Gamma_{34} \\ A_1\Gamma_{41} + A_2\Gamma_{42} &= A_3\Gamma_{43} + A_4\Gamma_{44} \end{aligned}\right\}. \quad (14)$$

To compare the field distributions, it is convenient to introduce the following normalized variables into eq. (5).

$$\left.\begin{aligned} \bar{E}_z &= \frac{E_z}{E_\phi(0)} \\ \bar{E}_\phi &= \frac{E_\phi}{E_\phi(0)} \\ \bar{E}_R &= \frac{E_R}{E_R(0)} \end{aligned}\right\}, \quad (15)$$

$$\left.\begin{aligned} \bar{H}_z &= \frac{H_z}{H_\phi(0)} \\ \bar{H}_\phi &= \frac{H_\phi}{H_\phi(0)} \\ \bar{H}_R &= \frac{H_R}{|H_R(0)|} \end{aligned}\right\}. \quad (16)$$

III. ELECTROMAGNETIC FIELDS FOR THE HE$_{11}$ MODE IN DISPERSIONLESS SINGLE-MODE LIGHTGUIDES

We begin our study by considering a germania-doped silica lightguide with $\Delta = 0.002$ and $\lambda = 1.33\ \mu$m. The profile parameters examined are $\alpha = 100$, 2, and 1. Thus, we span the range from rectangular through parabolic to triangular index functions. In Table I we calculate the values for the radii to make these lightguides dispersion-free.

Table I—Radii values for dispersion-free lightguides

α	$a_{\text{opt}(\mu m)}$
100	4.142
2	5.725
1	6.294

The normalized electromagnetic fields as a function of normalized radii for these three cases are shown in Figs. 2a, 2b, and 2c. Note that in all cases the z fields are much smaller than the other field components. In fact, these fields are less than 3 percent of the tangential fields. The magnitude of the R and ϕ components of the electric and magnetic fields for a given α are all essentially the same. This is not likely if the index profile becomes more complex. For example, a profile

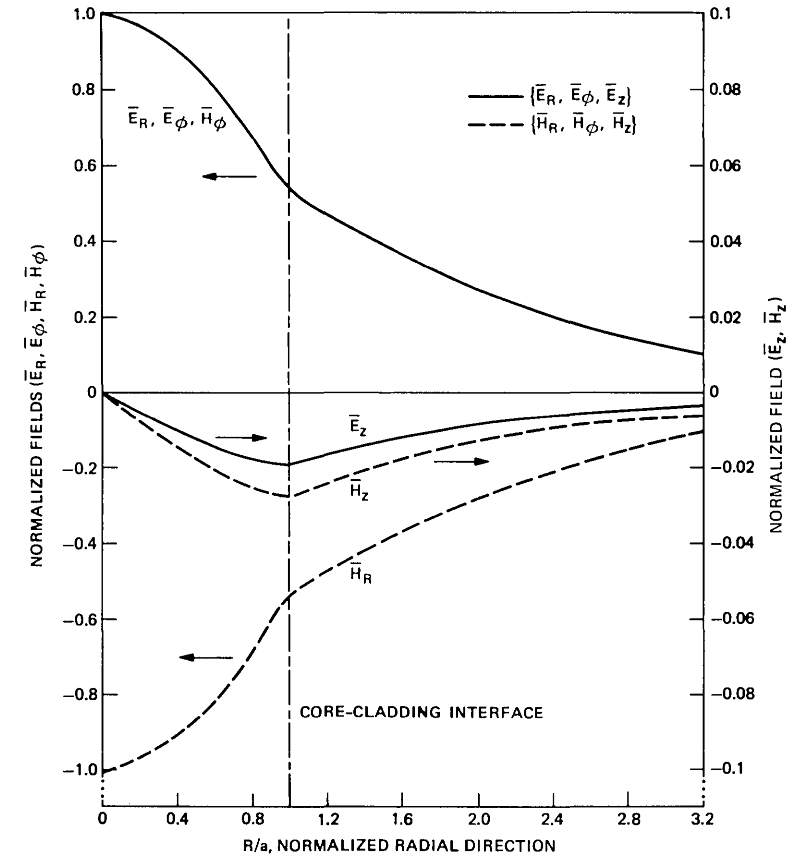


Fig. 2a—Normalized field distributions of the HE_{11} mode in a single-mode lightguide, where $\Delta = 0.002$, $\lambda = 1.33\ \mu m$, $\alpha = 100$, and $a_{\text{opt}} = 4.142\ \mu m$.

containing a central "burn out" and ripples, which would be characteristic of modified chemical vapor deposition (MCVD) profiles, would not have such a simple relationship between field components.

An interesting observation from Fig. 2 is that the slopes of the field components at the core-cladding interface change with α. For $\alpha = 100$, the field distribution near the interface forms a cusp, but it rounds progressively as α decreases. This is due solely to the index distribution in the core of the single-mode fiber.

Figure 3 shows the normalized transverse components of the electromagnetic field as a function of radial distance for the three α values. The curves are essentially identical up to $R = 3$ μm. However, beyond that distance they deviate appreciably. Also shown, by the vertical lines, are the optimum radii.

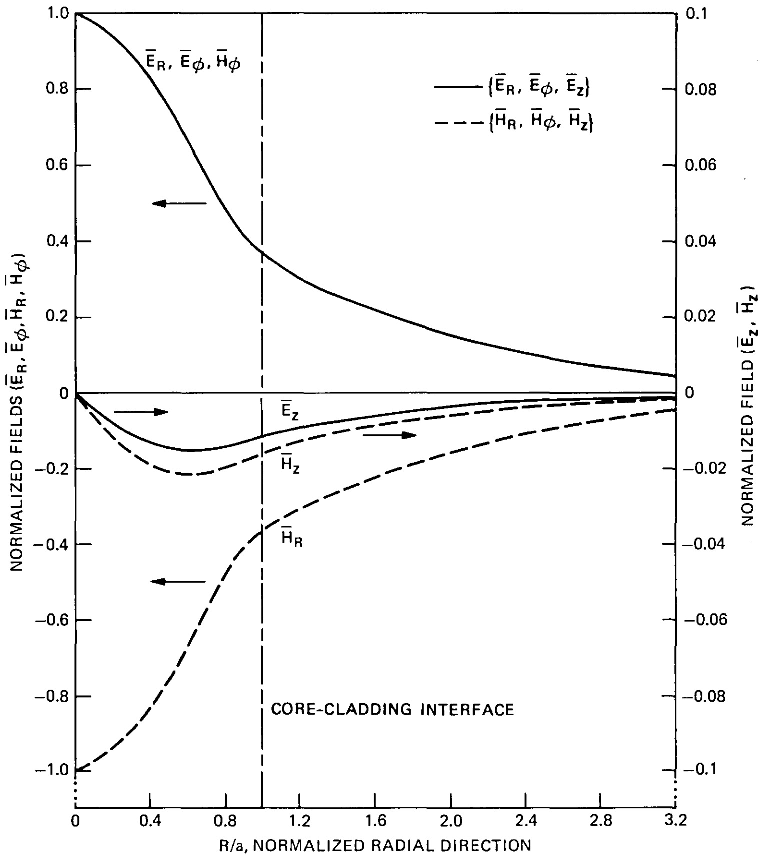


Fig. 2b—Normalized field distributions of the HE_{11} mode in a single-mode lightguide, where $\Delta = 0.002$, $\lambda = 1.33$ μm, $\alpha = 2$, and $a_{opt} = 5.725$ μm.

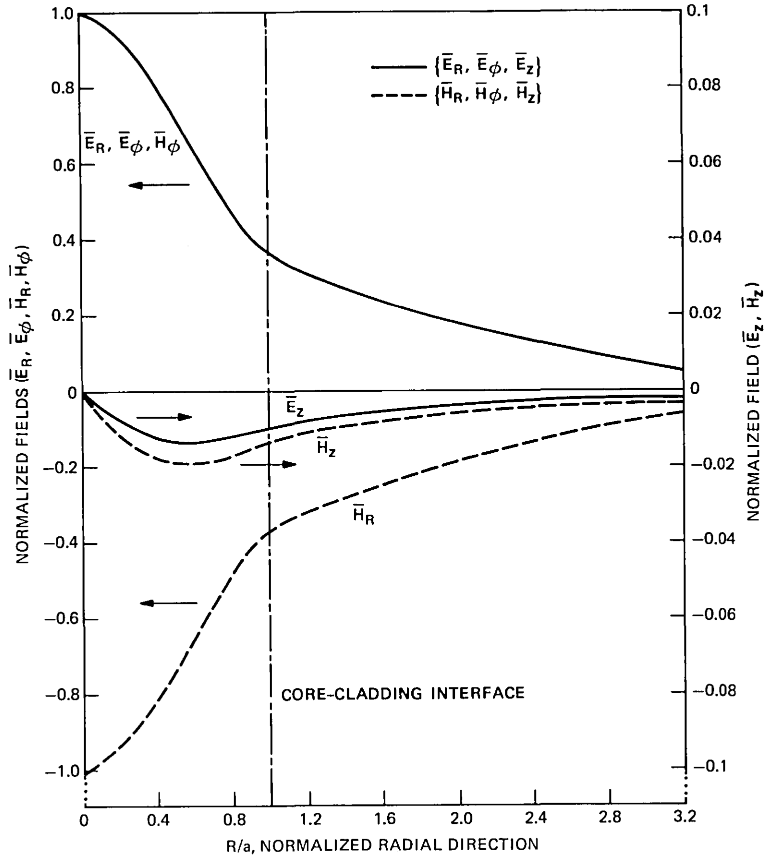


Fig. 2c—Normalized field distributions of the HE_{11} mode in a single-mode lightguide, where $\Delta = 0.002$, $\lambda = 1.33\ \mu m$, $\alpha = 1$, and $a_{opt} = 6.294\ \mu m$.

IV. ENERGY FLOW FOR THE HE_{11} MODE IN DISPERSIONLESS SINGLE-MODE LIGHTGUIDES

So far we have only considered the field in the fiber. However, in experimental practice it is more convenient to know the field intensity, which is the amount of energy flowing through the cross-section of the fiber. This can be calculated from the Poynting vector S in Ws/cm^2. The Poynting vector in the z direction, S_z is given by,

$$S_z = \tfrac{1}{2}(E_R H_\phi^* - E_\phi H_R^*), \tag{17}$$

where * indicates the complex conjugate of the variable.

We define the normalized Poynting vector I by

$$I = S_z/S_z(0). \tag{18}$$

Figure 4 shows the curves of I versus the normalized radial coordinate

R/a for three different α values. The normalized Poynting vector (field intensity) falls off more rapidly with normalized radius for lower α values.[10] As in Section III, when $\alpha = 100$ the Poynting vector develops a cusp at the core-cladding interface. We also note a near identity of the $\alpha = 1$ and $\alpha = 2$ curves. Thus, these two have nearly the same focusing power.

V. DEGREE OF FIELD CONFINEMENT

We know that the degree of field confinement in a fiber is related to its microbending loss.[4,5] In the fabrication of single-mode fiber cables for undersea applications, microbending loss has been one of the factors that determines the performance of the cable.

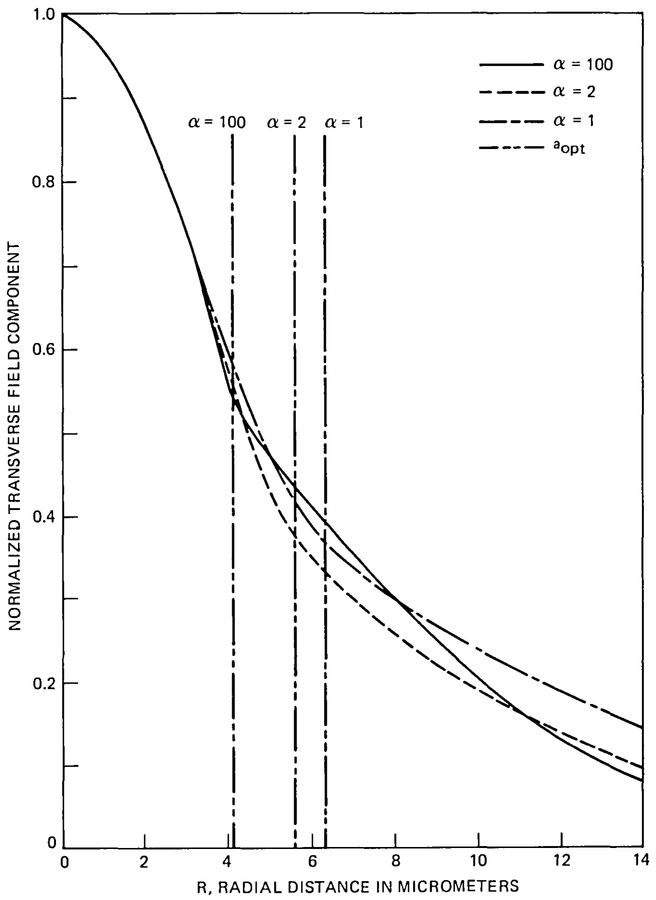


Fig. 3—Normalized field components versus radial distance for three different values of α.

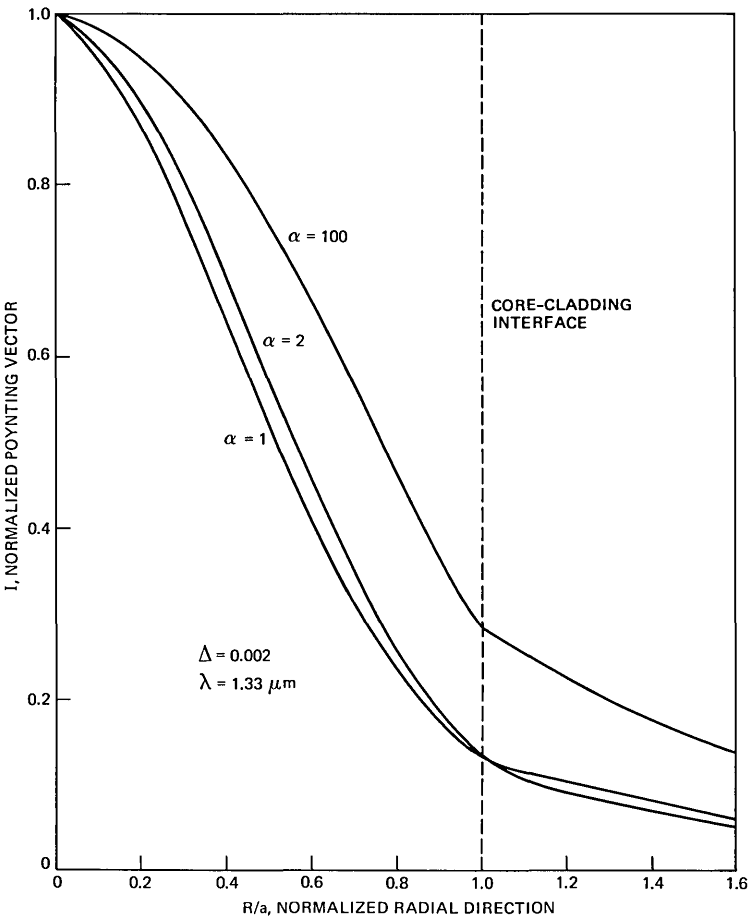


Fig. 4—Normalized Poynting vector versus normalized radial coordinate for three different values of α.

Figures 3 and 4 indicate that the field or power distribution is largely dependent on the index profile. To quantify the focusing power or confinement of a lightguide, we introduce a parameter Φ defined by:

$$\Phi = \frac{\int_0^{a_{\text{opt}}} S_z R dR}{\int_0^{\infty} S_z R dR}. \tag{19}$$

The parameter Φ represents the degree of power confined to the core with respect to the total propagating power. This ratio is plotted in Fig. 5, along with a_{opt}, as a function of α. We see that a_{opt} increases with decreasing α and that Φ reaches its peak very near $\alpha = 2$. A slightly larger value for Φ occurs if the profile is Gaussian; that is,

$$N = N_1 \exp\left[-\left\{\ln\left(\frac{N_1}{N_2}\right)\right\}\cdot\left(\frac{R}{a}\right)^2\right]. \tag{20}$$

This value of Φ is the open circle in Fig. 5. We suspect that this slightly larger value may be caused by the close matching of the field with the index profile. The Gaussian index profile and the $\alpha = 2$ profile yield ~40 percent increase in Φ over the step-index profile case. This may help in eliminating microbending loss in single-mode fibers without increasing Δ.

VI. CORE-TO-CLAD RATIO

In the design and fabrication of single-mode lightguides, it is customary to fix the core-to-clad ratio at 0.1. This value seems appropriate for step-index fibers. It is, therefore, quite important to investigate the behavior of the evanescent field for graded-index fibers. From eqs. 6 and 7, we can readily calculate the field intensity in the cladding for different values of α and any radius.

For the cases of $\alpha = 100$, 2, and 1, including a Gaussian index profile,

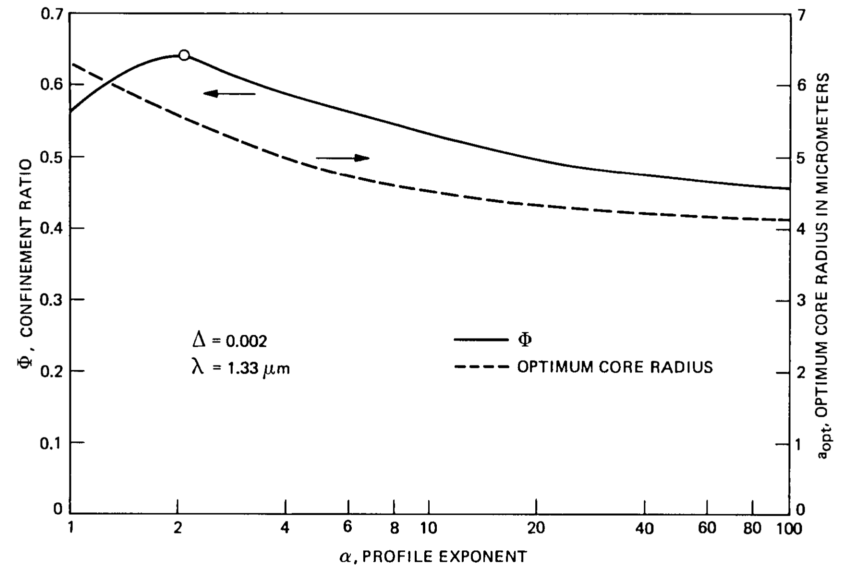


Fig. 5—Degree of field confinement in a single-mode fiber versus profile exponent α, (solid line). The dotted line shows the optimum core radius corresponding to the α value. The open circle indicates the maximum value of Φ obtained from a Gaussian-like index profile.

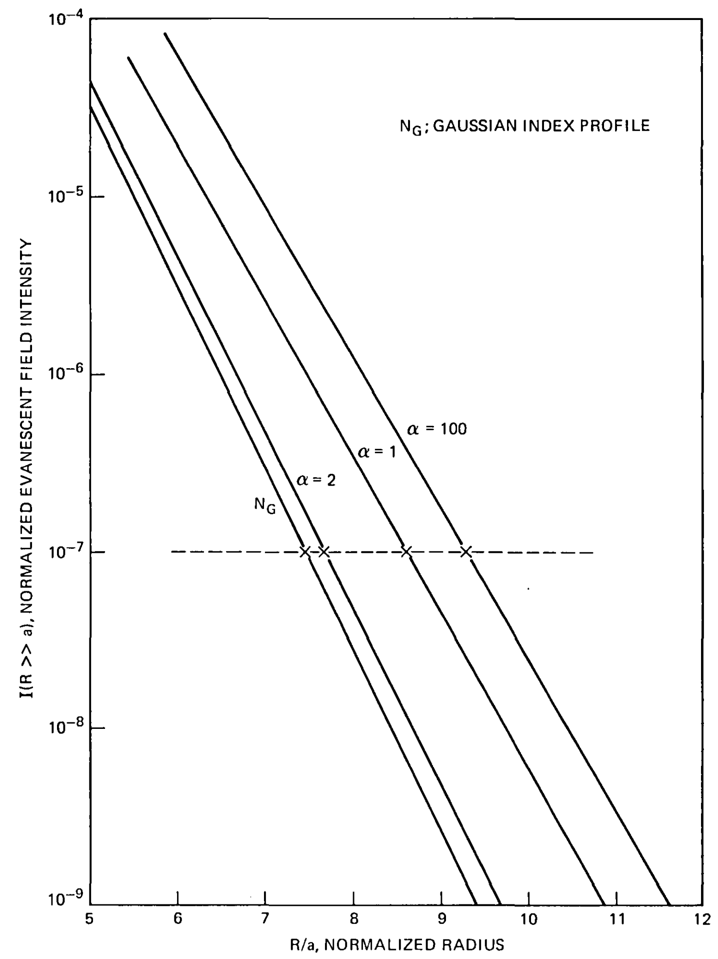


Fig. 6—Normalized evanescent field intensity at $R \gg a$ versus normalized radius for three different values of α and a Gaussian index profile. The horizontal dotted line is the cutoff level at $I = 10^{-7}$.

the results are given in Fig. 6. To compare the core-to-clad ratios for different α's we define an intensity level I at $R \gg a$ equal to 10^{-7} as the cutoff point. This corresponds to $R/a \sim 9.3$ for $\alpha = 100$. Accordingly, Fig. 6 shows that there are substantial differences in those values among the four cases. The clad-to-core ratio is reduced to 7.6 from 9.3 as α decreases to 2. The value for the Gaussian profile is very close to that for $\alpha = 2$. Finally, it is interesting to note that the value of I at the core-cladding interface is ~0.13 for $\alpha = 1$ and 2, but it is ~0.28 for $\alpha = 100$ (see Fig. 4).

VII. VALIDITY OF GAUSSIAN APPROXIMATION FOR THE FIELD FUNCTIONS OF HE_{11} MODE

As mentioned earlier, it is usually not possible to find an analytical expression for the electromagnetic field functions for the HE_{11} mode in a single-mode fiber. One exception to this is the step-index profile. Therefore, an approximate expression for the field distribution of the fundamental mode is frequently used to determine the propagation characteristics. A prevailing approximation is a Gaussian-like field.[11,12,13] Thus, we can write

$$E_g = E_0 \exp\left[-b \left(\frac{R}{a}\right)^m \right], \tag{21}$$

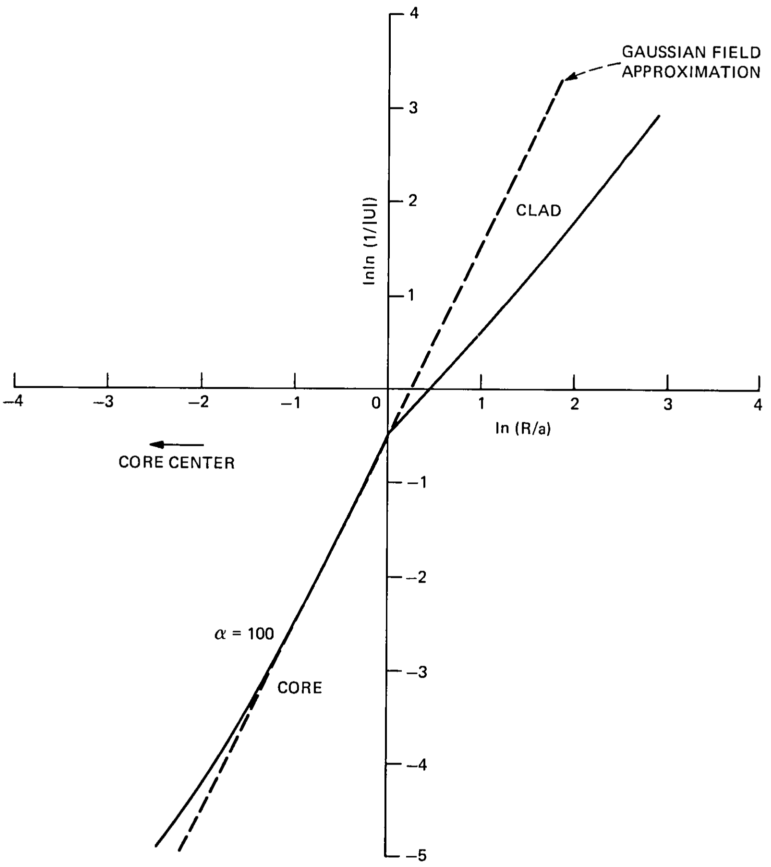


Fig. 7a—Comparison of Gaussian field distribution with exact field solutions. The solid lines are the exact values, and the dotted lines are the Gaussian approximation for $m = 2$ and $\alpha = 100$.

where E_0 and b are constants. Taking a double logarithm of both sides of eq. 21, we can rewrite it as

$$\ln \ln\left(\frac{1}{\bar{E}_g}\right) = \ln b + m \ln\left(\frac{R}{a}\right), \tag{22}$$

where

$$\bar{E}_g = E_g/E_0.$$

We introduce the dimensionless quantity U to represent any one of transverse electric or magnetic field components, for example, $\bar{E}_R$.

Equation (22) was plotted with $m = 2$ (precisely Gaussian) and compared with the exact values. The results are shown in Figs. 7a, b, and c for cases $\alpha = 100$, 2, and 1. The solid lines are the exact values, while the dotted lines are from the Gaussian approximation. In all

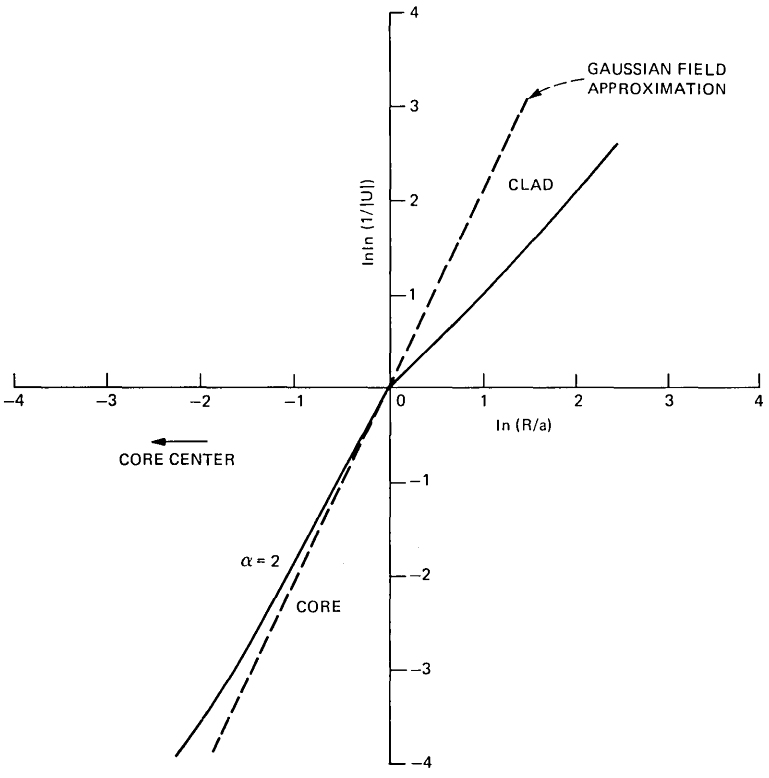


Fig. 7b—Comparison of Gaussian field distribution with exact field solutions. The solid lines are the exact values and the dotted lines are the Gaussian approximation for $m = 2$ and $\alpha = 2$.

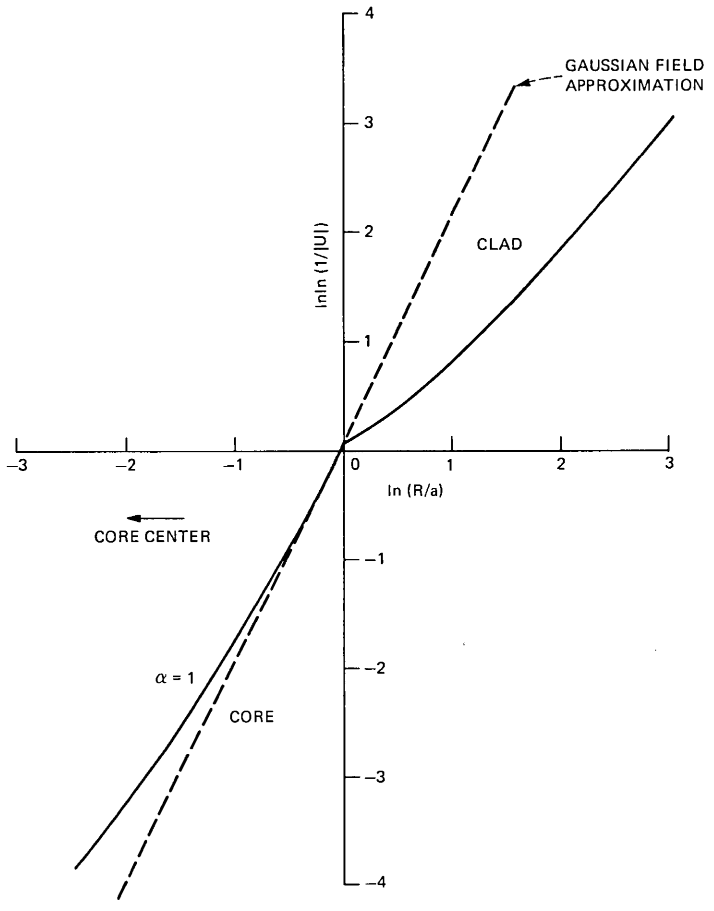


Fig. 7c—Comparison of Gaussian field distribution with exact field solutions. The solid lines are the exact values, and the dotted lines are the Gaussian approximation for $m = 2$ and $\alpha = 1$.

cases, the fields of the core region appear in the third quadrant of the figures. Those in the cladding region are shown in the first quadrant.

For the core region, when $\alpha = 100$ and it is near the core-cladding interface, there is satisfactory agreement between the Gaussian field function and the exact field function. There is poor agreement near the center of the core; this is also evident from Refs. 11 and 12. When $\alpha = 2$ or 1 there is much worse agreement between the exact and Gaussian functions in the core regions. For all values of α, the agreement in the cladding region is extremely poor. It is interesting to note that the slope of the exact field functions in the cladding for all α values is close to 1. This indicates that the field is decaying exponentially.

VIII. SUMMARY AND CONCLUSIONS

From a numerical solution to Maxwell's equations, we can accurately describe the field distribution of the HE_{11} mode in a single-mode lightguide. According to our calculated results, the fraction of power in the core reaches its maximum value near $\alpha = 2$. Thus, for a parabolic profile or Gaussian profile, the fraction of power within the core is ~40 percent larger than that of a step-index core. On the other hand, it is clear that the optimum core size increases with decreasing α value. A linear index profile ($\alpha = 1$) provides an optimum core size that is over 50 percent larger than that of a step-index core. Therefore, in designing a single-mode fiber, it is important to remember that one should choose a value of α to optimize certain characteristics, such as zero total dispersion, TE and TM cutoff, core size, manufacturing tolerances, field confinement, or microbending loss.

The work of Marcuse shows that for the case of step-index, single-mode fibers, microbending losses can either increase or decrease as a function of fiber radius, depending upon the statistics of the axis deformation function.[14,15,16] He also concludes that single-mode fibers with parabolic-index profiles may have smaller microbending losses than single-mode, step-index fibers. If the distortion power spectrum peaks sharply at low spatial frequencies, the advantage will be slight, if it exists at all. However, for distortions with a wider Fourier spectrum, the parabolic-index fiber should clearly be advantageous. The reason for this is that in the case of the distorted parabolic-index fiber, the sources of the radiation field are distributed throughout its volume, while in the case of the step-index fiber, they are located at the waveguide boundary. The constructive interference among the volume sources is never as pronounced as among the boundary sources.

It should be noted that when we increase the mode confinement we reduce the field at the core-cladding interface. This reduces the strength of the radiation sources because of microbending at this boundary. Furthermore, if there is a barrier layer such as B_2O_3, then bulk loss is reduced as well.

An additional discussion seems in order concerning the important TE and TM cutoff. As we have previously shown[1] for $\Delta = .002$ and $\lambda = 1.33\ \mu m$, the TE and TM cutoff is $1.0\ \mu m$ when $\alpha = 100$, and it shifts to $0.85\ \mu m$ for $\alpha = 1$. Attempts to increase the field confinement by increasing Δ, while keeping the core radius fixed, may move the cutoff rather close to the operating wavelength. Instead of increasing Δ it may sometimes be better to reduce α. The substantial increase in core size when $\alpha = 2$ is an advantage as far as coupling to a source is concerned. It has the important added advantage in that the clad-to-core ratio is reduced.

Finally, we must conclude that the Gaussian field approximation is likely to be of value only in the core region of a step-index fiber. It is always very poor in the cladding. However, the Gaussian-like approximation with $m \neq 2$ may be useful for graded-index, single-mode lightguides. For example, for $\alpha = 1$ and 2 the m values for best fit are ~1.6 and ~1.8, respectively.

IX. ACKNOWLEDGMENTS

The encouragement given us by R. J. Klaiber, L. S. Watkins, and M. I. Cohen is greatly appreciated.

REFERENCES

1. U. C. Paek, G. E. Peterson, and A. Carnevale, "Dispersionless Single-mode Lightguides with α Index Profiles," B.S.T.J. *60,* No. 5 (May–June 1981), pp. 583–98.
2. L. G. Cohen, C. Lin, and W. G. French, "Tailoring Zero Chromatic Dispersion into the 1.5–1.6 μm Low-Loss Spectral Region of Single-Mode Fibers," Electron. Lett., *15* (June 1979), pp. 334–5.
3. W. A. Gambling, H. Matsumura, and C. M. Ragdale, "Zero Total Dispersion in Graded-Index Single Mode Fibers," Electron. Lett., *15* (July 1979), pp. 474–6.
4. J. Sakai and T. Kimura, "Practical Microbending Loss Formula for Single-Mode Optical Fibers," IEEE J. Quantum Electron., *QE-15* (June 1979), pp. 497–500.
5. J. Sakai, "Simplified Bending Loss Formula for Single-Mode Optical Fiber," Appl. Opt., *18* (April 1979), pp. 951–2.
6. T. Li, "Structures, Parameters and Transmission Properties of Optical Fibers," Proc. IEEE, *68* (October 1980), pp. 1175–80.
7. G. E. Peterson et al., "An Exact Numerical Solution to Maxwell's Equations for Lightguides," BSTJ, *59* (September 1980), pp. 1175–96.
8. M. O. Vassel, "Calculation of Propagating Modes in a Graded-Index Optical Fiber," Opto-Electronics, *5* (July 1974), pp. 271–386.
9. J. W. Fleming, "Material Dispersion in Lightguide Glasses," Electron. Lett., *14* (May 1978), pp. 326–8.
10. W. A. Gambling and H. Matsumura, "Propagation in Radially-Inhomogenous Single Mode Fiber," Opt. Quantum Electron., *10* (January 1978), pp. 31–40.
11. D. Marcuse, "Gaussian Approximation of the Fundamental Modes of Graded-Index Fibers," J. Opt. Soc. Am., *68* (January 1978), pp. 103–9.
12. A. W. Snyder and R. A. Sammut, "Fundamental HE_{11} Modes of Graded Optical Fibers," J. Opt. Soc. Am., *69* (December 1979), pp. 1663–71.
13. H. Matsumura and T. Suganuma, "Normalization of Single-Mode Fibers Having an Arbituary Profile," Appl. Opt., *19* (September 1980), pp. 3151–8.
14. D. Marcuse, "Microbending Losses of Single-Mode, Step-Index and Multimode Parabolic-Index Fibers," B.S.T.J. *55,* No. 7 (September 1976), pp. 937–55.
15. D. Marcuse, "Theory of Dielectric Optical Waveguides," New York: Academic Press, 1974.
16. D. Marcuse, "Radiation Losses of Parabolic-Index Slabs and Fibers with Bent Axes," Appl. Opt., *17* (March 1978), pp. 755–62.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Priority Queuing Networks

By R. J. T. MORRIS

(Manuscript received October 17, 1980)

Priority service disciplines are widely used in computer and communications systems. Many such systems can be modeled by queuing networks, but presently developed theory does not allow solution of these models when priority service disciplines are present. For priority queuing networks that have a homogeneity property, we give some explicit results for mean delay and throughput. However, the assumption of homogeneity is too restrictive for many applications. We identify some examples of systems for which inhomogeneous two-node priority queuing networks are appropriate models and yield to exact analysis. The results allow some conclusions to be drawn about using priorities in a two-node closed network to establish grades of service. We also use the results to evaluate a commonly used approximation technique for priority queuing systems.

I. INTRODUCTION AND SUMMARY

Priority service disciplines are widely used in computer and communication systems. One common application of priorities is in the establishment of multiple grades of service whereby deferrable or background work is scheduled according to a lower priority. In other applications, a device may give prioritized service to a class of jobs known to be short so as to increase overall system throughput. For purposes of performance analysis, computer and communication systems have often been modeled as queuing networks. However, the theory of queuing networks in its present form (see Refs. 1, 2) does not provide solutions for even simple networks with priority disciplines, except in an approximate manner.[3-5]

There are very few exact results for queuing networks (i.e., queuing models with more than one service station or node) with priority to be found in the literature. One known result concerns a general service time, single-server queue with preemptive or nonpreemptive priority and finite exponential source.[6] This model can be thought of as a two-node closed queuing network where the second node is a pure delay or infinite-server group. In a paper by Avi-Itzhak and Heyman the mean cycle times were obtained for a central server model with priorities, under the assumption that the mean service times and routing patterns are the same for each priority class.[7] In other computer and communication applications, network priorities have only been represented approximately, using heuristics for the central server model[3,4] and a packet switching network.[5] While these approximation techniques may be adequate in accuracy for the parameter ranges of some applications, much further work is needed in improving and validating these techniques and, ultimately, in developing exact analytical results wherever they are tractable.

Our goals are to obtain insight into the solution form for some simple cases of queuing networks with priorities, to obtain an initial evaluation of the accuracy of existing approximation techniques, and to draw some conclusions on the performance of some simple network priority structures occurring in practice. In Section II, we describe a general class of priority queuing networks that are homogeneous in the sense that all customer classes are treated identically with respect to service time and routing. For homogeneous networks, we are able to give a mean delay and throughput analysis. However, it will be seen that the homogeneity assumption is sufficiently restrictive as to prevent application of these results in many situations. Subsequently, we focus on two specific examples of systems that can be modeled by simple queuing networks, but in which priority disciplines and inhomogeneity play crucial roles. These examples suggest several different two-node priority queuing network models that yield to exact analysis.

The first example we consider is a computer system consisting of a central processing unit (CPU) and an input/output (I/O) device, which processes both time-critical transactions, as well as nontime-critical batch jobs. The system is designed to give priority to the transactions at both the CPU and I/O device (in contrast to Refs. 3 and 4 where it is assumed that only the CPU observes priorities). This suggests use of the two-node, closed queuing network model A of Fig. 1a, with one node representing the CPU, and the other, the I/O device. The model has separate queues for each priority class at each node, and priority is observed preemptively at both nodes.

The second example is a full-duplex data link which is used for transmission of messages under a window flow control protocol. There are two grades of messages, the premium grade and the standard grade. When both premium grade messages and acknowledgments receive preemptive priority, model A is applicable (refer to Fig. 5 which is explained further in Section VI). However, since acknowledgments are typically shorter than messages, another configuration is

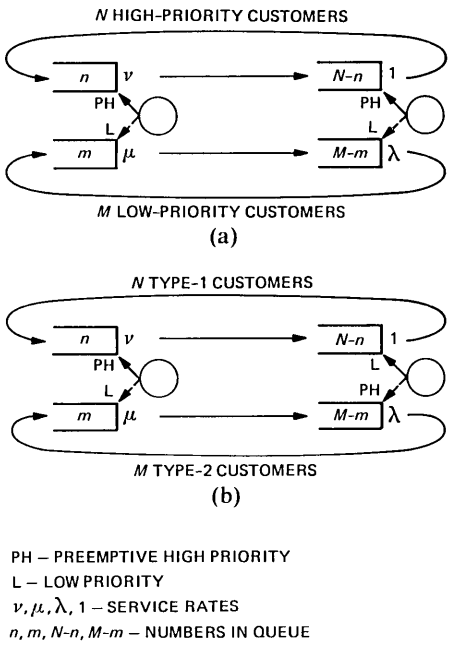


Fig. 1—Schematics of models A and B.

suggested wherein acknowledgments of either grade are given preemptive priority; this leads to model B shown in Fig. 1b.

In both models A and B there are a fixed number of customers in each class and service time distributions at a node are assumed to be exponential, but are not required to be the same at a node for each customer class (in contrast with homogeneous networks or the first-come first-served nodes described in Ref. 1). Because of the exponential assumption, the priorities can be understood to be either preemptive-resume or preemptive-restart (with resampling). For each model, service within a customer class at a node can be thought of as first-come first-served, but all equations and results remain valid for any other discipline within the priority class which does not take into account the actual service time requirement when selecting a customer for service.

The general approach we use in the analysis of models A and B, and several similar models, is to set up the balance equations (steady-state Kolmogorov forward equations) for the Markov chain describing the number of customers of each priority class at each node. These partial difference equations generally do not satisfy the well-known local balance condition[1,2] but, nevertheless, can still be solved to obtain the stationary distribution. This distribution allows throughput and mean delay to be computed for each customer class.

These results can be applied to obtain some general conclusions about the two systems we have used as motivating examples. In the computer system that processes both transactions and batch jobs, we find that if the transactions are bottlenecked at one device (CPU or I/O), the batch jobs need to be even more strongly bottlenecked at the other device, if a significant batch throughput is to be attained. Specifically, we show that if the transactions have a bottleneck of strength x at one node, the batch jobs need to have a bottleneck of strength x^N at the other node (where N is the transaction multiprogramming level), if the batch jobs are to be able to fulfill a role as "filler" work. In the data link example, we find a similar result: if standard grade message traffic is carried in purely a background mode, fairly extreme parameters are necessary before its introduction becomes attractive. On the other hand, if some compromise of the premium traffic performance is permitted, then an appreciable amount of standard grade message traffic can be carried by using data link capacity that would otherwise be wasted. For each system, we identify hazards that can occur when the lower-priority work is allowed to interfere with higher-priority work. Refer to Sections V and VI for further details.

In Section VII, we use the results to evaluate the effectiveness of a well-known approximation technique. We find that accuracy of the approximation technique varies from good to poor, depending on the parameters of the application. A criterion on the application parameters is proposed under which the approximation technique would be expected to perform well.

II. HOMOGENEOUS NETWORKS

In this section, we consider a class of queuing networks that allow preemptive priorities but are otherwise homogeneous in the sense that at any one node all customers are treated identically with respect to service rate and routing. The results rely on an observation similar to that made by Avi-Itzhak and Heyman in their analysis of the central server model.[7]

We first consider a closed queuing network of the Gordon-Newell type.[8] It consists of N service centers or nodes numbered $1, \cdots, N$. In departure from the Gordon-Newell formulation, there are P priority classes numbered $1, \cdots, P$, with the ith (priority) class containing K_i customers, $i = 1, \cdots, P$ and

$$\sum_{i=1}^{P} K_i = K.$$

At any node, a customer from a higher-numbered class takes preemptive priority over a customer from a lower-numbered class. The service time distribution at node j is exponential with rate μ_j for all customers, and the service discipline is first-come first-served within each priority class. After service at a node is completed, routing to another node is governed by a probability vector which is the same for each priority class. Let the state of the network be expressed by the quantities n_j^i, $j = 1, \cdots, N$, $i = 1, \cdots, P$, where n_j^i denotes the number of class i customers present at node j. Define the aggregate state variable

$$m_j^i = \sum_{k=i}^{P} n_j^i,$$

the number of priority class i or higher customers at node j. The key observation is that the random variable m_j^i is equivalent to that which would result if the network were modified by first removing customers of priority less than i (i.e., by setting $K_k = 0$, $1 \leq k < i$) and, thereafter, ignoring all priority service distinctions. This is because (*i*) lower-priority customers exert no influence on higher-priority customers, and (*ii*) regardless of whether priorities are observed between customers of classes $i, i+1, \cdots, P$, transitions in the total number (m_j^i) of customers of priority i, or larger, at a node are not altered (by the assumed uniformity of service rate and routing over priority classes). Thus, we can find the stationary distribution of m_j^i by the usual closed queuing network techniques.[1,2,8,9] The stationary distribution of the aggregate variable m_j^i is sufficient to determine the steady-state mean delay and throughput of each priority class at each node. This follows from the fact that for each i and j

$$E[n_j^i] = E[m_j^i] - E[m_j^{i+1}],$$

$$\Pr[n_j^i > 0,\ m_j^{i+1} = 0] = \Pr[m_j^i > 0] - \Pr[m_j^{i+1} > 0],$$

where m_j^{P+1} is understood to be identically zero. Hence, priority class i customers have a throughput at node j of

$$T_j^i = \mu_j[\Pr(m_j^i > 0) - \Pr(m_j^{i+1} > 0)]$$

and a mean delay (including service time) of

$$D_j^i = [E(m_j^i) - E(m_j^{i+1})]/T_j^i$$

by Little's Law. Note that these quantities are obtained in the process of carrying out the mean value analysis for a single-chain closed network with K customers.[9]

Similar results are obtained for an open network of the Jackson type.[10] All notation is the same as for the closed network, except that we no longer specify the number of customers of each priority class but, instead, we specify the rate λ_j^i of exogenous Poisson arrivals of priority class i to node j. We must now assume that the traffic equations admit a unique solution e_j^i representing the mean arrival rate of customers of priority class i to node j and that

$$\sum_{i=1}^{P} e_j^i < \mu_j$$

for each j. We make the analogous observation that the quantity m_j^i can be obtained by considering the network modified by turning off all arrival streams of priority less than i (i.e., setting $\lambda_j^k = 0$, $1 \le k < i$, $1 \le j \le N$) and, thereafter, ignoring priority distinctions at service. We then have

$$E[n_j^i] = E[m_j^i] - E[m_j^{i+1}],$$

$$T_j^i = e_j^i,$$

and

$$D_j^i = [E(m_j^i) - E(m_j^{i+1})]/e_j^i.$$

Now,

$$E[m_j^i] = \sum_{k=i}^{P} e_j^k \bigg/ \left(\mu_j - \sum_{k=i}^{P} e_j^k\right)$$

and, therefore,

$$D_j^i = \frac{\mu_j^{-1}}{\left(1 - \sum_{k=i}^{P} \rho_j^k\right)\left(1 - \sum_{k=i+1}^{P} \rho_j^k\right)},$$

where $\rho_j^k = e_j^k/\mu_j$ is the utilization of node j due to class k customers. We, thus, recognize the validity in a network context of the Cobham-type formula originally obtained by White and Christie[11] for the delay in an isolated preemptive priority $M/M/1$ queue when the service times are the same for each priority class.

Within the stringent limitations imposed by our homogeneity assumptions, some extensions to these results are possible. For example, we can allow the more general service disciplines shown by Kelly (see Ref. 2, pages 58 and 78) to lead to product form provided (*i*) the state dependence and server sharing embodied in these disciplines extend only to the customers of highest priority present at a node and ignore all lower-priority customers, and (*ii*) all customers are treated identically with respect to service time and routing.

The above results might possibly be useful in some queuing network applications. For example, a first-cut evaluation of the impact of introducing data packet priorities into a packet switching network could be carried out by representing exogenous packet arrivals as Poisson and data links as exponential servers,[12,13] and by assuming that the mean data packet length and the traffic routing pattern are the same for each priority class. If short control (e.g., acknowledgment) packets were to be given priority over data packets, we find that our homogeneity assumptions would be violated, although the effect of the short packets could be approximated in several ways.[5,13] Indeed, the case of control packets receiving priority serves to illustrate a common situation in which customers receiving priority have significantly smaller service time requirements. Thus, the results described in this section are expected to find limited use. The remainder of this paper does not make any such homogeneity assumption; unfortunately, by relaxing this assumption, we are able to treat only networks consisting of two nodes.

III. MODEL A: TWO NODES WITH PRIORITIES THE SAME AT EACH NODE

We now consider the two-node closed queuing network introduced in Section I as model A and shown in Fig. 1a. The network consists of two nodes—the left- and right-hand nodes. There are N high-priority and M low-priority customers. High-priority customers take preemptive priority over low-priority customers at each node. All service times are assumed exponentially distributed: the high-priority customers have a mean service time of ν^{-1} at the left node and 1 at the right; low-priority customers have a mean service time of μ^{-1} at the left node and λ^{-1} at the right. After service at one node is completed, customers are immediately routed to the other node without changing class. We assume ν, μ, λ, N, and M are all positive.

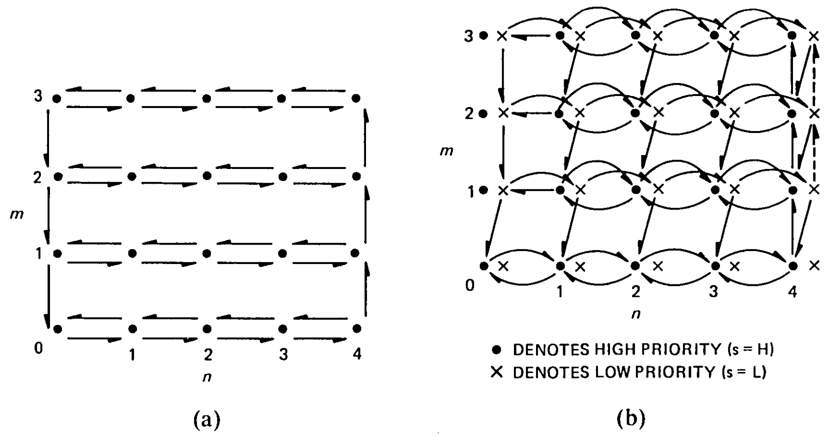
The state of the system is described by the vector (n, m) where n (respectively m) is the number of high- (respectively low) priority customers at the left node. The state (n, m) evolves as a Markov chain with stationary distribution $p(n, m)$. It is obvious that the stationary distribution is also the limiting distribution since the chain is finite and irreducible. The transitions of (n, m) are shown in Fig. 2a.

By definition, $p(n, m)$ satisfies the balance equations

$$\begin{aligned} p(n, m)[\nu 1_{\{n>0\}} + \mu 1_{\{n=0,m>0\}} + 1_{\{n<N\}} + \lambda 1_{\{n=N,m<M\}}] \\ = p(n-1, m) + p(n+1, m)\nu + p(N, m-1)\lambda 1_{\{n=N\}} \\ + p(0, m+1)\mu 1_{\{n=0\}}, \qquad 0 \le n \le N, \qquad 0 \le m \le M, \quad (1) \end{aligned}$$

where $1_{\{\ \}}$ denotes the indicator function which has value 1 (respectively 0) when the predicate within the braces is true (respectively false). Note that we are adopting the convention that $p(n, m) = 0$ when $(n, m) \notin [0, N] \times [0, M]$.

We wish to solve for $p(n, m)$. The technique we use is best explained by reference to the state transition diagram shown in Fig. 2a. First



m
n
0
1
2
3
4
(a)
(b)
● DENOTES HIGH PRIORITY (s = H)
X DENOTES LOW PRIORITY (s = L)

Fig. 2—(a) State transition diagram for model A shown for N = 4, $M = 3$. (b) State transition diagram for model A, but with left node nonpreemptive, shown for $N = 4$, $M = 3$.

note that for any $n > 0$, $p(n, m)$, $0 \le m \le M$ is expressible in terms of $p(n-1, m)$, $0 \le m \le M$. Hence, $p(n, m)$, $0 \le m \le M$, $0 < n \le N$ can be expressed in terms of the left boundary values $p(0, m)$, $0 \le m \le M$, and solution for $p(0, m)$, $0 \le m \le M$ rests on the balance equations for the right boundary $p(N, m)$, $0 \le m \le M$. This observation is generally true for an arbitrary two-dimensional birth-death process (provided all right-to-left horizontal transitions are present) but not always useful since the resultant equations for $p(0, m)$, $0 \le m \le M$ are not easily solved. Fortunately, in our case, the absence of vertical transitions from states (n, m), $0 < n < N$ caused by the priority structure results in relations for $p(0, m)$ which comprise a simple difference equation of order two which is easily solved and yields an explicit solution for $p(n, m)$. Before proceeding with this technique, we mention that a computational method based on such an observation has been proposed in which the recursive structure is used to reduce the problem of finding the stationary distribution of certain $N \times M$ birth-death processes to the solution of N equations in N unknowns.[14]

For notational ease, define $\alpha(m) = p(0, m)$, $0 \le m \le M$. Writing eq. (1) for $n = 0$ yields

$$p(1, 0) = \alpha(0)\nu^{-1} - \alpha(1)\mu\nu^{-1}, \tag{2}$$

$$p(1, m) = \alpha(m)(\mu + 1)\nu^{-1} - \alpha(m + 1)\mu\nu^{-1}, \qquad 0 < m < M \tag{3}$$

$$p(1, M) = \alpha(M)(\mu + 1)\nu^{-1}, \tag{4}$$

and for $0 < n < N$

$$p(n, m)(\nu + 1) = p(n + 1, m)\nu + p(n - 1, m), \qquad 0 \le m \le M \tag{5}$$

for which the general solution is

$$p(n, m) = (\alpha(m)(\nu^{-n} - \nu^{-1}) + p(1, m)(1 - \nu^{-n}))/(1 - \nu^{-1}),$$
$$0 \le n \le N, \qquad 0 \le m \le M, \qquad (6)$$

provided $\nu \neq 1$. Hence, the problem is reduced to determination of $\alpha(m)$, $0 \le m \le M$. This is done by writing eq. (1) for $n = N$,

$$p(N, 0)(\nu + \lambda) = p(N - 1, 0), \qquad (7)$$

$$p(N, m)(\nu + \lambda) = p(N - 1, m) + p(N, m - 1)\lambda, \qquad 0 < m < M, \qquad (8)$$

$$p(N, M)\nu = p(N - 1, M) + p(N, M - 1)\lambda. \qquad (9)$$

Substituting eqs. (2) and (6) into eq. (7),

$$\alpha(1)/\alpha(0) = (\lambda/\mu)[\nu^{-N}(\nu - 1)]/(\nu + \lambda - 1 - \lambda\nu^{-N}). \qquad (10)$$

Assuming $M > 1$, taking $m = 1$ in eq. (8), and using eqs. (2), (3), and (6) yields $\alpha(2)/\alpha(1) = r$, where

$$r = \frac{\lambda}{\mu} \frac{\mu - (\mu - \nu + 1)\nu^{-N}}{\nu + \lambda - 1 - \lambda\nu^{-N}}. \qquad (11)$$

Note that the denominator of r is not zero because $\nu \neq 1$. Taking $1 < m < M$ in eq. (8) with eqs. (3) and (6) yields the difference equation

$$\alpha(m + 1)[\mu\lambda\nu^{-N} + \mu - \mu\lambda - \mu\nu]$$
$$+ \alpha(m)[\mu\nu + \lambda\nu^{1-N} + 2\lambda\mu - 2\lambda\mu\nu^{-N} - \lambda\nu^{-N} - \mu]$$
$$+ \alpha(m - 1)[\lambda\mu\nu^{-N} + \lambda\nu^{-N} - \lambda\mu - \lambda\nu^{1-N}] = 0, \qquad 1 < m < M,$$

which has characteristic roots 1, r. But since $\alpha(2)/\alpha(1) = r$, we have

$$\alpha(m) = \alpha(1)r^{m-1}, \qquad 1 \le m \le M. \qquad (12)$$

It can now be verified that these results are consistent with the one unused eq. (9) and that the result holds true for $M = 1$.

Substituting eqs. (2) to (4), (10), and (12) into eq. (6) and simplifying yields the general solution for $\nu \neq 1$

$$p(n, m) = Cr^{m}[r\nu^{-n\delta(m-M)} - \nu^{(N-n)\delta(m)} + ((1 - \nu)\mu^{-1}$$
$$+ 1 - r)\nu^{-n}], \qquad 0 \le n \le N, \qquad 0 \le m \le M, \qquad (13)$$

where r is given by eq. (11) and $\delta(\cdot)$ is the Kronecker delta: $\delta(0) = 1$; $\delta(k) = 0$, $k \neq 0$. The normalizing constant C is obtained by demanding that $p(n, m)$, $0 \le n \le N$, $0 < m \le M$ is a probability distribution, yielding

$$C = \frac{\mu(1 - \nu^{-1})}{(1 - \nu^{-N-1})\left[(1 - \nu)\sum_{i=0}^{M} r^{i} + \mu(1 - \nu^{N})\right]} \qquad (14)$$

In eq. (14) and below we refrain from summing geometric series of the form

$$\sum_{i=0}^{I} x^i$$

to avoid a special statement for the case $x = 1$.

Our treatment has excluded the case $\nu = 1$ for which the solution to eq. (5) is $p(n, m) = \alpha(m)(1 - n) + p(1, m)n$. Rather than resolving for this case, it is easier to take the limit as $\nu \rightarrow 1$ in eqs. (13) and (14); this must yield the correct solution since the coefficients in eq. (1) are continuous in ν and the eigenvectors of a matrix are continuous functions of its elements. Equations (13) and (14) for $p(n, m)$ can be rewritten in a form which is well-defined for $\nu = 1$:

$$p(n, m) = Ds^m \Bigg[\mu^{-1}\nu^{-n} + (1 - s)\nu^{-1} \sum_{i=0}^{n-1} \nu^{-i} + 1_{\{m=0\}} \sum_{i=0}^{N-n-1} \nu^i + 1_{\{m=M\}} s\nu^{-1} \sum_{i=0}^{n-1} \nu^{-i} \Bigg], \tag{15}$$

where

$$s = \left(\sum_{i=0}^{N-1} \nu^{-i} + \mu^{-1}\nu^{1-N} \right) \Bigg/ \left(\sum_{i=0}^{N-1} \nu^{-i} + \lambda^{-1}\nu \right), \tag{16}$$

$$D^{-1} = \left(\sum_{i=0}^{N-1} \nu^i + \mu^{-1} \sum_{i=0}^{M} s^i \right) \sum_{i=0}^{N} \nu^{-i}, \tag{17}$$

and summations over descending ranges are taken to be zero. Since the solution given by eqs. (15) to (17) is continuous in $\nu > 0$, it is the general solution for all $\nu > 0$.

Remarks

(*i*) The system considered here can be generalized trivially to allow mean service time of the high-priority customers at the right node to be κ^{-1} (rather than unity) and to allow routing of a customer back to the node at which it has just completed. Let high-priority customers on completion at the left (respectively right) node be routed to the right (respectively left) node with probability p_{lr} (respectively p_{rl}) and be routed to the node at which completion has just occurred with complementary probability $1 - p_{lr}$ (respectively $1 - p_{rl}$). Let the low-priority customers have similarly defined probabilities q_{lr}, q_{rl}. Then the results in eqs. (13) to (17) remain valid with ν, μ, λ replaced, respectively, by $\nu p_{lr}/\kappa p_{lr}$, $\mu q_{lr}/\kappa p_{rl}$, $\lambda q_{rl}/\kappa p_{rl}$.

(*ii*) It is readily verified that the marginal distribution of the high-priority customers $p(n, \cdot) = \sum_{m=0}^{M} p(n, m)$ agrees with the result for an ordinary two-node closed network, viz.

$$p(n, \cdot) = \nu^{-n} \Big/ \sum_{i=0}^{N} \nu^{-i}.$$

Of course, this must be the case, since the high-priority customers experience no interference from the low-priority customers.

(*iii*) Let T_H and T_L denote the throughput of high- and low-priority customers, respectively. Then

$$T_H = \nu(1 - p(0, \cdot)) = \nu\left[1 - \left(\sum_{i=0}^{N} \nu^{-i}\right)^{-1}\right] \quad \text{and}$$

$$\begin{aligned} T_L &= \mu[p(0, \cdot) - p(0, 0)] \\ &= \sum_{i=1}^{M} s^i \Big/ \left(\sum_{i=0}^{N-1} \nu^i + \mu^{-1} \sum_{i=0}^{M} s^i\right) \sum_{i=0}^{N} \nu^{-i}. \end{aligned}$$

If the generalization referred to in (*i*) is adopted, then as well as the replacements specified in (*i*), the quantities T_H, T_L must be multiplied by κp_{rl} if they are to have the units of customers per unit time.

Mean delay formulas for high- and low-priority customers at each node are immediately obtained from $p(n, m)$ by Little's law, but are omitted here.

(*iv*) In the special case that $\mu = \nu$ and $\lambda = 1$, i.e., service times do not depend on the priority class, the throughputs can be obtained from the considerations of Section II. In that case, the distribution of $n + m$ will be the same as a two-node closed network with $N + M$ customers and no priorities. Hence, we can immediately write down

$$\begin{aligned} T_H &= \nu\left[1 - \left(\sum_{i=0}^{N} \nu^{-i}\right)^{-1}\right], \\ T_L &= \nu\left[1 - \left(\sum_{i=0}^{N+M} \nu^{-i}\right)^{-1}\right] - T_H \\ &= \nu\left[\left(\sum_{i=0}^{N} \nu^{-i}\right)^{-1} - \left(\sum_{i=0}^{N+M} \nu^{-i}\right)^{-1}\right], \end{aligned}$$

and this checks with the result in (*iii*).

Note also that in this case ($\mu = \nu$, $\lambda = 1$), then $s = \nu^{-1}$ and $p(n, m) = D\nu^{-1-m}$, $0 < m < M$. Thus, if we observe the system only at instants when there is at least one low-priority customer at each node, the distribution of high-priority customers is seen to be uniform.

(*v*) When $s \geq 1$, using the expressions in (*iii*)

$$\lim_{M \to \infty} \frac{T_H}{\nu} + \frac{T_L}{\mu} = 1,$$

i.e., the utilization of the left-hand node approaches unity as the number of low-priority customers increases. It follows from the definition of s that $s \geq 1$ if and only if $\lambda/\mu \geq \nu^N$. Using this fact and reversing the two nodes shows conversely that if $s \leq 1$, utilization of the right-hand node approaches unity as $M \rightarrow \infty$. Thus, whether $s > 1$ or $s < 1$ determines which node becomes the limiting factor in trying to obtain increased total throughput by introducing additional low-priority customers.

The following criterion can be deduced. Suppose a two-node system with N circulating customers has a bottleneck of strength ν, $\nu > 1$ at the right node. For moderate values of N, the right node will be almost completely utilized and the left node underutilized. If the low-priority customers have a bottleneck at the left node of strength at least ν^N, i.e., $\lambda/\mu \geq \nu^N$, then the left node can have a utilization as close as desired to unity by introduction of sufficiently many low-priority customers. If the low-priority customers have a bottleneck weaker than ν^N at the left node, then complete use of the left node can never be achieved by introducing low-priority customers. This rule of thumb can be deduced intuitively as follows. The high-priority customers can be thought of as causing a reduction of processing rate to $\mu[1 - T_H/\nu]$ at the left node and to $\lambda[1 - T_H]$ at the right node. Thus, the left node can be fully utilized if and only if $\mu(1 - T_H/\nu) \leq \lambda(1 - T_H)$ which reduces to the condition $\lambda/\mu \geq \nu^N$. Of course, the formulas for T_H and T_L make precise the actual throughputs achieved as a function of the parameters. This is illustrated in Section V by an example.

(*vi*) The same solution technique can be used to obtain results for more than two priority classes, although the solution complexity increases. We state the result for a three-class, two-node system with number in each class N, M, L (in order of decreasing priority), with service time at the left node μ^{-1} (for all classes) and 1 at the right, $\mu \neq 1$. If $p(n, m, l)$ describes the stationary probability of having n, m, l customers at the left (in order of decreasing priority), then, provided N, M, L, μ are positive,

$$p(n, m, l) = \begin{cases} b(\mu^{-n} - \mu^{-1-N}), & l = 0, \quad m = 0 \\ B\mu^{-l}(\mu^{-n} - \mu^{-1-N}), & 0 < l \leq L, \quad m = 0 \\ B\mu^{M-m} - B\mu^{-1-n}, & l = 0, \quad 0 < m < M \\ B(1 - \mu^{-1})\mu^{-l-n}, & 0 < l < L, \quad 0 < m < M \\ B\mu^{-L}(\mu^{-n} - \mu^{-N-m-1}), & l = L, \quad 0 < m < M \\ B\mu^{-l}(1 - \mu^{-n-1}), & 0 \leq l < L, \quad m = M \\ b\mu^{-M-N-L}(1 - \mu^{-n-1}), & l = L, \quad m = M \end{cases},$$

where

$$B = b(1 - \mu)/(1 - \mu^{M+N+1})$$

and

$$b = (1 - \mu^{-1})/[(1 - \mu^{-N-1})(1 - \mu^{-M-N-L-1})].$$

A result for three priority classes is sometimes useful in that it allows comparison of the performance of a designated customer class with two aggregated classes: one representing those customer classes of higher priority and the other those customer classes of lower priority.

(*vii*) A variation on model A which will be of interest in Section VI is the case where priority is again preemptive at the right node but nonpreemptive at the left node. We use the same notation as above, except that the state description now becomes (n, m, s), where $s = H$ (respectively L) when the left node is processing high- (respectively low) priority customers. The state transition diagram for this model is shown in Fig. 2b where transitions out of the transient states $\{(n, 0, L), 0 \le n \le N; (0, m, H), 0 < m \le M\}$ are omitted and we have adopted the convention that $s = H$ when $n = m = 0$. The stationary probabilities $p(n, m, s)$ can be solved for by a technique similar to that used above. Namely, letting $p(N, m, L) = \alpha(m)$, $1 \le m \le M$, $p(0, 0, H) = \alpha(0)$ and writing balance equations for all states with $s = L$, yields expressions for $p(n, m, L)$, $1 \le m \le M$, $0 \le n \le N$ and $p(1, m, H)$, $0 \le m \le M$ in terms of $\alpha(\cdot)$. The balance equations for states (n, m, H), $0 \le m \le M$, $1 \le n < N$ yield $p(n, m, H)$, $0 \le m \le M$, $1 \le n \le N$, again in terms of $\alpha(\cdot)$. The analysis is completed by solving the third-order constant coefficient difference equation for $\alpha(\cdot)$ that is obtained from the balance equations for states (N, m, H), $0 \le m \le M$. This approach leads to a solution which, although closed form, is of limited use because of its complexity.

Instead, we now briefly describe a much simpler approximate solution which appears justifiable for our applications. The system is approximated by omitting the transitions shown by dashed lines in Fig. 2b, i.e., $(N, m, L) \rightarrow (N, m + 1, L)$, $0 < m < M$. With these transitions omitted, the solution steps simplify and we obtain

$$p(n, m, L) = \beta(m)(\nu - 1)(\mu + 1)^{-n-1}(1 + \mu^{-1})^{\delta(n-N)}, \qquad 1 \le m \le M, \quad 0 \le n \le N,$$

$$p(n, m, H) = \beta(m)(1_{\{m>0\}} - \nu^{-n}) + \beta(m + 1) \cdot \{-1 + [\mu\nu^{-n} - (\mu + 1)^{-n}(\nu - 1)]/(\mu - \nu + 1)\}1_{\{m < M\}}, \qquad 0 \le m \le M, \quad 1 \le n \le N \quad \text{or} \quad m = n = 0,$$

where

$$\beta(m) = Ct^{m-1}(1 - \nu^{N})^{\delta(m)},$$

$$t = \frac{\lambda(1 - \nu^{-N})(\mu - \nu + 1)}{(\mu - \nu + 1)(\nu + \lambda - 1) + \lambda(\mu + 1)^{-N}(\nu - 1) - \lambda\mu\nu^{-N}},$$

C is a normalizing constant, $M > 1$, $N > 1$, $\nu \neq 1$, $\nu \neq \mu + 1$, and all unspecified probabilities are zero. As before, the cases $\nu = 1$, $\nu = 1 + \mu$ are obtained by taking limits.

The omission of the dashed transitions amounts to a denial of service to low-priority customers at the right node when $n = N$ and $s = L$. This is a justifiable approximation when the probability is small that N high-priority customers and one low-priority customer can be served at the right in less time than it takes to serve one low-priority customer at the left. This condition is stated as $(1 + \mu)^{-N}\lambda/(\lambda + \mu) \ll 1$, and this expression is shown to hold for our applications in Section VI. The approximation causes an underestimation of low-priority throughput and an overestimation of high-priority throughput. Although we omit details here, the opposite bounds (an upper bound for low-priority throughput and a lower bound for high-priority throughput) could also be obtained by replacing the transition $(N, m, L) \rightarrow (N, m + 1, L)$ by $(N, m, L) \rightarrow (N, M, L)$, $0 < m < M$, leading, in turn, to a system which is solved the same way.

IV. MODEL B: TWO NODES WITH PRIORITIES REVERSED

Model B, shown in Fig. 1b is now considered. This queuing network differs from model A only in that the priority at the right node has been reversed. We now refer to N type-1 customers and M type-2 customers (see Fig. 1b) with n(respectively m) being the number of type 1 (respectively type 2) customers at the left node. We assume N, M, ν, μ, λ are all positive.

The state transition diagram for this model is shown in Fig. 3. One immediately recognizes that states $\{(n, m)$: $n > 0$ and $m < M\}$ are transient, i.e., in equilibrium, one of the two high-priority queues is always empty. This is also easily deduced by considering system behavior at instants after the state $(0, M)$, i.e., both high-priority queues empty, is reached. One of the low-priority queues completes a

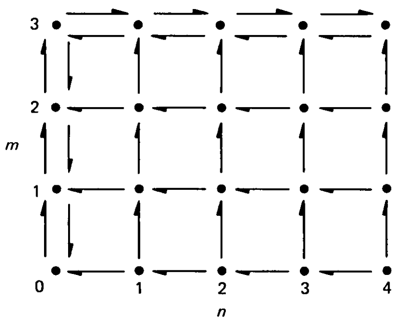


Fig. 3—State transition diagram for model B shown for $N = 4$, $M = 3$.

service and then that customer type prevents service of the other customer type until the state $(0, M)$ is again attained.

Let $p(n, m)$ be the stationary probability of state (n, m). Using the observation as to which states are persistent, we have immediately that

$$p(0, m) = C(\lambda/\mu)^m, \qquad 0 \le m \le M \qquad \text{and}$$
$$p(n, M) = C(\lambda/\mu)^M \nu^{-n}, \qquad 0 \le n \le N,$$

where

$$C^{-1} = \sum_{m=0}^{M=1} (\lambda/\mu)^m + (\lambda/\mu)^M \sum_{n=0}^{N} \nu^{-n}$$

and all other probabilities are zero. The values of $p(n, m)$ are also the limiting probabilities.

Remarks

(*i*) For the case $\nu = \mu$, $\lambda = 1$, this result can be obtained directly from standard closed queuing network results, together with the observation regarding persistent states.

(*ii*) For this model, there is no absolute priority given to either type of customer over the other—the service received by each type is determined by parameter values. We can write down the throughput of type 1 (respectively type 2) customers, denoted T_1 (respectively T_2):

$$T_1 = \nu \sum_{n=1}^{N} p(n, M) = \nu \Bigg/ \left[1 + \left(\sum_{n=1}^{N} \nu^{-n} \right)^{-1} \sum_{m=0}^{M} (\lambda/\mu)^{-m} \right],$$
$$T_2 = \mu \sum_{m=1}^{M} p(0, m) = \lambda \Bigg/ \left[1 + \left(\sum_{m=1}^{M} (\lambda/\mu)^{-m} \right)^{-1} \sum_{n=0}^{N} \nu^{-n} \right].$$

(*iii*) In view of the earlier observation regarding transient states, we point out one aspect of the behavior of this system. As already described, in equilibrium, the system will alternate between periods during which only customers of one type are processed. The distribution of the lengths of these periods can be obtained using a standard $M/M/1/K$ (K waiting positions) busy period argument. In the situation that $\nu \ll 1$ (respectively $\lambda \ll \mu$), while each customer type may perceive satisfactory (long-term) *average* throughput, the duration of the period during which only type 1 (respectively type 2) customers are served can be extremely long. The adverse impact of such behavior on the tails of delay distributions is obvious. This phenomenon should be taken into account when an application requires good short-term, as well as long-term performance.

(*iv*) An interesting result for this system is that the delay of one type of customer at its higher-priority queue is not influenced by the presence of customers of the other type. For example, the mean delay of type 1 customers at the left node is given by

$$\sum_{n=1}^{N} n\nu^{-n} \Big/ \sum_{n=0}^{N-1} \nu^{-n}$$

for any λ, μ, M, corresponding to the ordinary closed queuing network result where M would be zero. This invariance is explained as follows. If the random variable n is observed only at instants when it satisfies $n > 0$, then it is indistinguishable from its behavior in an ordinary queuing network where M would be zero. This is because, in equilibrium, $n > 0$ implies $m = M$, and so the type 1 customer sees no interference from type 2 customers. But type 1 customer delays at the left are only measured when $n > 0$ and therefore, the distribution of delay is unaffected by type 2 customers.

(v) A variation on model B where one node, say the left, has nonpreemptive priorities can be solved by the same technique. In this case, the only persistent states are $\{(n, m, L)\colon n = 0 \text{ or } m = M\}$ and $\{(n, m, H)\colon n > 0 \text{ and } m \geq M - 1\}$.

V. COMPUTER SYSTEM EXAMPLE

We now use the results we have developed to evaluate several performance issues in a computer system. We consider a simplified model of a computer system consisting of a CPU and I/O device. The system is primarily intended to process time-critical transactions which it does with a multiprogramming level of N. Each transaction makes I/O requests requiring a mean service time of 10 ms separated by CPU processing of mean duration 5 ms. After a certain number of loops between the CPU and I/O device, the transaction is completed and leaves the system. At this point, the transaction is considered to immediately re-enter the system in accordance with the assumption that there is always a backlog of transactions waiting outside the system.*

The transaction workload is clearly I/O-bound and we ask whether introduction of CPU-bound batch "filler" or background work at lower priority will result in a worthwhile improvement of CPU utilization and, consequently, total throughput. Suppose batch jobs are introduced with a permitted multiprogramming level of M and that they require γ seconds of processing on the average between visits to the I/O device where mean service time is 10 ms. We again assume that the batch multiprogramming level of M is maintained by a backlog of work.*

* Each transaction or batch job alternately visits the CPU and I/O device, beginning with the CPU, ending with the I/O device and looping between the two an arbitrarily distributed number of times. Variations on this "scenario" can be modeled using the technique in Remark (i), Section III.

We will initially assume that the transactions are given preemptive priority at both the CPU and I/O device. This arrangement would reflect an attitude that the performance of high-priority transaction work should not be compromised by introduction of background work. Hence, we use model A, and will be assuming that all service times are exponentially distributed. Because of this exponential assumption, the preemption can be either resume or restart (with resampling). Preemptive-resume is more appropriate at the CPU, and preemptive-restart (with resampling) is more appropriate at an I/O device such as a moving head disk where the data transfer time is typically small compared to seek and latency (justifying the restart assumption) and service time depends on the physical location of the last interrupting request's data (justifying the resampling assumption). To answer the question regarding improvement in CPU utilization based on the results of Section III, we plot CPU utilization as a function of the batch CPU service time γ for various N, M. Figure 4a gives the results for $(N, M) = (2, 0), (2, 2), (2, 10), (2, \infty)$ and $(5, 0), (5, 2), (5, 10), (5, \infty)$. When the high-priority multiprogramming level $N = 2$, we see that the batch CPU times must be of the order of 10–20 ms before significant improvement in CPU utilization occurs, and for times in excess of 40 ms, almost complete CPU utilization is attained. For the case $N = 5$, the batch CPU times need to be 100–200 ms to get significant improvement, with 320 ms being the time for almost complete CPU utilization. These results can be anticipated using the rule of thumb developed in Remark (v) of Section III. For this example, the high-priority traffic experiences a

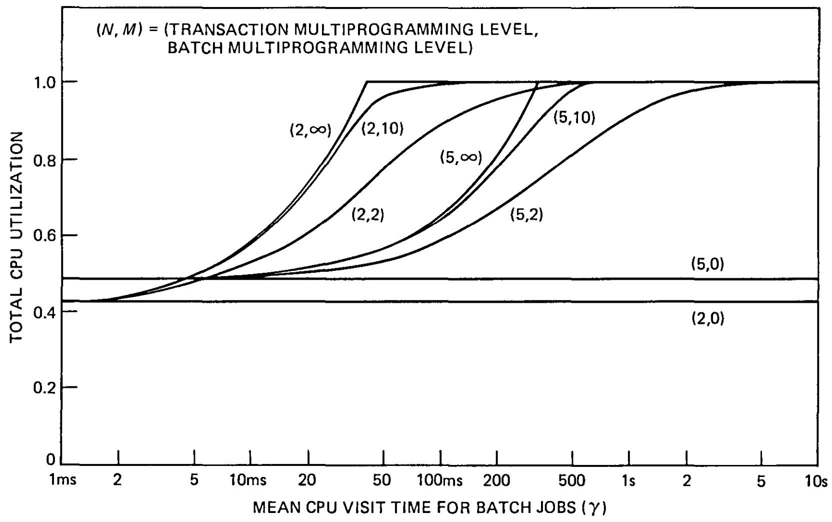


Fig. 4a—CPU utilization as a function of mean CPU batch time for various multiprogramming levels when transactions receive priority at both the CPU and I/O device.

bottleneck of strength 2. If this bottleneck is stronger, or the high-priority multiprogramming level is larger, our rule of thumb shows that the batch work has to be much more strongly CPU-bound to justify its introduction.

In the above arrangement, the batch work needs to be heavily CPU-bound to make its introduction worthwhile because batch I/O requests encounter the transaction bottleneck at the I/O device. Even though a batch job may only need infrequent I/O, it is often prevented from continuing by transaction I/O. In such circumstances, one might consider giving batch jobs high priority for I/O since, with appropriate parameters, batch jobs will rarely hold up transactions. The performance of such an arrangement is now evaluated using the results for model B. Figure 4b gives the results for the same parameters as before but with (N, M) = (2, 0), (2, 2), (2, ∞) and (5, 0), (5, 2); for this arrangement, we must show high-priority CPU utilization as well as total CPU utilization, since the former varies with M and γ. We observe that the introduction of batch jobs at a low multiprogramming level can yield a considerable increase in total CPU utilization. This increase can be accomplished with only a small effect on transaction throughput provided γ (the batch CPU service time) is 50 to 100 ms or larger. For γ comparable or smaller than the transaction CPU service time of 5 ms, a large degradation of transaction throughput occurs. If γ is smaller than 10 ms, then as M increases, transaction throughput approaches zero. For certain parameter combinations (e.g., $N = 5$, $\gamma = 100$ ms) the latter arrangement offers a larger improvement in total CPU utilization

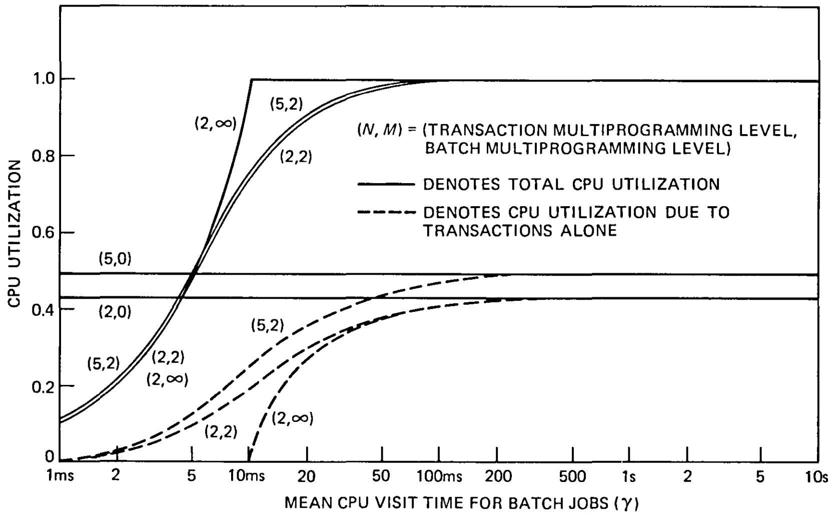


Fig. 4b—CPU utilization as a function of mean CPU batch time for various multiprogramming levels when transactions receive CPU priority and batch jobs receive I/O priority.

than the former, with only minor accompanying degradation of transaction throughput. In general, the extra total throughput offered by this "reversed priorities" arrangement must be weighed against any deterioration in transaction service, as quantified by the model. We mention that there may be a hazard in such a priority scheme since if all the batch jobs are simultaneously undergoing an abnormal flurry of I/O activity (or have actual mean service times substantially smaller than those being modeled) transaction processing might be temporarily halted as discussed in Remark (*iii*) of Section IV.

VI. DATA COMMUNICATIONS EXAMPLE

We consider a full-duplex communication channel terminated at end points labeled P and Q by front-end communication processors. We assume the following simple transmission protocol. Messages are transmitted from one endpoint to the other and individual short acknowledgments are returned in the reverse direction. The acknowledgments serve the dual purpose of error control and flow control, and an endpoint must stop transmitting when it has a number W of outstanding acknowledgments. Such a flow control scheme is often referred to as window flow control, and W the window length. This protocol has been studied by Reiser in the network context using closed queuing networks with suitable heuristics to approximate the effect of different message sizes at first-come first-served queues and prioritized acknowledgments.[5] Our models here are less sophisticated with a two-node queuing network representing a single full-duplex channel, but they do allow some exact results for two chains with different message sizes and priority. The questions we seek to answer relate primarily to the effect of providing two grades of service between points P and Q.

We assume that the end points P and Q return an acknowledgment as soon as they complete receiving a message. This is tantamount to assuming that the front-end processors are fast in comparison to the data links and have sufficient memory space so that acknowledgments are rarely withheld for purposes of flow control. We are also assuming that the data channels are essentially error free and retransmissions are rarely needed. Messages and acknowledgments are assumed to require an exponentially distributed time for transmission through the data channel. This assumption is more reasonable for messages than for acknowledgments where it is tolerable since acknowledgments are usually relatively short. There are two grades of service available, referred to as grades 1 and 2. Grade 1 service is regarded as having premium throughput characteristics, whereas grade 2 is designed to operate in a background mode to obtain increased use of the channel. We consider three configurations, referred to as schemes I–III, which are shown in Fig. 5.

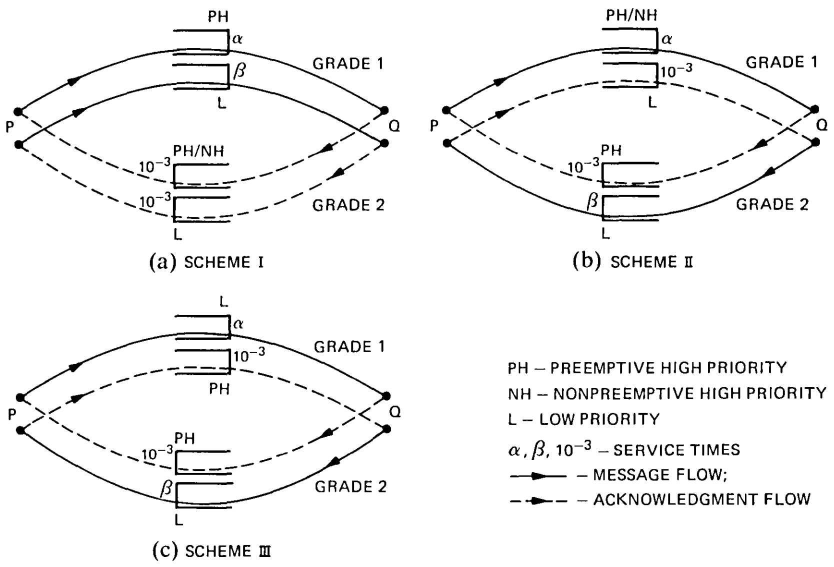


Fig. 5—Three schemes to prioritize a data link.

6.1 Three schemes to prioritize a data link

Scheme I

We consider a grade 1 transfer to be in progress from P to Q and evaluate the possibility of introducing grade 2 message flow in the same direction. In order that grade 1 service be minimally affected by the introduction of grade 2 service, we specify that grade 1 messages and acknowledgments receive higher transmission priority than grade 2 messages and acknowledgments. This configuration is shown in Fig. 5a. Preemption of message is reasonable in packetized or framed transmission where data streams are, or can be, broken into smaller parts for transmission. Depending on the implementation, preemption of acknowledgments may or may not be possible and we consider both possibilities. Hence, we use model A where we identify the left (respectively right) node with the Q-to-P (respectively P-to-Q) channel of the full-duplex link. Suppose an acknowledgment takes a mean of 1 ms for transmission, a grade 1 message an average of α seconds, and a grade 2 message an average of β seconds. Let the window size for grade 1 (respectively grade 2) messages be N (respectively M). Then taking, for example, $N = M = 4$, $\alpha = \beta = 3$ ms and allowing preemption of grade 2 acknowledgments, we find that grade 1 throughput is 330.58 messages/s and grade 2 throughput is 2.72 messages/s, i.e., grade 2 service accounts for only 0.8 percent of the total utilization of the P-to-Q channel. If we disallow preemption of acknowledgments, then the model described in Remark (*vii*) of Section III is applicable and yields 330.56 messages/s and 2.73 messages/s as approximations to the grade 1 and 2 throughputs, respectively. The criterion for validity of the approximation reads $1/1024 \ll 1$ in this case. These results are easily anticipated since grade 1 messages already utilize 99.2 percent of the P-to-Q channel and there is little point in introducing grade 2 service in the same direction.

Scheme II

When there is grade 1 message transfer from P to Q and only acknowledgment traffic from Q to P, it would seem worthwhile to carry grade 2 messages from Q to P assuming, of course, that there is a demand. As before, grade 2 messages receive lower preemptive priority and grade 2 acknowledgments lower preemptive or non-preemptive priority. This arrangement, shown in Figure 5b, calls for use of model A, but this time, with the left (respectively right) node identified with the P-to-Q (respectively Q-to-P) channel. We take the same definitions for N, M, α, β and a 1-ms acknowledgment time. Then with $N = M = 4$, $\alpha = \beta = 3$ ms and preemption of acknowledgments, we find that grade 1 throughput is 330.58 messages/s and grade 2 throughput is 7.14 messages/s. Without preemption of acknowledgments, throughputs become 328.99 and 11.87, respectively, and the criterion for validity of the approximation reads $1/64 \ll 1$. When β is increased to 60 ms, grade 2 throughput (with preemption of acknowledgments) becomes 5.63 messages/s but the messages are 20 times longer so total effective message data throughput is increased by 34 percent. Without preemption of acknowledgments, throughputs become 7.79 messages/s for grade 2 and 329.54 messages/s for grade 1. This is a 47 percent increase in data throughput. In this case, the criterion for validity of the approximation is $1/976 \ll 1$. As expected, grade 2 service in the opposing direction only attains a significant throughput when its messages are much longer than those of grade 1. When this is not the case, the grade 1 messages cause an impediment to grade 2 acknowledgments that prevents a worthwhile grade 2 throughput.

Scheme III

The priority given to both grade 1 messages and acknowledgments in Scheme II reflects a reluctance to allow grade 1 service to be more than minimally degraded by grade 2 service. The Q-to-P channel is underutilized because the grade 2 acknowledgments suffer the grade 1 bottleneck in the P-to-Q channel. But since acknowledgments are relatively short, we now ask how much grade 2 service improves and grade 1 service deteriorates when all acknowledgments receive priority over all messages. This arrangement is shown in Fig. 5c and N, M, α,

β are defined as previously. By using the results of model B, a 1-ms acknowledgment time and $\alpha = \beta = 3$ ms, grade 1 and 2 throughputs are found to be 248 messages/s and both channels are 99.4 percent utilized. Introducing grade 2 traffic has yielded a 50-percent increase in total traffic carried, but at the expense of a 25-percent degradation in grade 1 throughput. The effect of grade 2 acknowledgments on grade 1 messages is reduced when β is increased. For example, when β is 6 ms, grade 1 throughput is 292 messages/s and grade 2 throughput is 118 messages/s. Now grade 2 service has yielded a 60-percent increase in total message data throughput at the expense of a grade 1 throughput degradation of 12 percent. The introduction of grade 2 service has caused grade 1 message delay to increase from 10.65 ms to 12.30 ms and grade 1 acknowledgment delay remains at 1.45 ms (see Remark (*iv*) of Section IV). Utilization of the P-to-Q (respectively Q to P) channel is now 99.3 percent (respectively 99.95 percent).

As already stated in Section IV, a system with priorities allocated in such a way will tend to alternate between periods where customers of only one type (grade in this case) are processed. Hence, for this scheme, we need to make certain that during periods when grade 2 service is occurring grade 1 performance is not significantly disrupted in the short term. In this case, the fact that the 1-ms acknowledgment time is considerably shorter than β, makes it unlikely that a second grade 2 message will complete transmission before the acknowledgment from the former message is returned and, hence, that grade 1 service will be able to continue without an intolerably long delay. On the other hand, if the grade 2 source were to send a long sequence of very short messages, grade 1 communications could be interrupted for a considerable period. In an actual implementation, it might be desirable to incorporate a mechanism to prevent this occurrence.

Although we have only examined a limited set of parameter values, we can summarize the results of this section. In the absence of rather extreme traffic parameters, there is little justification for introduction of a lower grade of service which operates essentially in a background mode. On the other hand, if some degradation of the premium service grade is tolerable, then otherwise unused channel capacity can carry an appreciable amount of lower-grade traffic.

VII. COMPARISON WITH AN APPROXIMATION TECHNIQUE

Our final application will be an evaluation of a convenient and commonly used approximation technique for handling priorities.[3,4,5] The technique considers the low-priority customers at a node to have a dedicated server of rate $\mu_L\ (1 - \rho_H)$, where μ_L is the low-priority service rate and ρ_H is the utilization due to high-priority customers at that node. As noted in Ref. 5, this approximation is justifiable when the interruptions caused by high-priority traffic are frequent but of short duration. This suggests a criterion (sufficient condition) for satisfactory accuracy of the approximation: the high-priority busy cycle length at a node should be short in comparison with the low-priority service time at the same node.

Table I shows some results for model A with various parameter combinations N, M, ν, μ, λ. We tabulate the exact throughput T and mean delay D at each node for the low-priority customers using the results of Section II, and the approximations to these quantities based on the above approximation technique. We also tabulate the high-priority mean busy cycle B at each node to enable comparison of approximation accuracy with degree of satisfaction of the above criterion. For T, D, B the subscripts H, L distinguish high and low priority and l, r distinguish left and right nodes. Note that $B_{H,l} = (\nu - \nu^{-N})/(\nu - 1)$, $B_{H,r} = (\nu^{-1} - \nu^{N})/(1 - \nu)$.

In Table I note that when the criterion is satisfied at both nodes (cases 1, 2, 6, 7), the approximation is quite successful with errors of less than 2 percent. When the criterion is violated at one node only (cases 3, 8, 9), the approximate results might be regarded as satisfactory or unsatisfactory, depending on one's viewpoint. When the criterion is violated at both nodes (cases 4, 5, 10), both approximate throughputs and delays show large errors. Case 5 reflects a rather extreme choice of parameters and is included only to show the large errors which are theoretically possible.

Another vehicle for examining the effectiveness of the approximation technique is to compare it with the exact results for a homogeneous open network as considered in Section II. The approximation yields an expression for class i delay (including service time) at node j of

$$D_j^i \approx \mu_j^{-1} \Bigg/ \left(1 - \sum_{k=i}^{P} \rho_j^k\right),$$

which, in comparison with the exact result, is seen to be too small by a factor of

$$1 - \sum_{k=i+1}^{P} \rho_j^k .$$

Hence, for this type of network we would anticipate significant error if the approximation were applied to a priority class when higher-priority classes utilize a significant portion of a node's processing capacity. Indeed, the homogeneous network is a challenging test of the approximation technique since interruptions of a customer's service are of a duration at least comparable with the service time; our earlier criterion is never satisfied for such a network.

Table I—Comparison of the exact results for model A with an approximation technique

Case	N	M	ν	μ	λ	$B_{H,l}$	$B_{H,r}$	Technique	T_L	$D_{L,l}$	$D_{L,r}$
1	1	10	1.0	0.1	0.1	2	2.00	exact	0.0450	111	111
								approx.	0.0455	110	110
2	3	5	1	0.03	0.03	4	4.00	exact	0.00616	406	406
								approx.	0.00625	400	400
3	3	5	1	0.03	1.00	4	4.00	exact	0.750(−2)	656	10.5
								approx.	0.750(−2)	663	4.12
4	3	5	1	1.00	1.00	4	4.00	exact	0.139	18.0	18.0
								approx.	0.208	12.0	12.0
5	3	5	1	100.00	100.00	4	4.00	exact	0.416	6.12	6.12
								approx.	20.8	0.12	0.12
6	1	10	0.2	0.02	0.02	6	6.00	exact	0.00333	2923	77.5
								approx.	0.00333	2925	75.0
7	3	5	0.2	0.0005	0.01	156	6.24	exact	0.321(−5)	0.156(7)	128
								approx.	0.321(−5)	0.156(7)	125
8	3	5	0.2	0.1	0.01	156	6.24	exact	0.641(−3)	7646	156
								approx.	0.641(−3)	7664	136
9	3	5	0.2	0.0005	10.00	156	6.24	exact	0.321(−5)	0.156(7)	3.56
								approx.	0.321(−5)	0.156(7)	0.125
10	3	5	0.2	1.00	10.00	156	6.24	exact	0.00608	819	3.69
								approx.	0.00641	780	0.125

Note: (x) indicates $\times 10^{x}$

VIII. CONCLUSIONS

We have seen that the analysis of queuing networks is somewhat involved when local balance is not satisfied but that some useful results can still be obtained. It is clear that further results are needed to extend the applicability of these models. Section VII shows that further attention should also be directed towards establishing and improving the range of validity of existing approximation techniques.

REFERENCES

1. F. Baskett et al., "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," J. of ACM, *22,* No. 2 (April 1975), pp. 248–60.
2. F. P. Kelly, *Reversibility and Stochastic Networks*, New York: John Wiley, 1979.
3. M. Reiser, "Interactive Modeling of Computer Systems," IBM Systems Journal, No. 4 (1976), pp. 309–27.
4. K. C. Sevcik, "Priority Scheduling Disciplines in Queuing Network Models of Computer Systems," *Information Processing 77,* B. Gilchrist, Ed., IFIP, Amsterdam: North Holland Publishing Co., 1977.
5. M. Reiser, "A Queuing Network Analysis of Computer Communication Networks with Window Flow Control," IEEE Trans. on Comm., *COM-27,* No. 8 (August 1979), pp. 1199–1209.
6. N. K. Jaiswal, *Priority Queues*, New York: Academic Press, 1968.
7. B. Avi-Itzhak and D. P. Heyman, "Approximate Queuing Models for Multiprogramming Computer Systems," Operations Research, *21,* No. 6 (1973), pp. 1212–30.
8. W. J. Gordon and G. F. Newell, "Closed Queuing Systems with Exponential Servers," Operations Research, *15*(1967), pp. 254–65.
9. M. Reiser, "Mean Value Analysis of Queuing Networks, A New Look at an Old Problem," *Performance of Computer Systems*, M. Arato, A. Butrimenko, E. Gelenbe (Eds.), Amsterdam: North-Holland Publishing Company, 1979, pp. 63–77.
10. J. R. Jackson, "Jobshop-like Queuing Systems," Management Science, *10,* No. 1 (October 1963), pp. 131–42.
11. H. White and L. S. Christie, "Queuing with Preemptive Priorities or Breakdown," Operations Research, *6,* No. 1 (January 1958), pp. 79–95.
12. L. Kleinrock, *Communication Nets*, New York: McGraw-Hill, 1964.
13. L. Kleinrock, *Queueing Systems*, Vol. II, New York: John Wiley, 1976.
14. U. Herzog, L. Woo, and K. M. Chandy, "Solution of Queuing Problems by a Recursive Technique," IBM J. Res. Develop., (May 1975), pp. 295–300.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Efficient Realization Techniques for Network Flow Patterns

By F. R. K. CHUNG, R. L. GRAHAM and F. K. HWANG

(Manuscript received January 20, 1981)

In this paper, we describe several new techniques for use in the design of switched communications networks. These techniques apply to the development of traffic routes which realize network traffic flows in the context of an existing optimization method that assigns these flows. The general ideas involve the careful selection of basic variables and the successive reduction of the problem to one of convex hull formation in Euclidean n-space and finding Hamiltonian circuits for a class of highly structured graphs. We include several examples showing how these techniques are applied.

I. INTRODUCTION

Recently, R. H. Cardwell[1] proposed a switched communications network design algorithm for the future stored program control network. The networks under consideration are nonhierarchical in structure and take advantage of traffic noncoincidence in routing. The basic objective of Cardwell's algorithm is to design a minimum cost trunking network which, by using an appropriate routing strategy, can carry the necessary traffic load and, at the same time, meet the required grade of service.

In this paper, we describe an extremely efficient method for producing an appropriate routing strategy. One of our original intentions was to develop a mathematical framework into which dynamic routing problems, such as those described later, could be placed. Indeed, it seems likely that the approach used here may be valuable for examining other classes of such routing problems.

II. BACKGROUND

In Fig. 1 we show a block diagram of Cardwell's algorithm. Suppose we wish to design a network using the algorithm. (See Ref. 1 for a

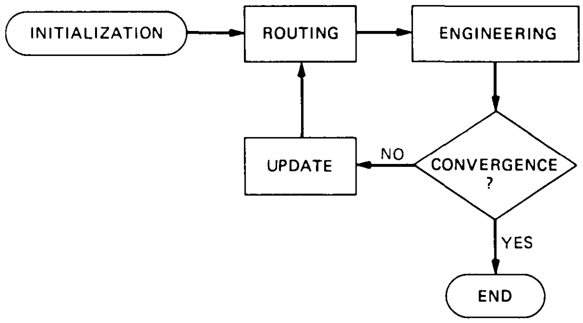


Fig. 1—Block diagram of Cardwell's algorithm.

more detailed description.) We start by initializing the blocking probabilities of each link. The routing module selects a set of the most economical paths for each pair of nodes and then assigns flow to the paths. Routes, which are ordered lists of paths, are then formed so that the probability that all paths in any list are busy is small enough to meet the required grade of service. Then, by means of a linear programming formulation, the routing module determines a network flow which minimizes the total cost, considering link costs and traffic noncoincidence. In the engineering module, the Erlang loss formula is then used to fix the number of trunks required for each link.[2] In the update module the ECCS method of Truitt is used to help minimize the network cost.[3] The blocking probabilities for all links are then updated. The whole process is now iterated until satisfactory convergence is achieved.

Figure 2 shows a block diagram of the routing module for the unified algorithm. A basic feature of our method is that actual routes are not formed until convergence has been obtained in an earlier part of the unified algorithm. Only after this occurs does the routing realization submodule generate the routes and provide the appropriate routing strategy. Refer to the work of Murray and Wong which gives efficient heuristic algorithms for solving the linear programming problems in this module.[4] The upper bound module is a new addition which helps

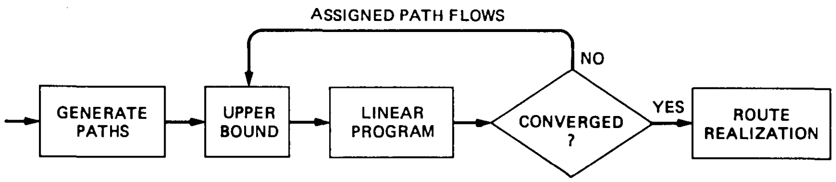


Fig. 2—Modified routing module.

the iterative procedure to converge more rapidly by setting stronger upper bounds for the carried loads of the various paths chosen.

One of the questions concerning this algorithm was the problem of synthesizing routes from the assigned path flows (the route realization block in Fig. 2). The following is a discussion of an efficient technique that accomplishes this.

III. CYCLIC ROUTING

Consider the special case of two nodes A and B. Assume there are n paths P_k, $1 \leq k \leq n$, between A and B. The amount of traffic to be carried on P_k is denoted by x_k, where we have normalized the traffic load so that one unit of traffic is attempted between A and B. The blocking probability for P_k is denoted by p_k. We will call the vector $\bar{x} = (x_1, \cdots, x_n)$ the desired traffic vector and $\bar{p} = (p_1, \cdots, p_n)$ the blocking probability vector. (The $\bar{x}$'s are actually outputs of the linear programming module.)

For a permutation π of $\{1, 2, \cdots, n\}$, by the route $R(\pi)$ generated by π, we mean the route in which the path $P_{\pi(1)}$ is tried and, if blocked, path $P_{\pi(2)}$ is tried. If that path is blocked, then path $P_{\pi(3)}$ is tried, etc.

The first question is: what are the traffic flows on the various P_k when route $R(\pi)$ is used? Let $q_k \equiv 1 - p_k$ and assume that π is the identity permutation, i.e., $\pi(k) = k$ for all k. Since one unit of traffic is initially attempted on P_1, the first path of $R(\pi)$, then q_1 units of traffic are carried on P_1 and p_1 units of traffic are blocked. These p_1 units are now attempted on P_2. Thus, $p_1 q_2$ units get carried and $p_1 p_2$ are blocked. Continuing this process, we see in general that on P_k, $p_1 p_2 \cdots p_{k-1} q_k$ units of traffic are carried and $p_1 p_2 \cdots p_{k-1} p_k$ are blocked. We condense this information into the flow vector $\mathbf{F}(\pi) = (F_1(\pi), F_2(\pi), \cdots, F_n(\pi)) = (q_1, p_1 q_2, p_1 p_2 q_3, \cdots, p_1 p_2 \cdots q_n)$. Note in particular that the amount of traffic which is blocked is just $p_1 p_2 \cdots p_n$, independent of π.

The overall plan is to use each route $R(\pi)$ a certain fraction $\alpha(\pi)$ of the time, as π ranges over all permutations of $\{1, 2, \cdots, n\}$, so as to achieve the desired traffic flow x_k on each P_k. In other words, if possible, find $\alpha(\pi)$ with

$$\alpha(\pi) \geq 0, \sum_{\pi} \alpha(\pi) = 1$$

so that

$$\bar{x} = \sum_{\pi} \alpha(\pi) \, \mathbf{F}(\pi),$$

where π ranges over all permutations of $\{1, 2, \cdots, n\}$.

This is exactly the same problem as deciding whether $\bar{x}$ is in the convex hull of points $\mathbf{F}(\pi)$ (considered as points in n-dimensional Euclidean space E^n) and, if so, finding a representation of $\bar{x}$ as a convex combination of the $\mathbf{F}_{(\pi)}$. Note that all the $\mathbf{F}(\pi)$'s are extreme points of the convex hull. Since

$$\sum_k \mathbf{F}_k(\pi) = 1 - p_1 p_2 \cdots p_n \text{ for any } \pi,$$

then the convex hull is actually (at most) an $(n - 1)$-dimensional polytope. Thus, any point in the convex hull can be represented as a convex combination of some choice of n extreme points $\mathbf{F}(\pi)$.

As an example, we consider in detail the case $n = 3$. In Table I, we list the six possible π's and the corresponding $\mathbf{F}(\pi)$'s.

We will denote the permutation π which sends i to $\pi(i)$ by the sequence $\pi(1)\pi(2) \cdots \pi(\mathrm{n})$. This should not be confused with the ordinary cycle notation for a permutation π (which will also be used). For example, the permutation of $\{1, 2, 3, 4, 5, 6\}$ given by $\pi(1) = 3$, $\pi(2) = 5$, $\pi(3) = 6$, $\pi(4) = 4$, $\pi(5) = 2$, $\pi(6) = 1$ can be written both as $\pi = (136)(25)(4)$ and $\pi = 3\ 5\ 6\ 4\ 2\ 1$.

Figure 3 shows a typical picture when these points are plotted in E^3. All six points lie on the plane $F_1 + F_2 + F_3 = 1 - p_1 p_2 p_3$. We should note here that we always assume $0 < p_k < 1$ for all k, since any path with blocking probability one can be removed without affecting the traffic flow, and any path which carries any traffic at all has positive blocking probability (less than one).

In general, we would like to be able to decide if the desired traffic vector $\bar{x}$ lies in the convex hull of $\mathbf{F}(\pi)$ and, if so, how to represent it as a convex combination of $\mathbf{F}(\pi)$. A natural choice to consider is a cyclic set of routes. For example, suppose we consider the three routes:

$$\pi_1 = 1\ 2\ 3,$$

$$\pi_2 = 2\ 3\ 1,$$

$$\pi_3 = 3\ 1\ 2.$$

Let us determine whether $\bar{x}$ is in the convex hull of these three points. Of course, a necessary condition is $\sum_i x_i = 1 - p_1 p_2 p_3$. In any case,

Table I—Flow vectors for the Case $n = 3$

π	$\mathrm{F}(\pi)$
1 2 3	$(q_1, p_1q_2, p_1p_2q_3)$
1 3 2	$(q_1, p_1q_2p_3, p_1q_3)$
2 1 3	$(q_1p_2, q_2, p_1p_2q_3)$
2 3 1	$(q_1p_2p_3, q_2, p_2q_3)$
3 1 2	$(q_1p_3, p_1q_2p_3, q_3)$
3 2 1	$(q_1p_2p_3, q_2p_3, q_3)$

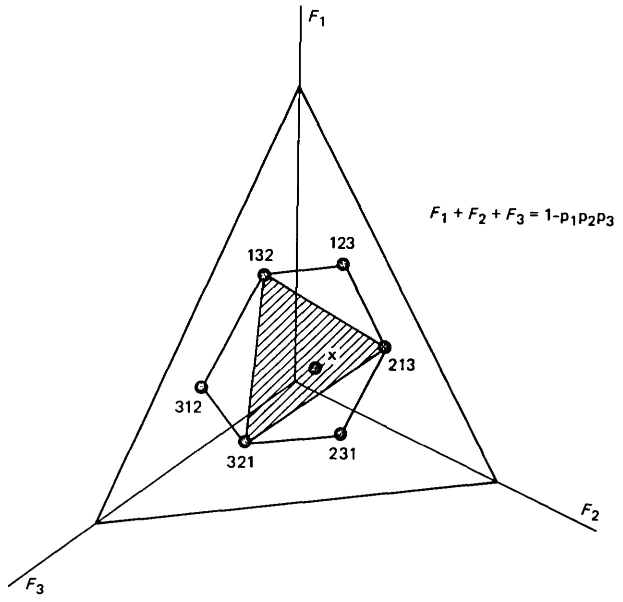


Fig. 3—Geometrical representation of the flow vectors for the Case $n = 3$.

since the convex hull is 2-dimensional, any point in it is a convex combination of some set of three extreme points. The cyclic sets seem reasonable choices since they apparently span rather large portions of the convex hull, although certainly not all of it. For example, in Fig. 3 we have shaded the convex hull of $F(\pi_1)$, $F(\pi_2)$, and $F(\pi_3)$. This is much larger than, say, the triangle spanned by $F(123)$, $F(132)$ and $F(231)$.

Therefore, we are looking for coefficients α_i such that

$$\sum_{i=1}^{3} \alpha_i \mathbf{F}(\pi_i) = \bar{x}, \tag{1}$$

with

$$\alpha_i \geq 0, \sum_i \alpha_i = 1.$$

By eq. (1), the α_i must satisfy

$$\sum_{i=1}^{3} \alpha_i F_k(\pi_i) = x_k, \; k = 1, 2, 3.$$

Expanding these equations using Table I, we obtain

$$\begin{aligned} \alpha_1 q_1 + \alpha_2 q_1 p_2 p_3 + \alpha_3 q_1 p_3 &= x_1, \\ \alpha_1 p_1 q_2 + \alpha_2 q_2 + \alpha_3 p_1 q_2 p_3 &= x_2, \\ \alpha_1 p_1 p_2 q_3 + \alpha_2 p_2 q_3 + \alpha_3 q_3 &= x_3. \end{aligned} \tag{2}$$

The determinant Δ of the system eq. (2) is given by

$$\Delta = \begin{vmatrix} q_1 & q_1p_2p_3 & q_1p_3 \\ p_1q_2 & q_2 & p_1q_2q_3 \\ p_1p_2q_3 & p_2q_3 & q_3 \end{vmatrix}$$

$$= q_1q_2q_3 \begin{vmatrix} 1 & p_2p_3 & p_3 \\ p_1 & 1 & p_1p_3 \\ p_1p_2 & p_2 & 1 \end{vmatrix}$$

$$= q_1q_2q_3(1 - p_1p_2p_3)^2.$$

Solving for the α_k, we have

$$\alpha_1 = \frac{1}{\Delta} \begin{vmatrix} x_1 & q_1p_2p_3 & q_1p_3 \\ x_2 & q_2 & p_1q_2q_3 \\ x_3 & p_2q_3 & q_3 \end{vmatrix}$$

$$= [(x_1/q_1) - (p_3x_3/q_3)]/(1 - p_1p_2p_3),$$

$$\alpha_2 = [(x_2/q_2) - (p_1x_1/q_1)]/(1 - p_1p_2p_3),$$

$$\alpha_3 = [(x_3/q_3) - (p_2x_2/q_2)]/(1 - p_1p_2p_3).$$

Letting

$$\sigma_k = \frac{x_k}{q_k}, \; \delta_k = \frac{p_kx_k}{q_k},$$

we see that the α_k are ≥ 0 if $\sigma_1 - \delta_3 \geq 0$, $\sigma_2 - \delta_1 \geq 0$, $\sigma_3 - \delta_2 \geq 0$.

For general n, a similar calculation shows that the corresponding system of n equations has determinant Δ given by

$$\Delta = q_1q_2 \cdots q_n(1 - p_1p_2 \cdots p_n)^{n-1}$$

and coefficient values

$$\alpha_{k+1} = [(x_{k+1}/q_{k+1}) - (p_kx_k/q_k)]/(1 - p_1p_2 \cdots p_n)$$

for the cyclic set of routes

$$\begin{array}{c} 1\,2\,3\,4 \cdots\cdot\, n \\ 2\,3\,4 \cdots\cdot\, n\,1 \\ 3\,4 \cdots\cdot\, n\,1\,2 \\ \vdots \\ n\,1\,2 \cdots\cdot\, n-1 \end{array}$$

where addition of indices is modulo n, i.e.,

$$\alpha = (\sigma_1 - \delta_n)/(1 - p_1 \cdots p_n).$$

Consequently, we succeed with this cyclic choice of routes if all the α_k's are at least 0, i.e.,

$$\sigma_2 \geq \delta_1, \sigma_3 \geq \delta_2, \cdots, \sigma_n \geq \delta_{n-1}, \sigma_1 \geq \delta_n. \tag{3}$$

Note that $\sum_i \alpha_i = 1$ follows at once from

$$\sum_i x_i = 1 - p_1 \cdots p_n$$

and, in particular, note that the labeling of the P_k is arbitrary. Any arrangement of the σ's and δ's satisfying eq. (2) will give us a cyclic set of routes which works, i.e., a set of routes which contains $\bar{x}$ in the convex hull.

In order to find these efficiently, we can do the following: From the given x_k and p_k form

$$q_k = 1 - p_k,$$

$$\sigma_k = \frac{x_k}{q_k},$$

$$\delta_k = \frac{p_k x_k}{q_k}.$$

We are just searching for a cyclic permutation $(j_1, j_2, \cdots, j_n)$ of $\{1, 2, \cdots, n\}$ such that

$$\sigma_{j_2} \geq \delta_{j_1}, \sigma_{j_3} \geq \delta_{j_2}, \cdots, \sigma_{j_1} \geq \delta_{j_n}.$$

To find this, form the directed graph G which has as its vertex set the set of paths P_k and an edge from each P_i to P_j for which $\sigma_j \geq \delta_i$. If in G we find a Hamiltonian circuit (i.e., a circuit passing through each vertex exactly once), say, $P_{j_1}P_{j_2} \cdots P_{j_n}(P_{j_1})$, then by the definition of the edges of G, we must have

$$\sigma_{j_2} \geq \delta_{j_1}, \sigma_{j_3} \geq \delta_{j_2}, \cdots, \sigma_{j_1} \geq \delta_{j_n},$$

which is precisely what we want.

Thus, we have shown that $\bar{x}$ can be realized from a cyclic set of routes if and only if G has a Hamiltonian circuit. Of course, the problem of finding a Hamiltonian circuit in an arbitrary graph is known to be an *NP*-complete problem (see Ref. 5 for an exposition of this term) and, therefore, almost certainly computationally intractable as the graph becomes large. Fortunately, however, the graphs G are far from arbitrary and, in fact, we can provide an algorithm for finding Hamiltonian circuits in them which runs in time $O(n \log n)$.

First, we may assume without loss of generality (by a suitable relabeling) that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n.$$

Note that a necessary condition for the existence of a Hamiltonian circuit in G is:

For all k, $2 \leq k \leq n$,

$$|\{i:\sigma_k \geq \delta_i\}| \geq n - k + 2, \tag{4}$$

where $|X|$ denotes the cardinality of the set X. To see this, note that if G has a Hamiltonian circuit, then for each k, there is at least one edge from a vertex in $\{P_k, P_{k+1}, \cdots, P_n\}$ to one in $\{P_1, P_2, \cdots, P_{k-1}\}$. Thus,

$$\sigma_k \geq \sigma_{t'} \geq \delta_{t''}$$

for some t', t'' with $t' \geq k > t''$. Therefore,

$$\{i:\sigma_k \geq \delta_i\} \supseteq \{i:i \geq k\} \cup \{t''\},$$

which implies

$$|\{i:\sigma_k \geq \delta_i\}| \geq n - k + 2.$$

In fact, eq. (4) is also a sufficient condition for G to be Hamiltonian. This can be seen from the following proof (by induction on n).

Suppose $n = 2$ and eq. (4) holds. Then clearly $\sigma_2 \geq \delta_1$ and G is Hamiltonian. Next, assume that eq. (4) is sufficient for all such graphs with $n - 1$ vertices. Suppose G has n vertices and satisfies eq. (4). Let G' be the induced subgraph on $\{P_1, P_2, \cdots, P_{n-1}\}$ (where we have assumed as usual that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$). It is easy to see that G' also satisfies eq. (4). By induction, G' has a Hamiltonian circuit, say, $P_{j_1}P_{j_2} \cdots P_{j_{n-1}}$. Since G satisfies eq. (4),

$$\sigma_n \geq \delta_{j_i} \text{ for some } i,\ 1 \leq i \leq n - 1.$$

But also

$$\sigma_k \geq \sigma_n \geq \delta_n \text{ for all } k,\ 1 \leq k \leq n - 1.$$

Thus,

$$P_{j_1} \cdots P_{j_i} P_n P_{j_{i+1}} \cdots P_{j_{n-1}}$$

is a Hamiltonian circuit in G and the induction step is completed. This proves that eq. (4) is, in fact, a necessary and sufficient condition for G to have a Hamiltonian circuit.

We summarize the preceding discussion in the following.

Algorithm for cyclic routing

Inputs: Paths $P_1, \cdots, P_n$ joining two given points A and B, and the corresponding blocking probability vector $\bar{p} = (p_1, \cdots, p_n)$ and desired traffic vector $\bar{x} = (x_1, \cdots, x_n)$.

Object: To find a permutation π of $\{1, 2, \cdots, n\}$ satisfying

$$\bar{x} = \sum_{i=1}^{n} \alpha_i F(\pi_i)$$

with

$$\alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i = 1,$$

where F is the flow vector function and π_i is the cyclic route i, $\pi(i)$, $\pi^{(2)}(i), \cdots, \pi^{(n-1)}(i)$ (i.e., P_i is tried first, then $P_{\pi(i)}$, etc.).

Algorithm:

(i) Calculate

$$\sigma_k = \frac{x_k}{1 - p_k} \text{ and } \delta_k = \frac{p_k x_k}{1 - p_k} \text{ for } 1 \leq k \leq n.$$

(Recall that we are always assuming that $0 < p_k < 1$ for all k.)

(ii) Relabel the σ_k, if necessary, so that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$.

(iii) Set $\pi \leftarrow (1)$, $i \leftarrow 2$, $y \leftarrow \delta_1$, $z \leftarrow 1$.

(iv) If $\sigma_i < y$, go to (vii). If $\sigma_i \geq y$, insert i after z in the cycle representation of π.

(v) If $y > \delta_i$, set $y \leftarrow \delta_i$, $z \leftarrow i$. If $y \leq \delta_i$, y and z are unchanged. If $i < n$, set $i \leftarrow i + 1$ and go to (iv).

(vi) The desired Hamiltonian circuit is $P_1 P_{\pi(1)} P_{\pi^{(2)}(1)} \cdots P_{\pi^{(n-1)}(1)}$. Define

$$\alpha_k = \frac{\sigma_k - \delta_{\pi^{-1}(k)}}{1 - p_1 p_2 \cdots p_n}, 1 \leq k \leq n.$$

$\bar{x}$ can be realized by using route π_k for the fraction α_k of the time, $1 \leq k \leq n$.

(vii) $\bar{x}$ cannot be realized by any set of cyclic routes.

End.

Note that except for (ii), in which $n \log_2 n$ operations are required in the ordering of the n σ_i's, all other steps require at most $O(n)$ operations. Thus, the computational complexity of the algorithm is $n \log_2 n + O(n)$ in time and $O(n)$ in space.

We point out that the desired traffic flow vector $\bar{x}$ can often be realized by more than one set of cyclic routes, i.e., the graph G might have more than one Hamiltonian circuit (each of which corresponds to a cyclic routing realization). The preceding algorithm will always produce one such realization provided any exists at all.

Two examples

Example 1: There are five paths between two points A and B. The desired traffic vector $\bar{x}$ and the blocking probability vector $\bar{p}$ are as follows:

$$\bar{x} = (0.185, 0.231, 0.220, 0.072, 0.242)$$

$$\bar{p} = (0.8, 0.7, 0.6, 0.5, 0.3).$$

Thus,

$$\bar{\sigma} = (0.924, 0.770, 0.550, 0.144, 0.346)$$

$$\bar{\delta} = (0.740, 0.539, 0.330, 0.072, 0.104).$$

The corresponding graph G is shown in Fig. 4.

From the algorithm, we find the Hamiltonian circuit, $P_1P_2P_3P_4P_5$, corresponding to the permutation $\pi = (12354)$. Therefore, we have the values shown in Table II.

The routing strategy is to use route π_i for the fraction α_i of the time.

Example 2: There are also five paths between A and B. However, the desired traffic vector $\bar{x}'$ and the blocking probability vector $\bar{p}'$ are slightly different from those in Example 1.

$$\bar{x}' = (0.191, 0.231, 0.220, 0.072, 0.242)$$

$$\bar{p}' = (0.7, 0.7, 0.6, 0.6, 0.3).$$

Thus,

$$\bar{\sigma}' = (0.638, 0.770, 0.550, 0.144, 0.346)$$

$$\bar{\delta}' = (0.445, 0.539, 0.330, 0.072, 0.104).$$

The corresponding graph G' is shown in Fig. 5.

From our algorithm, we find the Hamiltonian circuit, $P_2P_1P_3P_5P_4$, corresponding to the permutation $\pi' = (21354)$. Therefore, we have the values shown in Table III.

It is easily verified that

$$\bar{x}' = \sum_{i'=1}^{5} \alpha_{i'}\mathbf{F}(\pi_{i'}').$$

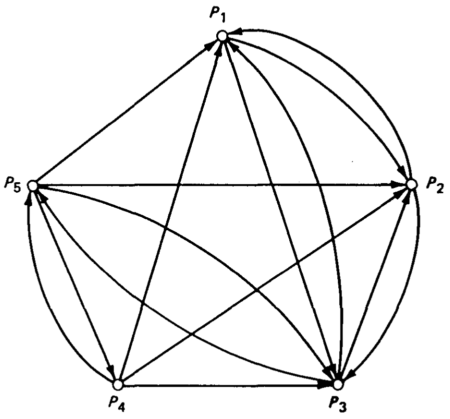


Fig. 4—Corresponding graph G for Example 1.

Table II—Values of coefficients for Example 1

i	Route π_i	Coefficient α_i
1	$P_1P_2P_3P_5P_4$	0.897
2	$P_2P_3P_5P_4P_1$	0.032
3	$P_3P_5P_4P_1P_2$	0.012
4	$P_4P_1P_2P_3P_5$	0.042
5	$P_5P_4P_1P_2P_3$	0.017

Note that there is another Hamiltonian circuit in G', namely, $P_1P_2P_3P_5P_4$, which gives an alternative cyclic routing realization as shown in Table IV.

Again, it is easy to verify that

$$\bar{x}' = \sum_{i=1}^{5} \alpha_i'' \mathbf{F}(\pi_i'').$$

IV. CYCLIC APPROXIMATIONS

As we mentioned earlier, the desired traffic vector $\bar{x}$ determined by the linear programming module can perhaps not be realized by a cyclic set of routes. In that case, we provide a routing strategy for approximating $\bar{x}$ by modifying our cyclic routing algorithm. This is most easily explained in terms of an example (this one was taken from data generated by a 28-point simulation of Cardwell[6]).

In this example, there are 8 paths from A to B. Table V shows the appropriate data.

Note that path 1 assumes the full traffic load, i.e., $x_1 = 1 - p_1$, which can be achieved if and only if every call requested is first attempted on path 1. In fact, this is a typical case of the existence of a least expensive direct line between two cities in a large toll switching network.

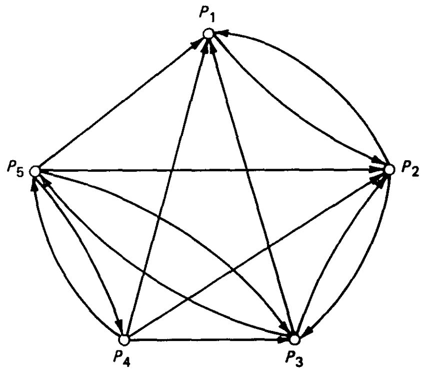


Fig. 5—Corresponding graph G' for Example 2.

Table III—Values of coefficients for Example 2

i	Route π_i'	Coefficient α_i'
1	$P_1P_3P_5P_4P_2$	0.103
2	$P_2P_1P_3P_5P_4$	0.730
3	$P_3P_5P_4P_2P_1$	0.109
4	$P_4P_2P_1P_3P_5$	0.042
5	$P_5P_4P_2P_1P_3$	0.017

There are several reasons why this $\bar{x}$ cannot be realized by cyclic routing. For example,

(i) Path 1 assumes the maximum possible traffic load, i.e., $x_1 = 1 - p_1$; this cannot happen with cyclic routing.

(ii) Traffic flow is highly unevenly distributed; in particular, paths 3, 6, and 8 get no traffic at all.

(iii) $\sum_i x_i \neq 1 - p_1p_2 \cdots p_8$.

Let us form the graph G as described in the cyclic routing realization, namely, G has vertices $\{P_1, P_2, \cdots, P_8\}$ and there is a directed edge from P_i to P_j if $\delta_i \leq \sigma_j$ (see Fig. 6).

Here, G has no Hamiltonian circuit (and, in fact, is not even connected). In this case, we approximate $\bar{x}$ by taking a combination of (possibly trivial) disjoint circuits in G. The precise way this is done is described by the following algorithm.

Algorithm for approximate cyclic routing

(*i*) Calculate

$$\sigma_k = \frac{x_k}{1 - p_k} \text{ and } \delta_k = \frac{p_k x_k}{1 - p_k}, \; 1 \leq k \leq n.$$

(*ii*) Relabel the σ_k, if necessary, so that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$. If $\sigma_n = 0$, define t by $\sigma_t > 0 = \sigma_{t+1}$, if $\sigma_n = 0$. Otherwise, define t to be n.

(*iii*) Set $\pi_1 \leftarrow 1$, $j \leftarrow 1$, $i \leftarrow 2$, $y \leftarrow \delta_1$, $z \leftarrow 1$.

(*iv*) If $y > \sigma_i$, go to (*vi*). If $y \leq \sigma_i$, insert i after z in the cycle representation of π_j.

Table IV—Alternative values of coefficients for Example 2

i	Route π_i''	Coefficient α_i''
1	$P_1P_2P_3P_5P_4$	0.591
2	$P_2P_3P_5P_4P_1$	0.339
3	$P_3P_5P_4P_1P_2$	0.012
4	$P_4P_1P_2P_3P_5$	0.042
5	$P_5P_4P_1P_2P_3$	0.017

Table V—Example not realized by cyclic routing

	$\bar{p}$	$\bar{x}$	$\bar{\sigma}$	$\bar{\delta}$	Approximation
1	0.30696	0.69304	1.00000	0.30696	0.69304
2	0.23850	0.00084	0.00111	0.00026	0.01832
3	0.30935	•	•	•	0.00233
4	0.45325	0.07555	0.13818	0.06263	0.06989
5	0.28685	0.13343	0.18710	0.05367	0.12343
6	0.33891	•	•	•	0.00116
7	0.60274	0.09685	0.24378	0.14694	0.08959
8	0.48781	•	•	•	0.00196

(*v*) If $y > \delta_i$, set $y \leftarrow \delta_i$, $z \leftarrow i$. If $y \leq \delta_i$, y and z are unchanged. If $i < t$, go to (*iv*). If $i = t$, go to (*vii*).

(*vi*) Set $j \leftarrow j + 1$, $\pi_j \leftarrow i$, $y \leftarrow \delta_i$, $z \leftarrow i$. If $i < t$, set $i \leftarrow i + 1$ and go to (*iv*). If $i = t$, go to (*vii*).

(*vii*) The routing strategy is given by using $\pi_1, \pi_2, \cdots, \pi_s$ as follows. Let

$$\beta_{i,k} = \sigma_k - \delta_{\pi_i^{-1}(k)},$$

$$\alpha_{i,k} = \frac{\beta_{i,k}}{\sum\limits_j \beta_{i,j}} \text{ for } k \in \pi_i.$$

Use the route

$$\pi_{1,i_1}\pi_{2,i_2} \cdots \pi_{s,i_s}$$

for the fraction

$$\alpha_{1,i_1}\alpha_{2,i_2} \cdots \alpha_{s,i_s}$$

of the time where

$$\pi_{k,i_k} = [i_k, \pi_k(i_k), \pi_k^{(2)}(i_k), \cdots].$$

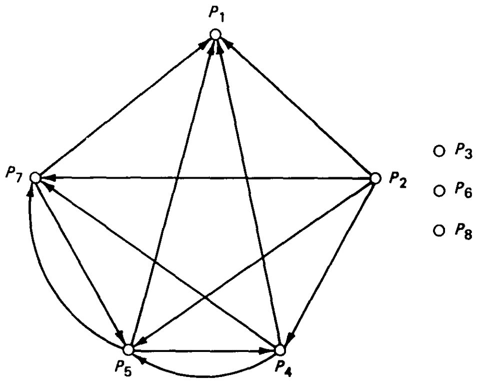


Fig. 6—Corresponding graph G for the Cardwell example.[6]

Continuing the example, we apply the above algorithm to the values in Table V. This results in the permutations $\pi_1 = (1)$, $\pi_2 = (547)$, and $\pi_3 = (2)$. The corresponding α's are given in Table VI. Note that we only use 5 of the 8 paths and the total blocking probability is 0.0057. If the blocking probability turns out to be too high to meet the required grade of service, we can make use of the remaining three paths in carrying the overflow by the modification shown in Table VII. The traffic flow generated by this routing strategy is listed in Table V under Approximation. Note that the approximation to the desired traffic flow is quite good.

As we pointed out before, one of the main reasons we cannot achieve the desired traffic flow exactly is that this is inherently impossible to do using convex combinations of the available routes, because of premature termination or inadequate constraints in the linear program. In the next section, we examine a method for correcting this difficulty.

Table VI—Values of coefficients for the Cardwell example

Route	Coefficient
$P_1P_5P_4P_7P_2$	$\alpha_{1,1}\alpha_{2,5}\alpha_{3,2} = 0.13132$
$P_1P_4P_7P_5P_2$	$\alpha_{1,1}\alpha_{2,4}\alpha_{3,2} = 0.27634$
$P_1P_7P_5P_4P_2$	$\alpha_{1,1}\alpha_{2,7}\alpha_{3,2} = 0.59234$

VII. UPPER BOUNDS

Again, we consider a set of paths $P_1, P_2, \cdots, P_n$ connecting two points and having blocking probabilities $p_1, p_2, \cdots, p_n$, respectively. The traffic flow on path P_i cannot exceed its capacity, namely, $1 - p_i$. Thus, an immediate upper bound on x_i for any realizable traffic flow vector $\bar{x}$ is

$$x_i \leq 1 - p_i \quad \text{for all} \quad i.$$

Similarly, for any two paths P_i and P_j, the total amount of traffic they can carry is $1 - p_ip_j$. Thus, if $\bar{x}$ is realizable then

$$x_i + x_j \leq 1 - p_ip_j.$$

Table VII—Values of coefficients for modified routes of the Cardwell example

Route	Coefficient
$P_1P_5P_4P_7P_2P_6P_8P_3$	0.13132
$P_1P_4P_7P_5P_2P_3P_6P_8$	0.27634
$P_1P_7P_5P_4P_2P_8P_3P_6$	0.59234

More generally, for any set of k indices $i_1, \cdots, i_k$, if $\bar{x}$ is realizable then

$$x_{i_1} + \cdots + x_{i_k} \leq 1 - p_{i_1} \cdots p_{i_k}. \tag{5a}$$

Furthermore, for $k = n$, eq. (5) must hold with equality, i.e.,

$$x_1 + \cdots + x_n = 1 - p_1 \cdots p_n. \tag{5b}$$

It is interesting to note that conditions (5a) and (5b) are also sufficient conditions for the realizability of $\bar{x}$ as a convex combination of flow vectors $\mathbf{F}(\pi)$. The proof is not difficult. Basically, it is as follows. Suppose $\bar{x}$ is an extreme point of the polytope ρ defined by the intersection of the half planes (5a) and the hyperplane (5b). Then $\bar{x}$ must satisfy at least one of the equalities in eq. 5a with equality, say without loss of generality.

$$x_1 + \cdots + x_r = 1 - p_1 \cdots p_r, \quad r < n. \tag{6}$$

We now use induction on n and express the point $\bar{x}' = (x_1 \cdots x_r)$ as a convex combination of the $r!$ flow vectors associated with paths $P_1, \cdots, P_r$. We next consider all the inequalities in eq. (5) which contain $x_1, \cdots, x_r$ as well as other x's. Typically, we might have

$$x_1 + \cdots + x_r + x_{j_1} + \cdots + x_{j_s} \leq 1 - p_1 \cdots p_r p_{j_1} \cdots p_{j_s}.$$

By eq. (6)

$$x_{j_1} + \cdots + x_{j_s} \leq p_1 \cdots p_r(1 - p_{j_1} \cdots p_{j_s}).$$

Again, we can use induction, this time on the new variables $y_k = x_k/p_1 \cdots p_r$, $r < k \leq n$, which satisfy the required analogues of eqs. (5a) and (5b). Finally, we piece together these two convex combinations to get the desired representation for $\bar{x}$. Since ρ is convex, then we are finished.

Of course, in actual practice some appropriate subset of the inequalities in eq. (5a) would be used in the upper bounding process (see Ref. 1).

VIII. CONCLUSIONS

In this paper we give necessary and sufficient conditions for determining whether a desired traffic flow vector (as specified by the linear programming solution portions of the algorithm) can be realized from a cyclic set of routes. The algorithm to verify the necessary and sufficient conditions can be implemented in $O(n \log n)$ time. When the conditions are not met, we propose an approximation method which uses several smaller cycles rather than a single cyclic set of routes.

In connection with the results we described earlier, it would be of interest to know what proportion of the volume of the polytope spanned by the $n!$ flow vectors $\mathbf{F}(\pi)$ can in general be reached by *cyclic* routes. For $n = 3$, it seems that we can always cover at least ⅔ of the volume (actually, area in this case; see Figure 3). We have not yet examined the general case. In fact, we do not even know whether or not cyclic routes always span a positive fraction of the volume (independent of n).

IX. ACKNOWLEDGMENT

The authors have benefited greatly from discussions with G. Ash, R. Cardwell, V. Mummert, and other members of the Traffic Network Planning Department, who not only provided us with our first exposure to the general problem, but who were instrumental in extending and integrating the techniques we describe into an overall trunk network design algorithm which shows great promise.

REFERENCES

1. G. R. Ash, R. H. Cardwell, and R. P. Murray, "Design and Optimization of Networks With Dynamic Routing," B.S.T.J., this issue.
2. D. Bear, *Principles of Telecommunication Traffic Engineering*, London: Peter Peregrinus, Ltd., 1976, p. 38.
3. C. J. Truitt, "Traffic Engineering Techniques for Determining Trunk Requirements in Alternate Routing Trunk Networks," B.S.T.J., *33*, No. 2 (March 1954), pp. 277–302.
4. R. P. Murray and R. T. Wong, private communication.
5. M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco: Freeman and Co., 1979.
6. R. H. Cardwell, unpublished work.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Design and Optimization of Networks With Dynamic Routing

By G. R. ASH, R. H. CARDWELL, and R. P. MURRAY

(Manuscript received March 6, 1981)

The growth of electronic switching systems and the high-capacity interoffice signaling network provide an opportunity to extend telephone network routing rules beyond the conventional hierarchy. Network models are described that illustrate the savings inherent in designing networks for dynamic, nonhierarchical routing. An algorithm for engineering such networks is discussed, and the comparative advantages of various path-routing and progressive-routing techniques are illustrated. A particularly simple implementation of dynamic routing called two-link dynamic routing with crankback is discussed and is shown to yield benefits comparable to much more complicated routing schemes. The efficient solution of embedded linear programming (LP) routing problems is an essential ingredient for the practicality of the design algorithm. We introduce an efficient heuristic optimization method for solution of the LP routing problems, which greatly improves computational speed with minimal loss of accuracy. We also project computational requirements for a 200-node design problem, which is the estimated size of the intercity Bell System dynamic routing network in the 1990s.

I. INTRODUCTION AND SUMMARY

The rapidly growing stored program control (SPC) network, consisting of electronic switching systems interconnected by common-channel interoffice (CCIS) signaling links, provides a significant opportunity to extend the telephone network routing rules beyond the conventional hierarchy. In the SPC network, there are no restrictions to hierarchical route choices or to routing rules which remain fixed in time, but we may rationally consider network configurations which use dynamic, nonhierarchical routing (DNHR). The term dynamic describes routing techniques which are time-sensitive, as opposed to present-day hierarchical routing rules which are time-fixed. An important variable in the dynamic routing strategy is the frequency with which network routing rules are updated.

1.1 Savings possibilities with dynamic routing

There are two major opportunities to improve the planned network design (forecast) with more advanced routing techniques. First, because of its fixed nature, present hierarchical routing cannot really take much advantage of load variations which arise from business/residence, time zones, seasonal variations, and other reasons. By allowing time varying, or dynamic routing, some of this penalty can be reduced. Second, the present hierarchical routing has rigid path choices, plus low blocking on final links which limit flexibility and reduce efficiency. If we choose paths based primarily on cost and relax the present rigidity in network structure, a more efficient network should result. The upper limits on improvement in these two areas are discussed first.

1.1.1 Noncoincidence effects

It is estimated from a 28-node intercity network model (Fig. 1) that about 20 percent of the network's first cost can be attributed to designing for time varying loads using our present static hierarchical routing techniques. To show this, we first designed a hierarchical network using a conventional cluster busy-hour approach. Then, to quantify the extra capacity being provided, we also designed the 28-node model for the individual hourly loads. These hourly networks were obtained by using each hourly load, and ignoring the other hourly loads, to dimension a hierarchical network that would perfectly match that hour's load. This procedure results in 17 separate network designs, one for each hour.

Figure 2 is a plot of the normalized network cost (including switching and facility cost) required for the cluster busy hour and hourly network designs. On the top line, the cluster busy-hour solution had a network capital cost of one unit to satisfy all 17 hours of load with fixed, hierarchical routing. The 17 hourly networks, shown on the lower curve, represent the normalized capital cost of the circuit miles and trunks actually required at each hour to satisfy the load. Three network busy periods are visible: morning, afternoon, and evening. We can also see a noon-hour drop in load, and an early-evening drop as the business day ends and residential calling begins in the evening. The hourly network curve separates the capacity provided in the cluster busy-hour solution into two components: below the curve is the capacity actually needed at each hour to meet the load; above the curve is the capacity which is available but is not needed at that hour. This additional capacity exceeds 20 percent of the total network capacity

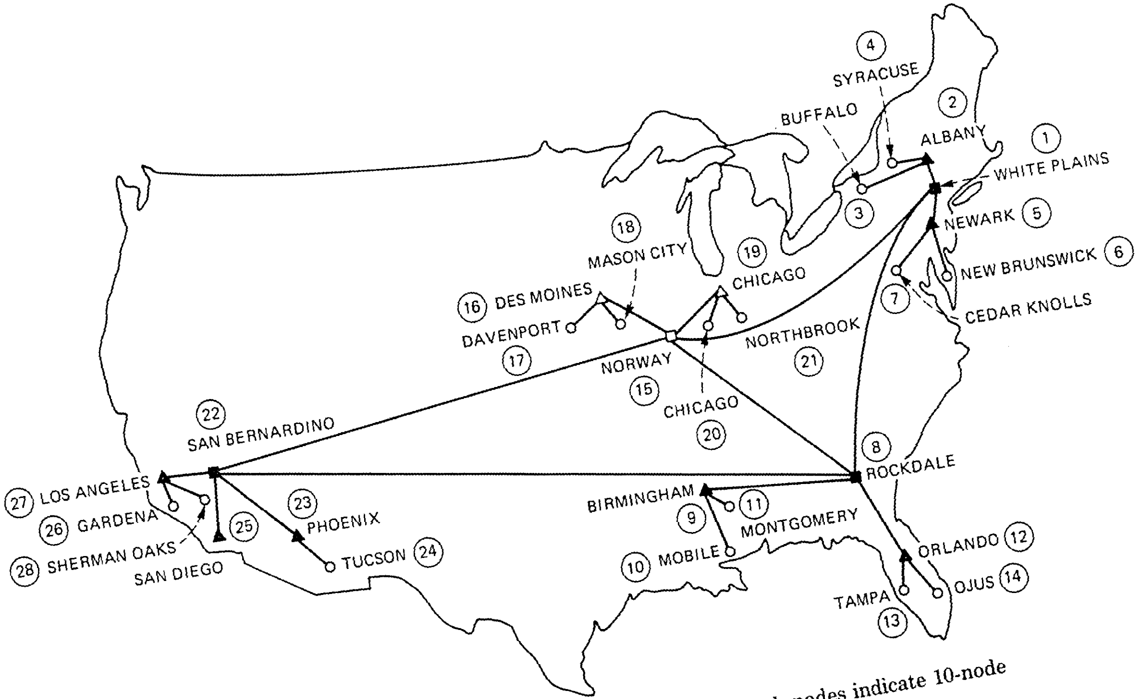


Fig. 1—Intercity network model for 28-node network. (Dark nodes indicate 10-node model.)

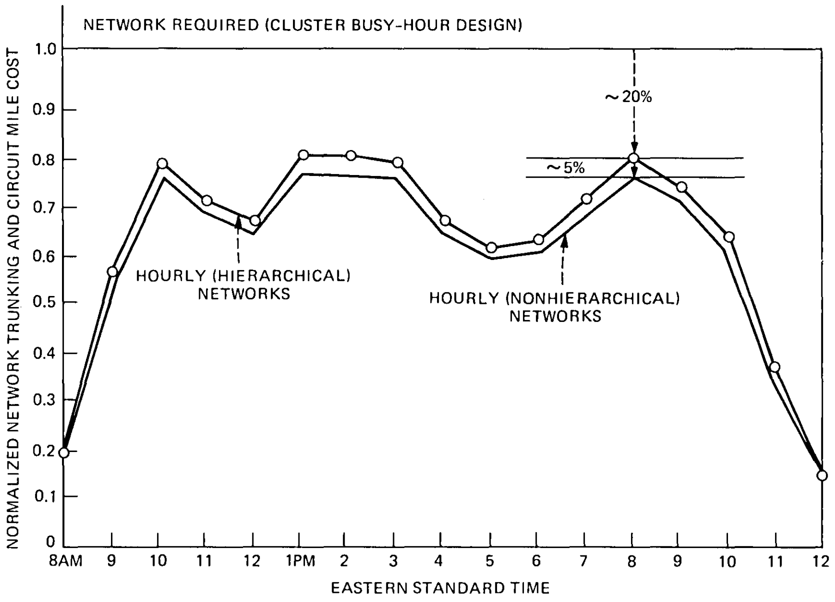


Fig. 2—Network first cost for 28-node network.

through all hours of the day. This gap represents the capacity put in the network to meet noncoincident loads, and suggests a maximum limit on network reduction which might be achieved through improved routing techniques.

1.1.2 Limited path selection effects

Additional benefits can be provided in network design by allowing a more flexible intercity routing plan that is not restricted to hierarchical routes. Our approach allows the selection of shortest (nonhierarchical) paths. Applied to each hourly load, this approach yields an overall savings of about 5 percent in comparison to the hierarchical hourly networks. Figure 2 also displays these results and shows that the 20 percent bound discussed above has increased to a total of 25 percent. This additional savings potential translates into actual benefits by introducing nonhierarchical shortest path routing into the design, as is done in the DNHR network design algorithm.

Figure 3 illustrates the limitation that the hierarchy imposes in the 28-node network between San Diego and Birmingham. The alternate paths between these points go through two regional centers, San Bernardino, Ca. and Rockdale Ga., providing relatively long paths. Selecting more direct paths, for example the Tucson, and Phoenix, Az. and Montgomery, Al. paths, would provide design benefits. Allowing the optimum choice of intercity routes beyond the hierarchical choices (i.e., nonhierarchical networks) yields design savings. This includes allowing the present final paths to use alternate routing, which in many cases would further improve the network efficiency.

1.2 Summary

In Section 3.2, we describe the route formulation of the unified algorithm (UA). In this formulation the allowed traffic patterns (routes) are formed for each point-to-point demand prior to traffic assignment in the routing optimization step. Three routing methods are considered in designing networks using the route formulation method:

(*i*) Progressive routing in which a call progresses through the network one switch at a time without retracing its path until it either reaches its destination or arrives at an intermediate switch from which it has no outlet.

(*ii*) Multilink path routing in which a call blocked by a busy trunk group on a path may use the capabilities of the SPC network to be "cranked back" to the originating node and attempt the next path in the route.

(*iii*) Two-link path routing, which is identical to multilink routing, except that a path from origin to destination may have at most two links.

We find that design savings on the order of 10–15 percent are possible when using these routing methods as compared to present hierarchical techniques. From the savings results and implementation considerations, we conclude that two-link routing is preferred.

We next consider another formulation of the UA called the path formulation, which is specifically tailored to examine two-link routing options. This method does not preselect allowable routes, but allows the traffic allocation step to assign traffic directly to paths in order to minimize network cost. Routes are formed after the optimization step to realize the desired flows. A flow feasibility algorithm is described

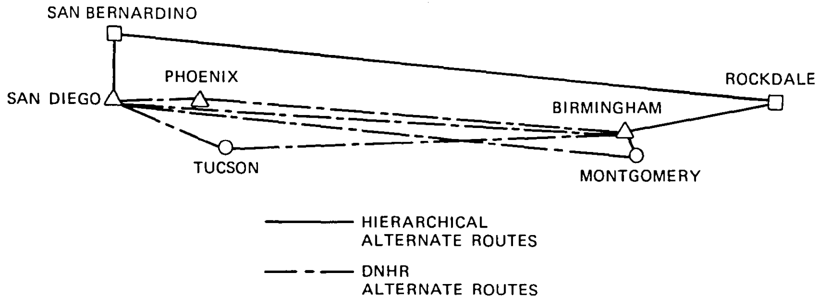


Fig. 3—Shortest path choice.

which forces the resulting path flows to be realizable. Three two-link routing methods for realizing the optimum path flows are then considered, varying in complexity from a very flexible method, CGH routing (developed by Chung et al)[1], to a very simple method called sequential routing. The latter method consists of offering all traffic to an ordered list of two-link paths with the overflow from one path being offered to the next path; the ordered list may change by time-of-day to take advantage of traffic noncoincidence.

We find that the routing techniques investigated using the path formulation achieve at least 1–2 percentage points additional savings over the routing techniques studied using the route formulation. We then find that sequential routing incurs an insignificant cost penalty when compared to more flexible routing schemes and, because of its simplicity, we conclude that sequential routing is the preferred routing method.

Efficient optimization techniques are considered in Section IV. These methods allow the design of very large networks for dynamic routing using reasonable computer resources. Finally, potential Bell System applications are discussed in Section V.

II. DYNAMIC ROUTING CONCEPTS: DESIGN, SERVICING, AND CONTROL

Figure 4 illustrates the three primary components of the network design and administration functions as three interacting feedback loops around the network. The network offered load is shown to consist of predictable, average demand components, unknown forecast errors, and day-to-day variation components. The feedback controls function to regulate the service provided by the network through capacity and routing adjustments. Network design (or planned servicing) operates over a year-long interval, drives the network capacity expansion, and preplans routing patterns to minimize network costs. Demand servicing accounts for the existing capacity and, on a weekly basis, fine-tunes link sizes and routing patterns to account for forecast errors inherent in the year-long design loop. Real-time control makes limited adjustments to the preplanned routing patterns to account for normal daily shifts in load patterns.

Network provisioning for dynamic routing depends primarily on performing off-line calculations for network design and demand servicing. The off-line calculations select the optimal routing patterns from a very large number of possible alternatives in order to minimize the trunking network cost. The term dynamic routing frequently suggests an extensive search for the optimal routing assignment to be performed in real time. This extensive search is in fact being made but most of the searching is performed in advance using an off-line design system

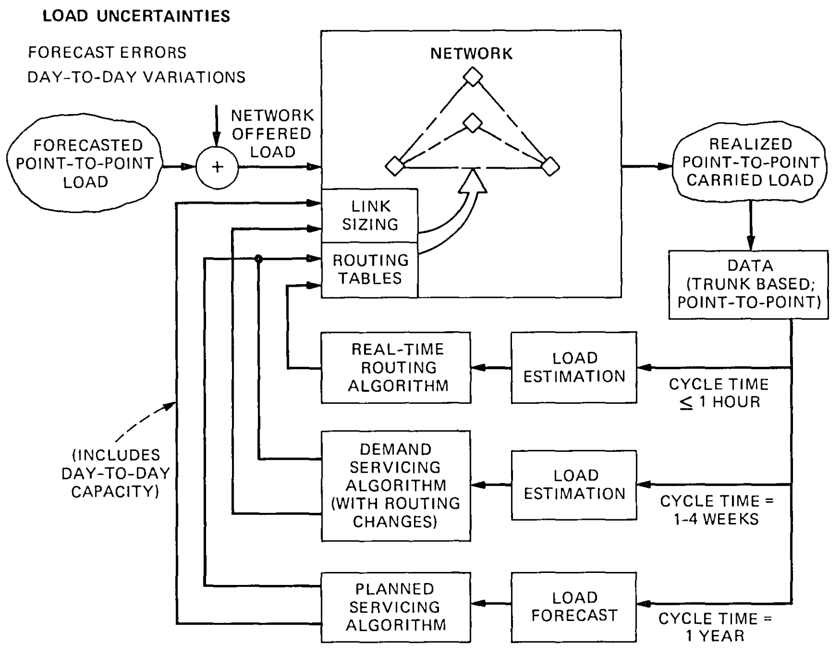

LOAD UNCERTAINTIES
FORECAST ERRORS
DAY-TO-DAY VARIATIONS
FORECASTED POINT-TO-POINT LOAD
NETWORK OFFERED LOAD
+
NETWORK
LINK SIZING
ROUTING TABLES
REALIZED POINT-TO-POINT CARRIED LOAD
DATA (TRUNK BASED; POINT-TO-POINT)
REAL-TIME ROUTING ALGORITHM
LOAD ESTIMATION
CYCLE TIME ≤ 1 HOUR
(INCLUDES DAY-TO-DAY CAPACITY)
DEMAND SERVICING ALGORITHM (WITH ROUTING CHANGES)
LOAD ESTIMATION
CYCLE TIME = 1-4 WEEKS
PLANNED SERVICING ALGORITHM
LOAD FORECAST
CYCLE TIME = 1 YEAR

Fig. 4—Planned servicing, demand servicing, and real-time control as interacting feedback loops around the network.

and an off-line demand servicing system. The effectiveness of the design depends on how accurately we can forecast the expected load on the network. Errors associated with the forecast are corrected in the demand servicing process described in the companion article.[2] The only routing decisions necessary in real time involve conditions that also become known in real time: day-to-day load variations, network failures, and network overloads. Procedures for real-time routing are also described in the companion article.

III. DESIGN ALGORITHM

3.1 Overview

In this section, we describe the algorithm used to design near minimum cost nonhierarchical networks using dynamic routing. This algorithm is termed UA because it combines into one systematic procedure various network design concepts, such as

(*i*) Using time-sensitive dynamic routing to take advantage of traffic noncoincidence,

(*ii*) Routing traffic along the least costly paths,

(*iii*) Favoring large, more efficient trunk groups, (*iv*) Using efficient trunk group blocking levels determined by the economic hundred call seconds (ECCS) method, and

(*v*) Minimizing incremental network cost.

The first two concepts were described in Section 1.1. A brief description of the other three concepts incorporated in the UA is given below.

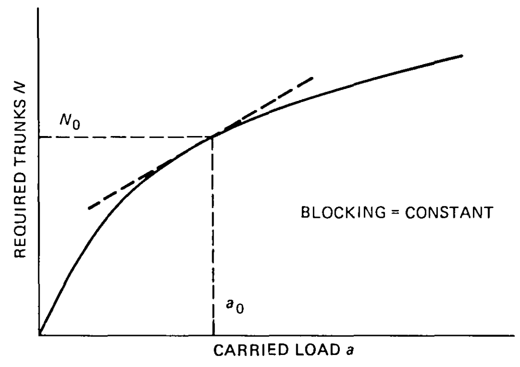
3.1.1 Favoring large trunk group

Figure 5 illustrates the number of trunks, N, required to carry a particular carried load, a, at constant blocking. From the shape of the curve comes the well-known fact that at constant blocking the number of additional trunks required to carry an increment of offered load decreases as the trunk group size increases. Hence, it is advantageous to combine several traffic parcels into one large parcel to be routed over a large trunk group since one large trunk group is inherently more efficient than several smaller trunk groups.

In the UA, larger trunk groups are favored through the use of a link incremental cost metric proportional to the slope ($\partial N/\partial a$) of the trunks versus load curve. Thus, the link metric indicates the attractiveness of this link to carry additional traffic.

3.1.2 Use efficient blocking levels

Figure 6 illustrates the cost trade off between carrying traffic on the direct trunk group between A and B, and the alternate network that overflow calls will use. The problem is to find the optimum value of blocking (or, equivalently, the number of trxnks) to handle the offered load at a minimum cost. This question was first answered by Truitt[3] who derived the concept of an ECCS based on the direct path to



LINK METRIC $M_l \propto \frac{C\partial N}{\partial a}$, WHERE C = COST PER TRUNK

Fig. 5—Efficiency of large trunk groups.

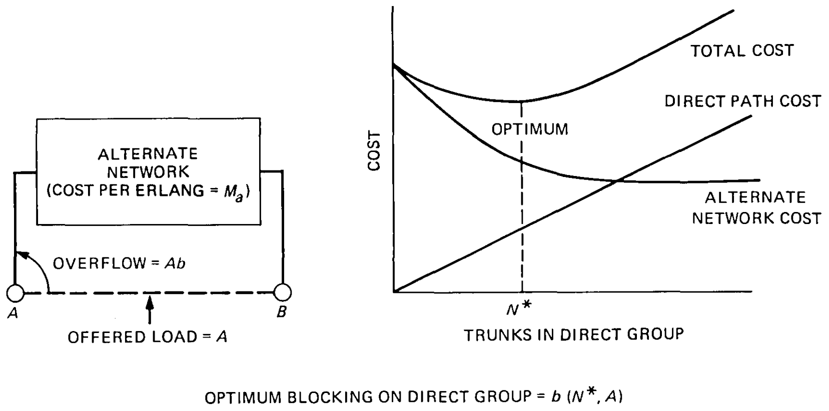


Fig. 6—Optimum blocking (ECCS method).

alternate path cost ratio and the marginal capacity of the alternate path. Truitt's ECCS method is commonly used today in both intercity and metropolitan network design. This method is also used in the UA.

3.1.3 Minimize incremental network cost

Network cost and performance are nonlinearly related. Hence, the network design problem is inherently a nonlinear programming problem. To avoid the complexities associated with nonlinearity, the network cost function can be linearized around the present operating point and the linearized (incremental) cost function minimized to yield a minimum cost network.

This approach of minimizing the incremental network cost has been successfully used by other investigators. Yaged[4] has used this technique to find a near minimum cost facility network to satisfy trunk demands when the facility links display a concave facility cost versus channel capacity relationship. For his problem, Yaged demonstrated that this technique satisfied the Kuhn-Tucker conditions which are necessary (but not sufficient) for optimality. An analogous approach was used by Knepley[5] who applied the minimal incremental cost concept to the design of the automatic voice network (AUTOVON).

Figure 7 shows the iterative loop for the route formulation of the UA. Basic input parameters include trunk cost, point-to-point offered loads, and required point-to-point grade-of-service (GOS).

The router finds the shortest paths (sequences of links) between points in the network. Using assumed link blocking levels, the router then forms the paths into candidate routes (sequences of paths) and determines the proportion of flow appearing on each path in the route for each unit of offered load. This method of forming routes from

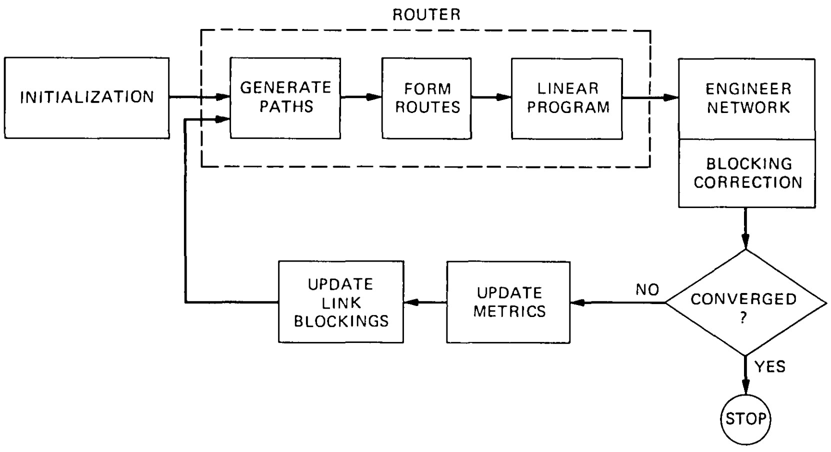


Fig. 7—Unified algorithm iterative loop.

assumed link blockings is a key feature of the UA. It eliminates the nonlinear relation between link blocking, number of trunks, and offered load from the optimization step, and it also permits investigation of a wide variety of routing schemes.

The LP then assigns flow to the candidate routes to minimize network cost. The output from the router is the optimum routing plan consisting of the routes to be used in each hour. This routing is provided to the engineering program which determines the flow on each link and sizes the link to meet the design level of blocking used in the router step. Once the groups have been engineered, the cost of the network can be evaluated and compared to the last iteration.

If the network cost is still decreasing, the update module (*i*) computes the slope of the capacity versus load curve on each link and updates the link cost using this slope as a weighting factor, and (*ii*) computes a new level of link blocking using the ECCS method. The new link lengths and blockings are fed to the router which again selects shortest paths, and so on.

3.2 Detailed description

3.2.1 Initialization

An initial set of link blockings and metrics are calculated based on the ECCS method. Initial link blockings are determined assuming that the overflow path is the shortest two-link path between the endpoints with a marginal capacity of 28 CCS.

3.2.2 Router

The router consists of both a route generator and an LP. The route generator constructs a set of candidate routes for each point-to-point demand pair in each design hour. Each route candidate contains just enough paths to meet the GOS constraint. The LP then selects which routes will be used in each hour and in what proportion.

Since the method of constructing routes depends on the routing discipline (progressive, multilink, or two-link) to be used, we defer the discussion of how these various routes are formed to their respective sections. For now we assume that the route generator forms the proper number of routes for each demand pair, and calculates the portion of route carried load on each link for the routing discipline used. The operation of the UA is such that almost any routing scheme can be used, merely by using the proper route generator.

The second step in the router is the LP, which assigns the offered traffic to the candidate routes in order to minimize the network incremental cost.

First, we introduce the following notation:

L = number of links.
K = number of demand pairs.
H = number of design hours.
J_k^h = number of routes for demand pair k in hour h.
P_{jk}^{ih} = proportion of carried load on route j for point-to-point demand pair k on link i in hour h.
M_i = incremental link cost metric in terms of dollar cost per erlang of carried traffic for link i.
R_k^h = offered load to demand pair k in hour h.
r_{jk}^h = carried load on route j of demand pair k in hour h.
A_i^h = offered load to link i in hour h.
a_i = maximum carried load on link i over all hours.
g_{jk}^h = route blocking on route j of demand pair k in hour h.
b_i^h = blocking on link i in hour h.

Then the LP will select the r_{jk}^h and the resulting a_i so as to minimize

$$\sum_{i=1}^{L} M_i a_i$$

subject to

$$\sum_{k=1}^{K} \sum_{j=1}^{J_k^h} P_{jk}^{ih} r_{jk}^h \le a_i \qquad i = 1, 2, \cdots, L \qquad h = 1, 2, \cdots, H$$

$$\sum_{j=1}^{J_k^h} \frac{r_{jk}^h}{1 - g_{jk}^h} = R_k^h \qquad h = 1, 2, \cdots, H \qquad k = 1, 2, \cdots, K$$

$$r_{jk}^h \ge 0, \ a_i \ge 0.$$

Inputs to the LP are P_{jk}^{ih} and g_{jk}^h from the route generator, M_i from the previous metric calculation, the link blockings b_i^h, and the R_k^h. Outputs from the LP are the r_{jk}^h, the assignment of carried load to the routes, and a_i, the associated link capacity (maximum carried load). In many cases, the IBM LP package (MPSX-370) was used to obtain the results reported here. All point-to-point traffic was first assigned to its least expensive route to form a feasible solution to the LP; this solution was used as a starting basis. In those cases where a heuristic optimization method (HOM) (see Section IV) is used to solve the LP, the output will be a set of r_{jk}^{h}, which approximates the optimal route flows.

3.2.3 Network engineering

After the LP has assigned traffic to routes, the network must be engineered to achieve a link blocking no higher than the assumed blocking used as input to the router. In this way, the GOS constraint will be satisfied, or at least the GOS will be no worse than that calculated by the router. If the GOS is not satisfactory, it is corrected by the blocking correction algorithm described below.

To arrive at a consistent set of hourly blockings and offered loads, an iteration scheme is used. The iteration uses the present estimates of the link offered loads to size each link in its peak hour and calculate blocking estimates in side hours. After all groups have been sized, new proportions of carried load are calculated using the blocking estimates and the routing pattern given by the LP. The link flows are then recalculated and the process repeated. The iteration is continued until the sum of the absolute blocking changes is less than a prescribed convergence threshold. Engineering can be accomplished either by using a single parameter traffic model or a two-parameter traffic model. Results given in this article are for the single-parameter case. Fractional trunks were allowed so as to achieve the required blocking exactly. This stabilizes the iterative loop and speeds convergence.

3.2.4 Blocking correction algorithm

If a route blocking in the engineered network exceeds a threshold, the blocking on the first path is decreased until the route blocking is equal to the desired GOS. The additional traffic which must be carried to reduce the route blocking to the desired GOS will, thus, be carried on the path which has the minimum incremental cost, and the network cost increase required to correct the route blocking should be close to minimal.

Once an engineered network solution is obtained, the route blockings needing correction are rank ordered and the highest route blocking is corrected first. After the new link blockings are obtained, routes are once again checked for blocking violations and the entire process is repeated until an engineered network solution is found which does not violate the route blocking constraint. The blocking correction has been made part of the engineering loop as shown in Fig. 7.

3.2.5 *Calculation of new metrics*

The expression for the link metric is $C_i \partial N_i / \partial a_i$, or the cost per trunk multiplied by the rate of change of trunks required to keep the blocking constant with a changing carried load. Hence, this is the incremental cost to carry an increment of load at constant blocking on link i. In particular, the partial derivative is approximated by

$$M_i = \frac{C_i[N_i(a_i + \Delta a_i) - N_i(a_i)]}{\Delta a_i},$$

where

C_i = cost of one trunk on link i
$N_i(a)$ = trunks required on link i for carried load a (for the link blocking b_i)
Δa_i = incremental carried load (normally set to 5 percent of a_i)

3.2.6 *Calculation of more efficient blockings*

The ECCS approach of Truitt[3] is used (Fig. 6) to calculate ECCS values in the UA. The objective is to calculate the number of trunks, N^*, (and, hence, the link blocking, B) that will minimize the total cost of carrying load A over the combination of the direct path and the alternate paths. To do this the network cost is first written as:

$$\begin{aligned} \text{Cost} &= CN + \alpha M_a \\ &= CN + AbM_a, \end{aligned} \tag{1}$$

where α is the overflow load from link AB ($\alpha = Ab$) and M_a is an equivalent metric for the alternate route network. A partial derivative is taken of eq. 1 with respect to N and the resulting expression set equal to zero to obtain the minimum.

3.3 *Candidate routing methods*

3.3.1 *Progressive routing*

Progressive routing is familiar since the Bell System hierarchy is an example of progressive routing. In this scheme, when a call is sent from one node to another node, the control of the call is also passed to the next node. No crankback to a previous node is allowed, but the call must continue toward its destination at each stage, or be blocked. The main difficulty with progressive routing is to avoid looping. In the hierarchy this is prevented automatically by the structure of the network. In our nonhierarchical design, the assumption was made that the history of the call could be carried via CCIS. In that way, the electronic switching processor would know the nodes to which the call had already been routed, and disallow them as the next outlet choice.

Besides preventing looping, route control is also used to promote efficient trunk use. Basically, we prohibit excessive alternate routing which can result in calls routing on paths with many links, thus "stealing" trunks from calls which can complete on one or two links. This situation has a cascading effect and can result in inefficient trunk use, with fewer call completions than otherwise possible. To promote efficient trunk use, we eliminate paths with a large number of links which are unnecessary to meet the required GOS.

In the dynamic version of progressive routing, traffic is allocated to the most economical next node choices on a time varying basis.

3.3.1.1 Route proportions and blocking. A simple example of the computation of route blocking and proportions is given in Fig. 8. From the assumed blocking on each link and the progressive routing pattern, the load offered to, and overflowing from, each link is calculated. From this information, the route blocking and proportions are determined.

3.3.2 Multilink path routing

Path routing implies selection of an entire path between points in the network before a connection is actually provided on that path. If a connection on one link in a path is blocked, the call then seeks

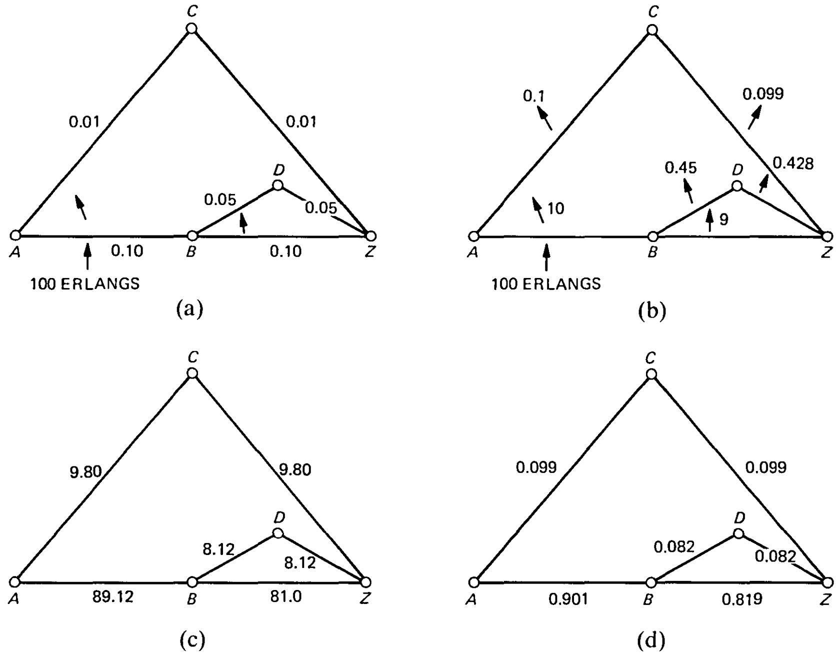


Fig. 8—Example of progressive route proportions. (a) Routing and link blocking. (b) Overflow loads (total carried load = 98.9 erlangs) (c) Link-carried loads. (d) Link proportions.

another complete path. Implementation of such a routing technique could be done through control from the originating office, plus a multiple link crankback capability to allow paths of greater than two links to be used. Path-to-path routing is nonhierarchical, and allows the choice of the most economical paths rather than being restricted to hierarchical paths.

Dynamic path routing is achieved by allocating fractions of the traffic to routes, and allowing the fractions to vary as a function of time. To generate more than one route for each point-to-point pair, one approach is to use cyclic routing. This method has as its first route $(1, 2, \cdots, M)$, where the notation (i, j, k) means all traffic is offered first to path i, which overflows to path j, which overflows to path k. The second route of the cyclic router is a cyclic permutation of the first route: $(2, 3, \cdots, M, 1)$. The third route is likewise $(3, 4, \cdots, M, 1, 2)$ and so on. This approach has computational advantages because its cyclic structure requires considerably fewer calculations to find the proportions for all routes than does a general collection of paths. The route blockings of cyclic routes are identical; what varies from route to route is the proportion of flow on the various links.

3.3.2.1 Route proportions and blocking. Figure 9 illustrates that some links may be common to more than one path and, hence, route blocking calculations and route carried flow calculations can become involved. From the assumed blocking on each link and the path-to-path routing pattern, the load offered to, and overflowing from, each link is calculated and from this information the route blocking and proportions are determined. More complicated routes are handled by a method given in Ref. 6.

3.3.3 Two-link path routing

In the design of multilink path networks, about 98 percent of the traffic was routed on one- and two-link paths even though paths of greater length were allowed. Because of switching costs, paths with one or two links are usually less expensive than paths with more links. Therefore, two-link path routing was introduced and uses the greatly simplifying restriction that paths can be two links in length at most. It requires only single-link crankback to implement and uses no common links, but is otherwise identical to the multilink scheme. It achieves nearly the same network savings as multiple-link path routing, and appears to be very attractive as a network routing alternative. Computation of route proportions is greatly simplified for two-link routing, since common links cannot occur on one route.

3.4 Route formulation results and conclusions

We consider here the cost of a 10-node subset of the 28-node network

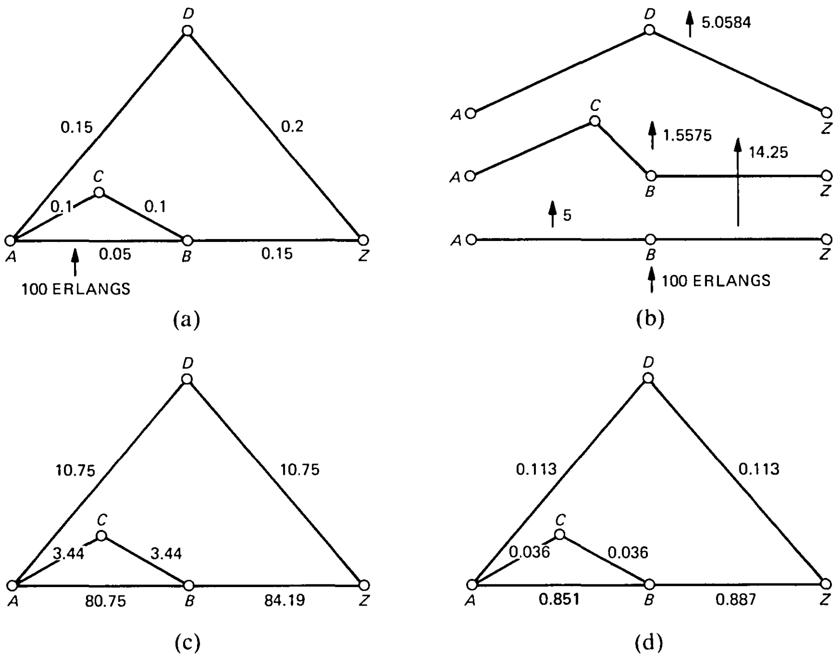


Fig. 9—Example of multilink proportions. (a) Link blockings: routing is *ABZ* → *ACBZ* → *ADZ*. (b) Path overflow loads and blocking: carried load = 10.749 erlangs and blocking = 0.32 for path *ADZ*; carried load = 3.4425 erlangs and blocking = 0.3115 for path *ACBZ*; carried load = 80.75 erlangs and blocking = 0.1925 for path *ABZ*. (c) Link-carried loads; route carried load = 94.94 erlangs. (d) Link proportions.

(Fig. 1) designed for multihour loads. Results for large networks are in general agreement with these results. We illustrate designs for hierarchical, progressive, multilink, and two-link networks to satisfy the traffic loads for a single hour of load and also for three network busy hours (10 a.m., 1 p.m., and 8 p.m.). The 10-node hierarchical networks were designed using current standard practices. In the design of DNHR networks, the IBM mathematical programming system, MPSX-370, was used to solve the necessary LP in the multihour design and was run to optimality in each iteration (this is feasible for the 10-node network problem). The GOS objective was 0.005 blocking, and five routes were allowed for each point-to-point demand pair in each hour.

3.4.1 Ten-node single hour results

The UA can design a network for a single hour simply by assigning all the traffic for a particular point-to-point pair to the least expensive route for that pair. There is no need to generate more than one route for each point-to-point pair since the direct route is the least expensive in the single-hour case.

Table I gives single-hour network design results using the 10 a.m.

load data, together with the percent savings for progressive routing, multilink routing, and two-link routing in comparison to the network engineered for hierarchical routing. The average network point-to-point GOS is also shown for each network design. The UA design cost usually converged in about five iterations. The savings for progressive routing and two-link routing are only slightly smaller than multilink routing. The average network GOS for the DNHR networks were all better than the hierarchy.

The primary reasons that the UA can save about 6–7 percent over a hierarchical design appear to be that (I) the UA has a better choice of routing, and (II) all groups can be sized for an efficient blocking level. In the 10-node network, the algorithm used paths from Los Angeles, Ca. to Orlando, Fl. that passed through Birmingham, Al. and Phoenix, Az., along with the more normal paths through San Bernardino, Ca. and Rockdale, Ga. used by the hierarchical design. Additionally, no existing final groups were sized for one percent blocking, hence, the average trunk occupancy was higher. For example, the Rockdale to White Plains, N.Y. group was sized for 16 percent blocking by the UA, and paths through the subtending sectional centers (in the hierarchy) were used to carry traffic overflowing the Rockdale–White Plains group so that the overall point-to-point blocking objective was met. In fact, the average blocking on groups that would be interregional finals in a hierarchy was about 21 percent in the UA. This resulted in higher occupancy of these expensive interregional groups.

3.4.2 Ten-node multihour results

We now discuss hierarchical, progressive, multilink, and two-link networks to satisfy the traffic loads for three network busy hours (10 a.m., 1 p.m., and 8 p.m.). From the results in Table II we conclude that there is little difference in potential network cost savings between progressive routing, two-link routing, and multilink routing. In fact, it appears that using CCIS crankback and originating node control will only save about an additional one percent in network cost. The reason is that most traffic in the various dynamic routing networks is routed on the same links, because for many point-to-point pairs, these routing

Table I—Single-hour unified algorithm results for 10-node network (10 a.m. load)

Network Routing	Cost	GOS	Savings (%)
Hierarchical	$5,949,500	0.009	
Progressive	5,567,800	0.004	6.4
Multilink	5,511,100	0.005	7.4
Two-link	5,555,900	0.005	6.6

Table II—Network designs for 10-node network (based on three hours)

Network Routing	Cost	Savings (%)	Network (GOS)	Hour
Hierarchical	$7,160,000			
Progressive	6,043,100	15.6	0.003	10 a.m.
			0.002	1 p.m.
			0.003	8 p.m.
Multilink	5,980,100	16.5	0.002	10 a.m.
			0.001	1 p.m.
			0.002	8 p.m.
Two-link	6,064,300	15.3	0.003	10 a.m.
			0.002	1 p.m.
			0.003	8 p.m.

methods carry a significant amount of traffic on the direct path and on the same two-link, first-alternate path.

Because progressive routing, two-link routing, and multilink routing designs are very close in cost, the preferred routing method should be based on ease of implementation. Progressive routing requires a history of visited nodes to be sent with each call to prevent looping. Since no central point has complete control of a particular call, it would also be quite difficult to measure point-to-point blocking. We contrast this to the use of originating node control in multilink or two-link routing which makes it easier to measure point-to-point blocking. The blocking measurement is necessary for network servicing in order to adjust routing and augment trunk groups to satisfy unforeseen loads; the blocking measurement would indicate when corrective action is necessary. Having originating node control of every call is also helpful for real-time routing, which attempts to maximize use of the network in the face of unusual load conditions. For a description of servicing and real-time routing, see Ref. 2. On the basis of these implementation considerations and the comparable savings, two-link routing appears to be the preferred routing method.

3.5 Path formulation

As explained earlier, the route formulation decided on the possible routes a call may take prior to the LP assigning traffic to the candidate routes at minimum cost. The choice of routes was limited because of the large number of candidates. For example, the number of routes that can be formed from ten paths is 10!, or over 3 million routes. Hence, the restricted choice of routes could result in suboptimality, since a better route not contained in those generated may exist.

The path formulation forms routes after the optimization step. Hence, the LP and "form routes" blocks would be interchanged in Fig. 7. The LP assigns traffic directly to the candidate paths at minimum cost.

The first step in the router stage is to generate the required number of one- and two-link paths. These paths are then passed to the LP, which is somewhat different in structure than that of the route formulation. This difference arises since the amount of flow that can be carried on a particular path depends on the blocking on that path and on the flow assigned to all other paths comprising the particular route. For instance, if the blocking on a path were 20 percent and the offered load were 100 erlangs, it would be impossible to carry more than 80 erlangs on this path. Hence, some method is needed to determine upper limits on path flow so that the resulting flows selected by the LP are feasible. Such questions of feasibility did not arise in the route formulation since the link blocking probabilities were embedded in the link proportions.

3.5.1 Flow feasibility algorithm

An iterative method of using upper bounds to force flow feasibility is shown in Fig. 10. Here we incorporate flow feasibility constraints into the router stage. Immediately after the generation of paths, initial upper bounds on path flows are set for use by the first LP iteration. At this point, nothing is known about the amount of flow which is optimal on any path. Hence, we desire to constrain the LP as little as possible. For this reason, the initial upper bound on flow on any path j for demand pair k is set according to the following formula:

$$UPBD_{jk} = R_k(1 - B_{jk}),$$

where

$UPBD_{jk}$ = upper bound on flow on path j of demand pair k,
R_k = offered load to demand pair k,
B_{jk} = blocking on path j of demand pair k.

(The dependence of these quantities on the hour has been suppressed for clarity.)

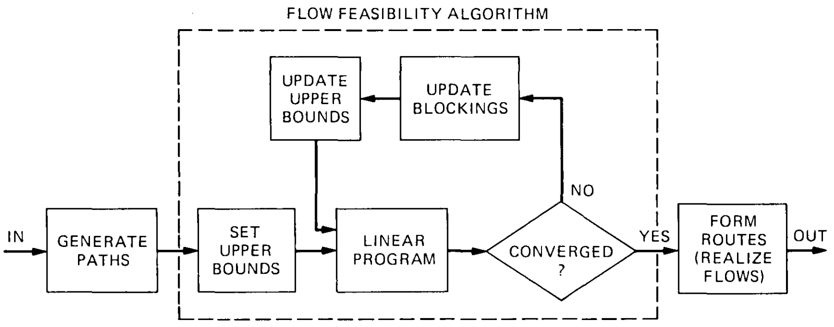


Fig. 10—Unified algorithm path formulation router detail.

Hence, the initial upper bound on flow is set, assuming that the entire offered load can be offered to any path independently of the load offered to any other path. Thus, the resulting flows can be infeasible since there might not be enough offered load to simultaneously achieve the desired flow on all paths for the same demand pair. For instance, suppose

$$\begin{aligned} B_{1k} &= 0.2, \\ B_{2k} &= 0.1, \\ B_{3k} &= 0.2, \\ R_k &= 10 \text{ erlangs}, \end{aligned}$$

Then

$$\begin{aligned} UPBD_{1k} &= 8 \text{ erlangs}, \\ UPBD_{2k} &= 9 \text{ erlangs}, \\ UPDB_{3k} &= 8 \text{ erlangs}. \end{aligned}$$

We assume that the required GOS is 0.005 so that the flow on all three paths should total 9.95 erlangs; this required flow is feasible since an overall blocking of $B_{1k}B_{2k}B_{3k} = 0.004$ is possible should all paths be used. Now suppose that the LP chooses for this demand pair the optimal flows

$$\begin{aligned} r_{1k} &= 8 \text{ erlangs}, \\ r_{2k} &= 1.95 \text{ erlangs}, \\ r_{3k} &= 0, \end{aligned}$$

where r_{ik} is now redefined as the carried flow on path i of demand pair k. The only way to realize the desired flow of 8 erlangs on path 1 is to offer path 1 the entire 10 erlangs. This means that 2 erlangs will overflow path 1. These 2 erlangs can then be offered to path 2, but can result in a maximum flow of 1.8 erlangs due to the blocking on path 2. Hence, the desired flows are infeasible. A method to compute new upper bounds to force these flows toward a more feasible solution will be discussed shortly; attention will now be focused on the structure of the LP used with the path formulation.

An LP to optimize path flows will solve the following problem:

minimize

$$\sum_{i=1}^{L} M_i a_i$$

subject to

$$\sum_{k=1}^{K} \sum_{j=1}^{J_k^h} P_{jk}^{ih} r_{jk}^h \le a_i \qquad i = 1, 2, \cdots, L$$

$$h = 1, 2, \cdots, H$$

$$\sum_{j=1}^{J_k^h} r_{jk}^h = G_k^h \qquad \begin{array}{l} h = 1, 2, \cdots, H \\ k = 1, 2, \cdots, K \end{array}$$

$$r_{jk}^h \le UPBD_{jk}^h \qquad \begin{array}{l} h = 1, 2, \cdots, H \\ k = 1, 2, \cdots, K \\ j = 1, 2, \cdots, J_k^h \end{array}$$

$$r_{jk}^h \ge 0, \qquad a_i \ge 0,$$

where we redefine

P_{jk}^{ih} = 1 if path j for demand pair k uses link i in hour h,
= 0, otherwise,
r_{jk}^h = carried load on path j for demand pair k in hour h,
J_k^h = number of paths for demand pair k in hour h,
G_k^h = total carried load for demand pair k in hour h.

The total carried load for demand pair k in hour h is related to the total offered load for demand pair k in hour h, as follows. The minimum blocking that can be achieved on demand pair k is

$$E_k^h = \prod_{j=1}^{J_k^h} B_{jk}^h,$$

where B_{jk}^h = blocking on path j for demand pair k in hour h. Let

$$\text{GOS} = \text{desired grade-of-service}$$

and the blocking on demand pair k in hour h will be

$$f_k^h = \max[E_k^h, \text{GOS}].$$

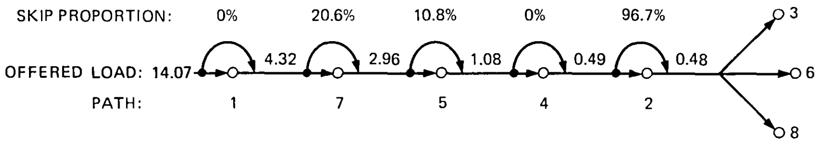
Then,

$$G_k^h = R_k^h[1 - f_k^h].$$

Thus, the total carried flow is determined by the GOS, unless E_k^h is greater than this desired GOS. If the GOS constraint cannot be met, all paths are required to be at their maximum flow to minimize the blocking. A blocking correction algorithm, similar to that used in the route formulation, is used in the engineering stage to correct those routes whose blockings are unacceptable.

Returning to Fig. 10, the next step in the flow feasibility algorithm is to update the link blockings in all hours based on the current link flow. This can be done by calculating the link size so that the maximum allowed blocking in any hour is not exceeded, and then calculating the blocking in all hours. After the blockings have been updated, the upper bounds need to be recalculated based on the current desired flows (determined by the LP), so as to obtain a more feasible solution.

The method used to recalculate the upper bounds is best illustrated by an example. The data in Fig. 11 show how a routing method, called



1	2	3	4	5	6	7	8
		LP	LP	REALIZED	REALIZED		
PATH	PATH	CARRIED	OFFERED	OFFERED	CARRIED	UPPER	
NUMBER	BLOCKING	LOAD	LOAD	LOAD	LOAD	BOUND	VIOLATION
1	0.307	9.75	14.07	14.07	9.75	9.75	0
7	0.603	1.36	3.43	3.43	1.36	1.72	0
5	0.287	1.88	2.64	2.64	1.88	2.11	0
4	0.453	1.06	1.94	1.08	0.59	0.59	0.47
2	0.239	0.012	0.016	0.016	0.012	0.37	0
3	0.309	0	0	0.48*	0	0.33	0
6	0.339	0	0	0.48*	0	0.32	0
8	0.488	0	0	0.48*	0	0.25	0

*OFFERED LOAD ASSIGNED FOR UPPER BOUND CALCULATION AS DESCRIBED IN THE TEXT.
EXCEPT FOR PATH NUMBER AND PATH BLOCKING, ALL ENTRIES ARE IN ERLANGS.

Fig. 11—Upper bound determination using a skip-one-path algorithm.

skip-one-path routing, can be used to set upper bounds which force more feasible flows, while still allowing the LP some flexibility in choosing new flow patterns. (The algorithm is called skip-one-path because traffic is allowed to skip a path where it is not needed.) Basically, this algorithm works by keeping track of the offered load available and using this load to realize the desired flows in a sequential manner. The data in Fig. 11 were the flows selected by an LP that used the initial upper bounds to route 14.07 erlangs of load on a particular demand pair in a particular hour.

The first step in the algorithm is to calculate the load which must be offered to each path to realize the flow selected by the LP. This offered load, given in the fourth column of Fig. 11, has been calculated from the carried load on each path selected by the LP (column 3) divided by one minus the blocking on the path (column 2). The next step is to sort the path offered loads from largest to smallest. This has been done for the data in Fig. 11; note that path number 7 follows path number 1 in terms of offered load. The path numbers used here refer to an internal ordering used by the algorithm.

Once the path ordering has been determined, the algorithm proceeds as follows. As the largest offered load desired by the LP is equal to the total offered load of 14.07 erlangs, all the load must be offered to path 1 as shown in the diagram in Fig. 11. Hence, none of the offered load is "skipped over" path 1. Applying the load in this way will realize the desired flow on the first path. Note that path 1, which carries the greatest flow, is at its upper limit (a common situation). With the given blocking of path 1, the overflow from path 1 is 4.32 erlangs.

Thus, the offered and carried loads desired on path 1 can be achieved, as shown in columns 5 and 6. Since the total demand load is available for path 1, and the blocking is assumed constant for this example, the upper bound on path 1 flow remains constant. The last column in Fig. 11 gives the violation, or amount by which the desired flow exceeds the new upper bound. In this case, the violation is zero.

Now consider path 7, which is next in order of offered load. The desired offered load to this path is 3.43 which is less than the overflow from path 1. The difference between these two loads, which is 4.32 − 3.43 = 0.89 (20.6 percent of 4.32), is skipped over path 7, and 3.43 erlangs is applied to path 7. This process of skipping can be accomplished by generating a random number before the call is offered to path 7. With probability 0.206, a call skips path 7 and is offered to the next path. A call that does not skip is offered to path 7.

Thus, the desired flow on path 7 can be realized. The upper bound on path 7 is calculated assuming the entire offered load (4.32 erlangs) could be offered to path 7. Note that this allows for more flow on path 7, if desirable, on the next iteration of the LP. The skip-one-path algorithm gives an actual offered load to path 7 of 3.43 erlangs with 2.07 erlangs overflow. The overflow is calculated as (0.603) (3.43) = 2.07. The total available load for any other path is now 2.07 + 0.89 = 2.96 erlangs.

Now consider path 5 which needs 2.64 erlangs of offered load. The amount of traffic to be skipped is 2.96 − 2.64 = 0.32, or 10.8 percent of the 2.96 erlangs available. The upper bound on the path 5 flow is based on 2.96 erlangs, which is the total available load at present that could be offered to path 5.

A different situation arises, however, when attempting to realize the desired offered load to path 4 of 1.94 erlangs. The total of the overflow from path 5 (0.76 erlangs) and the 0.32 erlangs skipped over path 5 is 1.08 erlangs which is the maximum load that can be offered to path 4. Hence, the LP has assigned more flow than can be realized. The maximum possible flow is 0.59; likewise, the upper bound is 0.59. Thus, there is a bound violation of 1.06 − 0.59 = 0.47 erlangs.

The process continues until the last path with a nonzero LP flow has been dealt with. At this point, all unused load, 0.48 erlangs in this example, is assumed to be available as offered load for all paths with a zero flow assigned by the LP. The upper bounds are then calculated in the same way as the initial upper bounds were set.

As mentioned earlier, the algorithm is called skip-one-path because traffic is allowed to skip a path where it is not needed. Actually, given that the amount of load to be skipped can be realized by generating random numbers, this algorithm yields a workable routing method to realize the desired LP flows, as will be discussed in Section 3.5.4.

Once the upper bounds have been calculated, the LP can be again executed to optimize the new problem. The sum of bound violations is available as a measure of flow feasibility. It is not necessary to begin the LP from "scratch" since the current routing patterns, with upper bounds updated to reflect the new flows, can be used as a starting basis.

3.5.2 Flow realization techniques

Once the path LP has converged, we must then realize the LP flows by forming the appropriate routes. The flow realization algorithm selects the routes. Three flow realization algorithms are discussed here and differ in their computational complexity and their flexibility in approximating the desired flows. Each algorithm treats the desired flows in each design hour independently, hence, the routing changes from hour to hour.

3.5.3 Routing algorithm-CGH

The CGH algorithm, named after Chung, Graham, and Hwang, who developed it, is composed of cyclic blocks. For example, suppose there are seven paths with desired flows r_i. One possible cyclic block realization of the seven r_i is

$$(1)\ (2\ 3\ 4)\ (5\ 6)\ (7).$$

The notation means that all the offered load to this route is first offered to path 1. The overflow from path 1 is then offered to a cyclic block composed of paths 2, 3, and 4. The term cyclic block means that a proportion β_i^k of the total load offered to the kth block is offered to cyclic permutation i, where cyclic permutation i is selected so that the ordering within the block is preserved but a different path appears first. In the cyclic block under consideration, a proportion β_1^2 of the input traffic will be offered to the paths in the order (2, 3, 4) and proportion β_2^2 to (3, 4, 2), etc. Offering traffic in this manner may be accomplished by generating a random number when a call is offered to the cyclic block. Note that all calls see the same blocking probability within the cyclic block since all paths are searched.

The realization algorithm must define the contents of each cyclic block and calculate the proportions β_i^k associated with the kth cyclic block. The basic steps to accomplish this are as follows, and the subsequent example should make the steps clear.

In the interest of brevity, notation dealing with demand pairs and design hours has been suppressed. Let

r_i = desired flow on path i,
B_i = blocking associated with path i,
$Q_i = 1 - B_i$ = connectivity of path i,

$\sigma_i = \dfrac{r_i}{Q_i}$ = desired offered load to path i,

$\delta_i = B_i\sigma_i$ = desired overflow load from path i,

J = total number of paths.

(*i*) Calculate σ_i and δ_i. Sort and relabel the σ_i, if necessary, so that

$$\sigma_1 \geq \sigma_2 \geq \sigma_3 \cdots \geq \sigma_J.$$

(*ii*) The first path in the cyclic block to be formed is the as yet unused path i with the largest σ_i.

(*iii*) Insert an as yet unused path i with largest remaining σ_i after a path j with $\sigma_i > \delta_j$, if such a path j exists. Repeat this step until no such j exists.

(*iv*) The current (kth) cyclic block ends with the last path inserted. If there is but one path in the block, set its coefficient to 1.0 and go to (*v*). If there is more than one path in the block, let

$$\alpha_l^k = \sigma_{m(l)} - \delta_j,$$

where $m(l)$ refers to the path in the lth position in the kth block, and j refers to the path preceding it in the cyclic ordering of the block. Note that the algorithm guarantees that all α_l^k are positive. Then calculate

$$\beta_i^k = \frac{\alpha_i^k}{\sum_{i=1}^{L} \alpha_i^k},$$

which is the cyclic coefficient associated with the ith path in the kth block, assuming there are L paths in the block.

(*v*) If there are remaining $r_i > 0$, return to (*ii*).

(*vi*) Add single path cyclic blocks at the end of the route, if necessary, until the GOS constraint is satisfied.

Table III shows an example of the algorithm and the resulting

Table III—The CGH routing example

Path Number	Path Blocking	LP Carried Load	LP Offered Load	LP Overflow Load	Realized Carried Load		Error
	(B_i)	(r_i)	(σ_i)	(δ_i)			
1	0.307	9.75	14.07	4.32	9.75		0
7	0.603	1.36	3.43	2.06	1.26		0.10
5	0.287	1.88	2.64	0.76	1.74		0.14
4	0.453	1.06	1.94	0.88	0.98		0.08
2	0.239	0.012	0.016	0.0037	0.26		0.25
3	0.309	0	0	0	0.06		0.06
6	0.339	0	0	0	0		0
8	0.488	0	0	0	0		0
						Total:	0.63

Route: (1) (7, 5, 4) (2) (3)
Coefficients: (100%) (59.2%, 13.4%, 27.4%) (100%) (100%)
Except for path number and path blocking, all entries are in erlangs.

routing. The data for this example is identical to that used for the example in Fig. 11. The path order in Table III has already been sorted on offered load (σ_i). Path 1 becomes the first path in the first cyclic block since σ_1 is the largest; it is also the only path in the first cyclic block since no other σ_i is larger than δ_1.

Path 7 begins the next cyclic block, since σ_7 is the largest remaining offered load. Path 5 follows path 7 in the second cyclic block since σ_5 (2.64) is greater than δ_7 (2.06). Likewise, path 4 follows path 5. The second block ends with path 4, since no other unused path has an offered load greater than δ_4.

The coefficients of the second cyclic block are computed as follows:

$$\begin{aligned}\alpha_1^2 &= \sigma_7 - \delta_4 = 3.43 - 0.88 = 2.55\\ \alpha_2^2 &= \sigma_5 - \delta_7 = 2.64 - 2.06 = 0.58\\ \alpha_3^2 &= \sigma_4 - \delta_5 = 1.94 - 0.76 = \underline{1.18}\\ \text{Total} & \qquad\qquad\qquad\qquad\quad = 4.31\end{aligned}$$

Then,

$$\beta_1^2 = \frac{2.55}{4.31} = 59.2 \text{ percent}$$
$$\beta_2^2 = \frac{0.58}{4.31} = 13.4 \text{ percent}$$
$$\beta_3^2 = \frac{1.18}{4.31} = 27.4 \text{ percent}$$

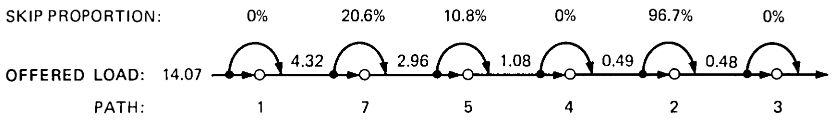
These coefficients β_i^k for the four blocks are shown below the route shown in Table III in order of starting path; thus, 59.2 percent of the traffic offered to the second block starts with path 7.

Path 2 forms a one-member cyclic block since it is the only path left with a positive σ. Note that path 3 was included to decrease the blocking from 0.0057 to 0.0017, thus, meeting the GOS objective of 0.005. The total path flow error (absolute difference between desired flow and realized flow) is shown to be 0.63 erlangs.

3.5.4 Skip-one-path algorithm

The skip-one-path algorithm can be used to realize path flows, as well as to calculate upper bounds. An example of skip-one-path routing is shown in Fig. 12. A No. 4 ESS routing data block could be modified to do skip-one-path routing by generating a random number before a call is offered to the next path. With a predetermined probability, a call would skip over the path without being offered to it and proceed to the next path in the routing sequence.

Once again, the first step is to sort the paths by offered load. The algorithm used to calculate the amount of offered traffic to skip the next path was discussed previously. Note that path 3 has been added



PATH NUMBER	PATH BLOCKING	LP CARRIED LOAD	LP OFFERED LOAD	REALIZED OFFERED LOAD	REALIZED CARRIED LOAD	|ERROR|
1	0.307	9.75	14.07	14.07	9.75	0
7	0.603	1.36	3.43	3.43	1.36	0
5	0.287	1.88	2.64	2.64	1.88	0
4	0.453	1.06	1.94	1.08	0.59	0.47
2	0.239	0.012	0.016	0.016	0.012	0
3	0.309	0	0	0.48	0.33	0.33
6	0.339	0	0	0	0	0
8	0.488	0	0	0	0	0
					TOTAL:	0.801

EXCEPT FOR PATH NUMBER AND PATH BLOCKINGS, ALL ENTRIES ARE IN ERLANGS

Fig. 12—Skip-one-path routing example.

to meet a GOS objective of 0.005. Also, Fig. 12 shows a path flow error of 0.80, which is greater than the 0.63 path flow error given by the CGH algorithm.

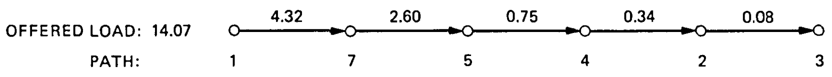
3.5.5 Sequential routing algorithm

A very simple method to realize desired path flows is termed sequential routing. This scheme simply sorts the desired flows on offered load (as do the other methods) and lets the first path overflow to the second path which overflows to the third path, and so on. Thus, traffic is routed sequentially from path to path with no probabilistic methods being used to get the realized flows closer to the desired flows. The reason that sequential routing works well is that most flow is carried on the first one or two paths, which are loaded to their upper bound, and errors in meeting flow on later paths are not significant.

Figure 13 shows a sequential routing example. The given blockings and desired flows are identical to those used in the other routing examples. Note that in this particular example, sequential routing has the highest error in flows of all the three routings studied. In general, sequential routing has the least flexibility of the three realization methods discussed here. We consider the effect of this flow inaccuracy on network cost.

3.6 Path formulation results and conclusions

Network designs were obtained for the CGH algorithm and the sequential routing algorithms using a 30-node network model. The results in Table IV show that the CGH algorithm is more accurate than



PATH NUMBER	PATH BLOCKING	LP CARRIED LOAD	LP OFFERED LOAD	REALIZED OFFERED LOAD	REALIZED CARRIED LOAD	|ERROR|
1	0.307	9.75	14.07	14.07	9.75	0
7	0.603	1.36	3.43	4.32	1.72	0.35
5	0.287	1.88	2.63	2.60	1.86	0.02
4	0.453	1.06	1.94	0.75	0.41	0.65
2	0.239	0.012	0.016	0.34	0.26	0.25
3	0.309	0	0	0.08	0.06	0.05
6	0.339	0	0	0	0	0
8	0.488	0	0	0	0	0
					TOTAL:	1.32

EXCEPT FOR PATH NUMBER AND BLOCKING, ALL ENTRIES ARE IN ERLANGS

Fig. 13—Sequential routing example.

the sequential algorithm, but the difference in final network cost was only about 0.5 percent. Additionally, these routing methods added between 1 and 2 percentage points to the savings achieved with the route formulation. While these results have been illustrated only for small network models, they have recently been confirmed using a full-scale, 215-node, intercity network model.

Hence, among path formulation routing alternatives, the very simple sequential routing technique achieves network design savings almost identical to those of much more complicated schemes. A routing method, such as CGH has additional costs not quantified in this study. For instance, a switching system with CGH routing would have to store traffic allocation proportions and markers to indicate where the cyclic blocks begin and end, along with the ordered list of paths. Sequential routing, on the other hand, needs only the ordered list of paths. Also, applying traffic allocation techniques such as those needed by CGH routing would require real time to generate and process the appropriate random numbers. Sequential routing needs no such traffic allocation and, hence, has a real-time advantage.

Table IV—Network designs for 30-node network (based on 16 hours)

Network Routing	Cost	Savings (%)
Hierarchical	$137,874,300	
Two-link (route formulation)	117,830,000	14.5
Sequential	115,534,300	16.2
CGH	114,849,300	16.7

IV. OPTIMIZATION METHODS AND RESULTS

4.1 Heuristic optimization method (HOM)

As mentioned in Section 3.2, an HOM was developed to solve the LP problems of the UA route formulation. This heuristic was revised to solve the LP problems of the path formulation (Section 3.4). For the sake of brevity, this section describes only the latter version. We discuss the three basic ideas underlying the HOM and provide a brief overview.

4.1.1 Rerouting of traffic

The first concept concerns the rerouting of traffic. A reroute is a reassignment of flow of a particular point-to-point pair from one path to another in a single design hour. Given an initial assignment of path flows for each point-to-point pair, the HOM progresses to its final solution by a sequence of reroutes. Thus, each iteration of the heuristic affects the flow on a few links and in only one design hour.

4.1.2 Marginal costs

Next, we discuss a concept which allows us to evaluate the potential cost savings of any reroute. The marginal link cost is an estimate for the rate of change of the total network cost function relative to the change in flow on a link during a particular design hour. The HOM uses an UPCOST and a DOWNCOST indicating the predicted cost change if we increase or decrease the link flow during a particular hour. We maintain marginal costs for every link during every design hour.

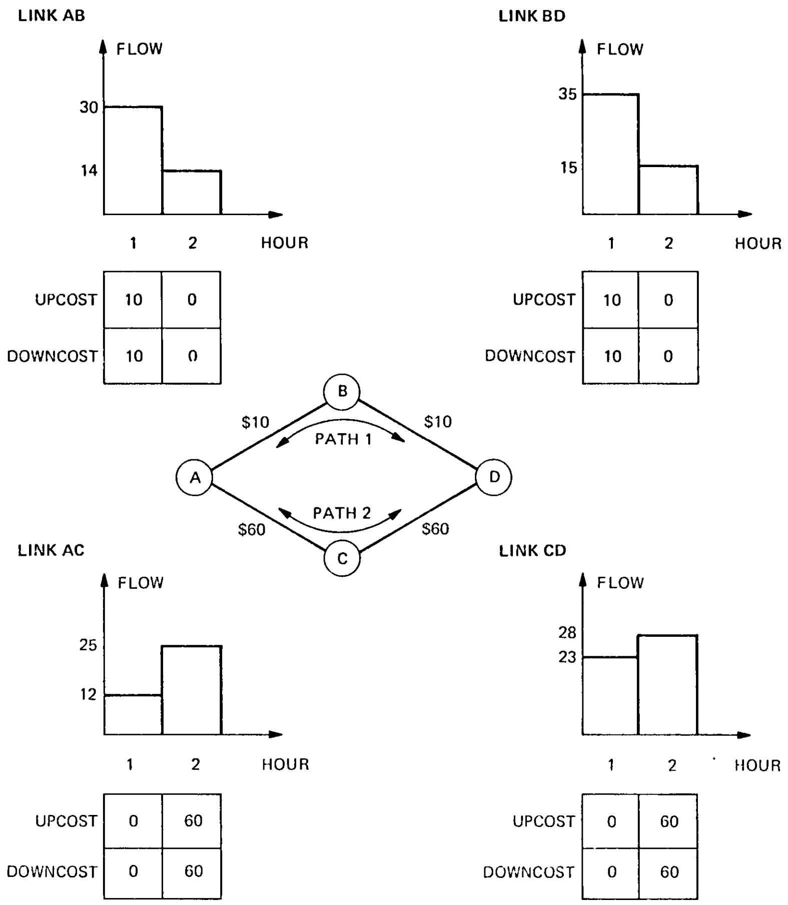
The rules for determining the marginal link costs of a link are simple. For a particular link we examine the flow during each design hour. If the peak flow on the link occurs in only one hour, then increasing or decreasing the flow in that hour will increase or decrease the capacity of the link. We then set the UPCOST and DOWNCOST in the peak hour equal to the metric of the link. If the peak flow occurs in more than one hour, then increasing the flow in one of the peak hours will increase the link capacity, while decreasing the flow in one of the peak hours will leave the capacity unchanged. We then set the UPCOST in all peak hours equal to the link metric and set the DOWNCOST in all peak hours equal to zero. In all design hours where the link flow is below the peak flow, we set the UPCOST and DOWNCOST equal to zero since increasing or decreasing the flow does not affect the link capacity.

Once the marginal link costs are computed, we can determine the marginal cost of diverting flow from one path to another in the following way. We first sum the UPCOSTs of the path that will gain flow, and then sum the DOWNCOSTs of the path that will lose flow.

Subtracting the latter sum from the former sum yields the marginal cost of the reroute. If this cost is negative, then the reroute is profitable.

Once we decide to perform a particular reroute whose marginal cost indicates that it is profitable, we then determine the amount of flow to divert. The rule for finding this quantity is to continue rerouting until the marginal cost of the reroute changes.

Figures 14 and 15 sketch an example of how the marginal link costs are determined and how they are modified when a rerouting of traffic occurs. Figure 14 shows two paths between nodes A and D in the first of two design hours. Each path has two links. The metrics for links AB, BD, AC, and CD are 10, 10, 60, and 60, respectively. Path 1 is initially assigned 20 erlangs of flow while path 2 is assigned none. The upper bounds on the path flows are 20 and 15 erlangs, respectively.



Link AB	Hour 1	Hour 2
UPCOST	10	0
DOWNCOST	10	0

Link BD	Hour 1	Hour 2
UPCOST	10	0
DOWNCOST	10	0

Link AC	Hour 1	Hour 2
UPCOST	0	60
DOWNCOST	0	60

Link CD	Hour 1	Hour 2
UPCOST	0	60
DOWNCOST	0	60

Fig. 14—Link flows and marginal link costs in example.

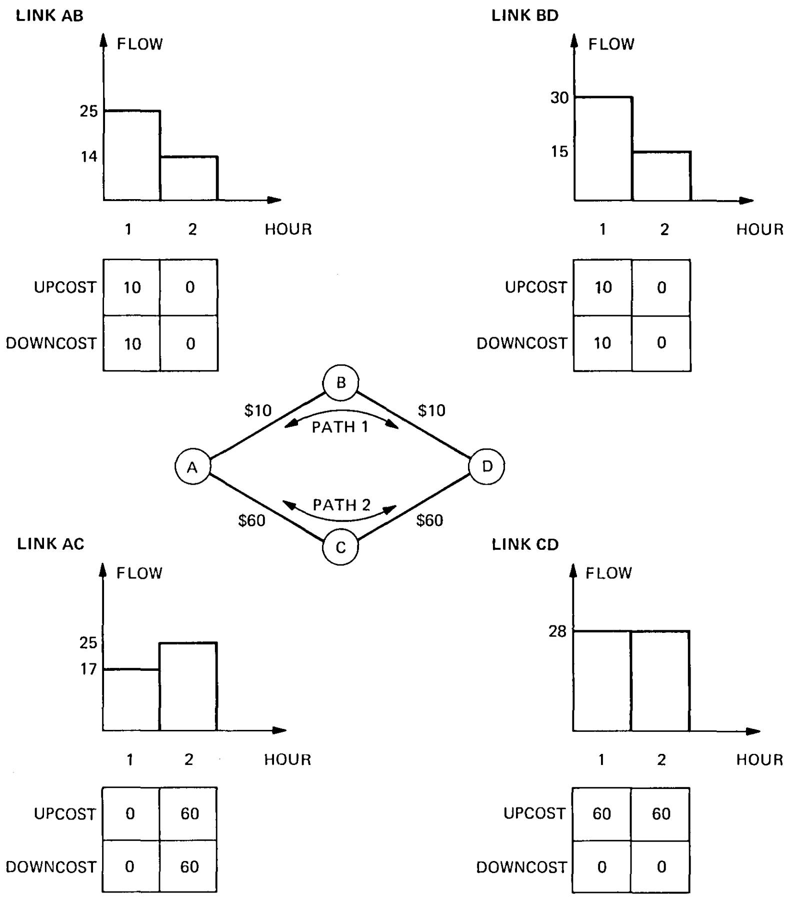


Fig. 15—Updated link flows and marginal link costs in example.

Figure 14 also shows the initial flow and marginal costs for each of the links in each of two design hours. For example, link AB carries 30 erlangs in hour 1 and only 14 erlangs in hour 2. Since it has a unique peak in hour 1, the UPCOST and DOWNCOST of link AB in hour 1 are set equal to 10, the metric of link AB. In hour 2, they are set equal to zero.

We can now use the marginal link costs to determine the marginal cost for rerouting traffic from path 1 to path 2. Summing the UPCOSTs of links AC and CD in hour 1, and subtracting from it the sum of the DOWNCOSTs of links AB and BD in hour 1 yields

$$(0 + 0) - (10 + 10) = -20,$$

the marginal cost of diverting from path 1 to path 2 in hour 1. Therefore, the reroute is a profitable one.

We next determine the amount of flow to divert. Now the marginal profit of the reroute will hold until one of the marginal link costs in the above calculation changes. We then reroute as much flow as possible until either the DOWNCOST of link AB, the DOWNCOST of link BD, the UPCOST of link AC, or the UPCOST of link CD changes in hour 1. Figure 15 describes the effect of rerouting 5 erlangs of flow in hour 1. Link CD has gained 5 erlangs of flow and now has 2 peak hours. The UPCOST in each hour is now equal to the link metric, while the DOWNCOST in each hour is zero. Since the UPCOST of link CD has changed from zero to sixty, 5 erlangs is the total amount of flow that we reroute. If after the marginal reroute cost is reevaluated we find that the reroute is still profitable, we continue to divert flow until the marginal reroute cost changes again. We continue in this manner until either the reroute is no longer profitable, or there is no more flow assigned to path 1, or the flow on path 2 reaches its upper bound. We then search for another profitable reroute.

4.1.3 Candidate list

The last concept for the HOM concerns the method for deciding how many candidate reroutes to evaluate before actually performing a particular reroute. In selecting a reroute pair, there is a tradeoff between the quality of the reroute found and the amount of time spent searching for it. Although we would like to find very profitable reroutes, the HOM should also be computationally efficient. The HOM uses a candidate list to find the next reroute to perform. This concept works in the following way. The first M point-to-point pairs are searched for profitable reroutes. The K most profitable reroutes are put into a candidate list and the most profitable reroute in the list is selected and performed. Once this particular reroute is no longer profitable, the remaining members of the list are reevaluated and the most profitable reroute is selected and performed. This process continues until there are no more profitable reroutes left in the list. The next M point-to-point pairs are then searched for profitable reroutes, a new list is generated, and the reroutes in the list are performed until they are no longer profitable. The HOM continues in this manner. Whenever the last point-to-point pair in the set of all point-to-point pairs is encountered, the next point-to-point pair to be considered is the first point-to-point pair in the set. The heuristic then "wraps around" the set of all point-to-point pairs. The HOM finally terminates when there are no profitable reroutes among all point-to-point pairs.

4.1.4 Overview

The three concepts we have described in this section (the rerouting of traffic, the marginal costs, and the candidate list) are used together in the HOM. After an initial feasible solution is selected, the marginal link costs are determined. A group of point-to-point pairs is searched and a list of the most profitable reroutes is formed. Each reroute is performed until it is no longer profitable. When there are no more profitable reroutes in the list, a new group of point-to-point pairs is searched and a new list is formed. The heuristic continues in this manner until there are no profitable reroutes. The next section describes typical examples of the computational results that have been obtained.

4.2 Computational results

We compared the HOM with MPSX/370 using an LP problem that was generated by the UA. The problem was derived from the 28-node network where the average number of paths per point-to-point pair was 9.4. The corresponding LP had 1402 rows, 7630 columns, and a density of 0.17 percent. We used as a reference point a nonoptimal solution obtained by MPSX/370 after 904 CPU seconds. The heuristic progressed very rapidly until it was within 0.2 percent of the MPSX/370 solution. It then terminated after only 2 CPU seconds. In contrast, MPSX/370 required 860 CPU seconds to produce a solution of similar quality. We see that the HOM can produce a near-optimal solution much more quickly than MPSX/370. Also, the UA contains many approximations so that an optimal solution to the LP is not necessary. In fact, there is little penalty in the network cost if the HOM is used.

4.3 Run times for large networks

To design a 200-node network with 6 design hours, the UA will require about 20 million bytes of memory, much of which is needed to solve the linear programs.

Tests with a 190-node intercity network model for 6 design hours indicate that the UA will require less than 4 hours of CPU time to design a large network. Half of this time will be spent by the HOM. With expected advances in hardware, the total time may become considerably smaller.

V. POTENTIAL BELL SYSTEM APPLICATIONS

The implementation of DNHR requires the following developments:

(*i*) Network design, servicing, and electronic switching system software.

(*ii*) Collection of point-to-point data. The network would be designed and administered using a point-to-point blocking criterion and the blocking must be measured to administer the network properly.

(*iii*) Mechanization of routing administration function. The proliferation of routing updates would necessitate a mechanized routing administration system.

(*iv*) Design along a network boundary separating the centralized intercity network from the decentralized metropolitan networks. This is to achieve efficient network designs, both in cost and in computing time.

(*v*) Modification of network operation support systems such as the network management systems.

(*vi*) Modifications of switch planning tools and methods. These must be modified to reflect DNHR design in order to model the network properly.

The expected benefits of dynamic routing are attractive. However, the fundamental nature of the changes raises service, cost, and feasibility issues that might substantially reduce the projected benefits.

Several studies are underway to assess fundamental issues, such as network management, switching and signaling loads, large scale optimization, and transmission performance.

VI. ACKNOWLEDGMENTS

R. T. Wong originally conceived and implemented the route formulation heuristic method and has made substantial contributions to many other aspects of this work. We also acknowledge the contribution of A. H. Westreich, who is responsible for the work on progressive routing. F. R. K. Chung, R. L. Graham, and F. K. Hwang were instrumental in developing the path formulation of the UA and developed the CGH routing algorithm. L. T. Nguyen has made substantial contributions to many of the algorithms discussed here. He also developed fast traffic algorithms which were used in the design of the very large networks reported here.

REFERENCES

1. F. R. K. Chung, R. L. Graham, and F. K. Hwang, Efficient Realization Techniques for Network Flow Patterns, B.S.T.J., this issue.
2. G. R. Ash, A. H. Kafker, and K. R. Krishnan, "Servicing and Real-Time Control of Networks with Dynamic Routing," B.S.T.J., this issue.
3. C. J. Truitt, "Traffic Engineering Techniques for Determining Trunk Requirements in Alternate Routed Networks," B.S.T.J., *33*, No. 2 (March 1954), pp. 277–302.
4. B. Yaged, Jr., "Long Range Planning for Communications Networks," Polytechnic Institute of Brooklyn Ph.D Thesis, 1971.
5. J. E. Knepley, "Minimum Cost Design for Circuit Switched Networks," Technical Note Numbers 36-73, Defense Communications Engineering Center, System Engineering Facility, Reston, Va., July, 1973.
6. W. Feller, *An Introduction to Probability Theory and Its Applications*, Third Edition, New York: John Wiley, 1968.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Servicing and Real-Time Control of Networks With Dynamic Routing

By G. R. ASH, A. H. KAFKER, and K. R. KRISHNAN

(Manuscript received April 3, 1981)

The design of a network for dynamic routing is made using the forecasted network loads. Load uncertainties arising from errors in the forecast and from daily variations in network load give rise to reserve or idle network capacity not immediately needed by current network demands. The reserve capacity can be reduced by the use of more flexible dynamic routing methods, which allow routing flexibility to help control network flow under load uncertainties. We illustrate techniques for changing network routing patterns in planned and demand servicing to counteract the effects of forecast errors. Included in the benefits are a reduction in both reserve capacity, estimated to be about 5 percent of network first cost, and in trunk rearrangements. We also present call-by-call simulation results for real-time routing enhancements to the basic routing algorithms. The real-time routing algorithms use dynamic trunk reservation techniques, and the simulation results illustrate the improvement in network efficiency and performance under normal daily load variations, network overloads, and network failures.

I. INTRODUCTION AND SUMMARY

Dynamic routing is a new routing system that uses nonhierarchical, time-variable routing patterns to minimize network cost, as opposed to present routing rules that are time-fixed. The term "dynamic" frequently suggests an extensive real-time search for the optimal routing patterns. Real-time, traffic-sensitive routing is indeed the limiting case of time-variable routing, but, as we will see, the degree of load uncertainty determines the needed extent for this "true dynamic routing."

A companion article describes algorithms for designing minimum cost traffic networks using dynamic routing.[1] These design procedures were investigated under idealized conditions of perfectly known loads: the effects of errors in predicting the loads and other load uncertainties were ignored.

If the future loads on the network were completely known, then it would be sufficient to design the minimum-cost network to meet these loads—for example, by applying the procedure described in Ref. 1. In actuality, various categories of load uncertainty are present that necessitate a somewhat different strategy in building the network.

Network demands are continually growing and shifting which means we must forecast, design, and plan the required capacity far enough in advance (approximately 1 to 2 years) to meet the load. Of course, these forecasts are subject to error and the recognition of this error influences our planning strategy in various ways. The goal is to provide sufficient capacity to meet the expected load on the network. In planned servicing, the servicer plans the network based on the forecast loads and the trunks already in place. Consideration of the in-service trunks results in a disconnect policy that may leave capacity in place even though it is not called for by the design.

There are, however, economic and service implications of the planned servicing policy. Insufficient capacity means that occasionally trunks must be connected on short notice if the network load requires it. This process is called demand servicing. For many reasons it is desirable to minimize the level of demand servicing. There is a trade-off between reserve capacity and demand servicing which we explore in this paper. The algorithms described in Ref. 1 are enhanced to provide efficient planned servicing and demand servicing procedures. Using small network models, we find that these algorithms provide a potential 5 percent reduction in reserve capacity, while retaining a low level of demand servicing.

Uncertain variations in the instantaneous network loads also imply that capacity is never perfectly matched to the demand. Loads on the network shift from hour to hour and from day to day, and some amount of reserve capacity is almost always present. Hence, there is an opportunity to seek out this capacity in real time. We discuss a real-time routing algorithm that finds and uses idle network capacity to satisfy current loads. The procedure is a straightforward enhancement to the planned dynamic routing patterns, and small models predict that network blocking probability is reduced from about 0.0025 to 0.0006.

II. DYNAMIC ROUTING BACKGROUND

2.1 Routing method

The proposed routing method, illustrated in Fig. 1, is called two-link dynamic routing with crankback.

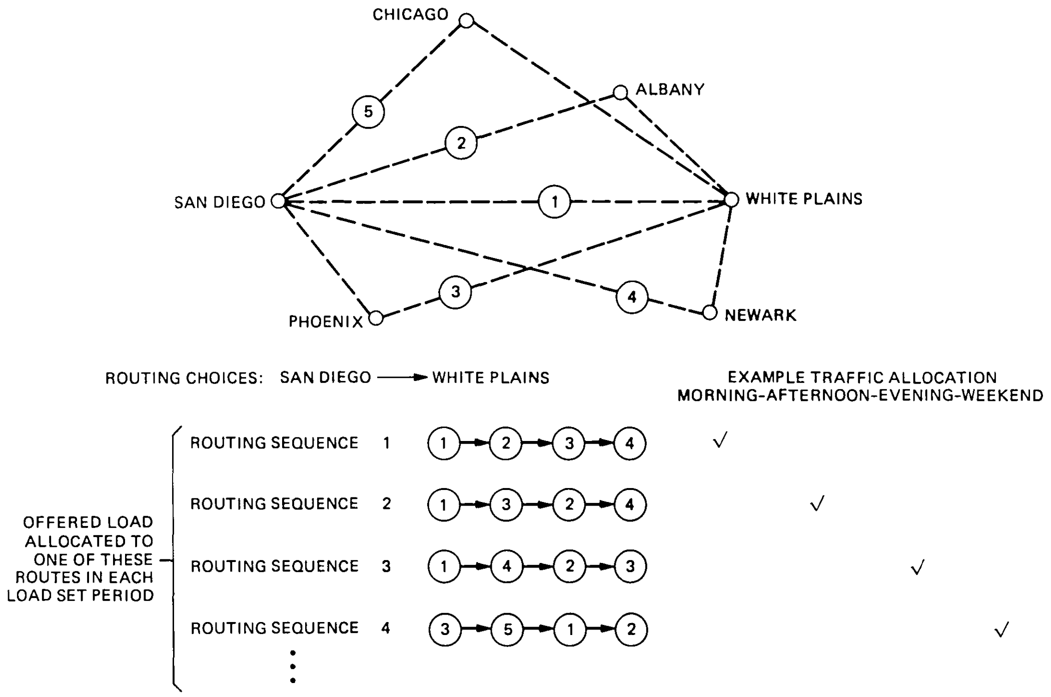

CHICAGO
ALBANY
SAN DIEGO
WHITE PLAINS
PHOENIX
NEWARK
5
2
1
3
4
ROUTING CHOICES: SAN DIEGO → WHITE PLAINS
EXAMPLE TRAFFIC ALLOCATION
MORNING-AFTERNOON-EVENING-WEEKEND
OFFERED LOAD ALLOCATED TO ONE OF THESE ROUTES IN EACH LOAD SET PERIOD
ROUTING SEQUENCE 1 1→2→3→4 ✓
ROUTING SEQUENCE 2 1→3→2→4 ✓
ROUTING SEQUENCE 3 1→4→2→3 ✓
ROUTING SEQUENCE 4 3→5→1→2 ✓
⋮

Fig. 1—Two-link dynamic routing with crankback.

The strategy was developed in the companion article and it capitalizes on two factors:

(*i*) Selection of minimum cost paths between the originating and terminating nodes, and

(*ii*) Designing optimal, time-varying routing patterns to achieve minimum cost trunking by capitalizing on noncoincident network busy periods.

The dynamic, or time-varying, nature of the routing scheme is achieved by introducing several route choices. The routes consist of different orderings of the available paths (in Fig. 1, five paths). Each path consists of one or at most two links or trunk groups in tandem. The originating office [San Diego, Ca. (SNDG)] in Fig. 1 retains control over a dynamically routed call until it is either completed to its destination or blocked from the network. A call overflowing the second leg of a two-link connection [e.g., the Albany, N.Y.-White Plains, N.Y. (ALBY-WHPL) link of the SNDG-ALBY-WHPL path] is returned to SNDG, the originating office, for possible further alternate routing. Control is returned by using the common-channel interoffice signaling (CCIS) crankback signal sent from the via-node to SNDG.

Each of four routing sequences illustrated in Fig. 1 uses a different order of the five paths. Each routing sequence results in a different allocation of link flows, but all satisfy the point-to-point grade-of-service requirement. Allocating traffic to the optimum route choice during each load-set-period leads to design benefits due to the noncoincidence of loads. This route selection changes with time as shown in the columns on the right, thus, it is dynamic. The example shown indicates that in the morning the routing strategy is to offer the SNDG-WHPL traffic to routing sequence number one (starting with the direct trunk group to WHPL overflowing to the two-link connection through ALBY) and, in the afternoon, to routing sequence number two [overflowing to the two-link connection through Phoenix, Az. (PHNX)]. In the evening, routing sequence number three is used.

2.2 Design algorithm

The basic steps of the dynamic nonhierarchical routing (DNHR) design algorithm are shown in Fig. 2.[1] The algorithm combines several techniques for achieving network savings into a single, unified approach.[2–5] The steps of the design algorithm illustrated in Fig. 2 show that it is an iterative technique consisting of a router, an engineering module, and an update module. See Ref. 1 for a more complete discussion of this algorithm.

III. SOURCES OF LOAD UNCERTAINTY AND CONTROL OF ROUTING

The telephone network is designed on the basis of forecasted loads,

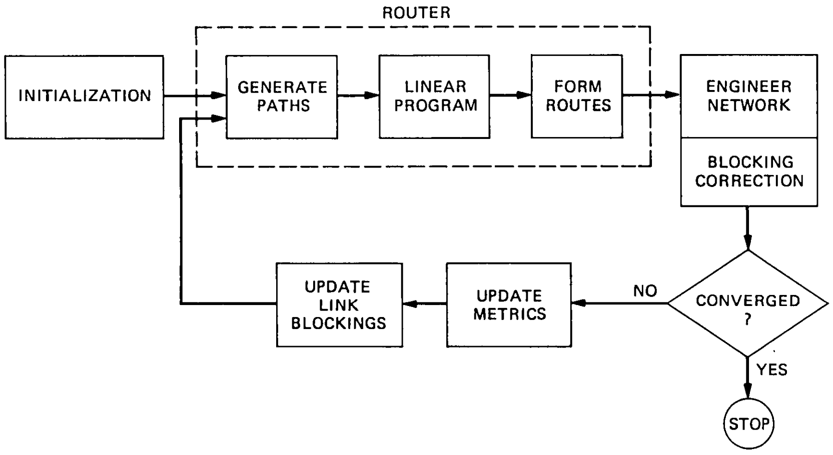


Fig. 2—Unified algorithm block diagram.

since the network capacity must be available before the loads occur. Errors in the forecast lead to uncertainty about the actual loads that will occur. In addition, each forecasted load is actually a mean load about which there occurs a day-to-day variation, characterized by a gamma distribution with one of three levels of variance.[6] Even if the forecast mean loads are correct, the actual realized loads exhibit a random fluctuation from day to day. Hence, there are two sources of load uncertainty: forecast error and day-to-day variation. Earlier studies have established that each of these sources of uncertainty requires the network to be augmented in order to maintain the grade of service.[7,8]

Control over network capacity is divided between planned servicing and demand servicing. Planned servicing is an annual process that determines where network capacity is needed to meet the future demand. Major inputs to planned servicing are the load forecast, which is subject to error, and the existing network. Trunk disconnects are determined in planned servicing when the forecast predicts declining or shifting loads, and the servicer is reasonably sure the trunks will not be needed in the next 1 to 2 years. This procedure reflects some reluctance to disconnect trunks, and results in a certain amount of reserve capacity being left in the network. Planned servicing drives the bulk of trunking activity, which is scheduled over the next yearly interval.

On occasion, the planned servicing strategy underprovides trunks at some point in the network, again, because of forecast errors, and the servicer must respond quickly to restore service. The process of correcting for these forecast errors is called demand servicing.[9] When some trunk groups are found to be overloaded as a result of the actual loads being larger than their forecast values, additional trunks are provided to restore the grade of service to the required value. Trunks will not usually be disconnected in demand servicing, and, as a result, the process leaves the network with a certain additional amount of reserve or idle capacity even when the forecast error is unbiased.[7]

The effects of day-to-day variation, unlike those of forecast error, are taken into account in the initial design of the network[8] and arise from the nonlinear relation between trunk-group load and blocking. When the load on a trunk-group fluctuates about a mean value, because of day-to-day variation, the mean blocking is higher than the blocking produced by the mean load. Therefore, additional capacity is provided to maintain the grade of service in the presence of day-to-day load variation.

The question is: To what extent can the capacity augmentation required by the uncertainties be reduced by dynamically controlling the routing patterns to meet the realized loads? A given realization of the loads can be expected to yield some parcel (point-to-point) loads which are higher than average and others which are lower. While part of the network is overloaded, another part might be underloaded. If the routing pattern can be adjusted to use the idle capacity of the underloaded portion of the network, the required capacity augmentation might be reduced.

Figure 3 shows the relation among the three levels of routing control, in which the design of trunk-group sizes and routing patterns of the network is viewed as feedback process. The outermost loop represents planned servicing in which link sizes and routing are planned approximately once a year. The next inner loop represents demand servicing which responds to service problems arising from unforecasted demand, at approximately one- to four-week intervals. From measurements of realized loads and blocking, the demand servicing algorithm determines augmentations to the link sizes and modifications to the routing patterns to correct for errors in the load forecast from which the design was made. The inner-most loop represents real-time routing in which only routing modifications are possible. This final level of routing control must deal with:

(*i*) Day-to-day load variations,

(*ii*) The effects of unforecasted demand, until the needed capacity augmentations can be made at the next demand servicing, and

(*iii*) Network management under overload and failure conditions.

IV. PLANNED AND DEMAND SERVICING TECHNIQUES

The flow diagram of Fig. 4 illustrates designing a network on the basis of forecast loads. Planned servicing accounts for both the current network and the forecast loads in planning network changes, and then

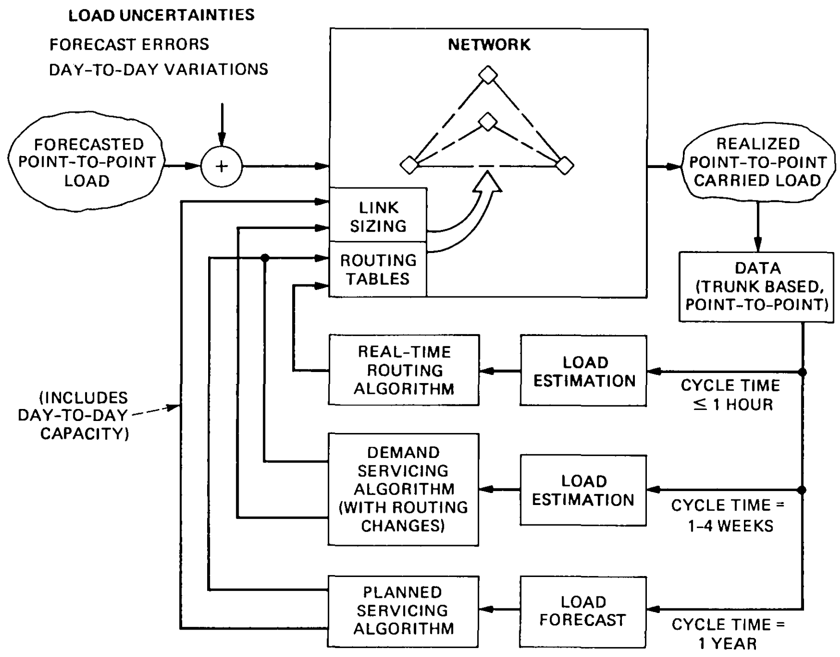

LOAD UNCERTAINTIES
FORECAST ERRORS
DAY-TO-DAY VARIATIONS
FORECASTED POINT-TO-POINT LOAD
+
NETWORK
LINK SIZING
ROUTING TABLES
REALIZED POINT-TO-POINT CARRIED LOAD
DATA (TRUNK BASED, POINT-TO-POINT)
(INCLUDES DAY-TO-DAY CAPACITY)
REAL-TIME ROUTING ALGORITHM
LOAD ESTIMATION
CYCLE TIME ≤ 1 HOUR
DEMAND SERVICING ALGORITHM (WITH ROUTING CHANGES)
LOAD ESTIMATION
CYCLE TIME = 1-4 WEEKS
PLANNED SERVICING ALGORITHM
LOAD FORECAST
CYCLE TIME = 1 YEAR

Fig. 3—Planned servicing, demand servicing, and real-time control as interacting feedback loops around the network.

demand servicing makes routing and trunking adjustments, if network performance under the realized loads becomes unacceptable because of errors in the forecast.

As discussed earlier, the planned servicing strategy tries to minimize reserve capacity, while maintaining an acceptable level of demand servicing. The model in Fig. 4 assumes that planned servicing is an annual process which predicts the required network capacity to meet the future demand. It considers both the demand forecast, which is subject to error, and the existing network. In dealing with forecast errors, planned servicing attempts to provide sufficient network capacity to meet these demands with a minimum of demand servicing. Our model assumes that the trunk network resulting from planned servicing is implemented immediately and then demand servicing is invoked to restore network service when shortages are detected.

4.1 Planned servicing methods

4.1.1 Conventional planned servicing

In the current hierarchical network, planned servicing begins by comparing the existing network with a network designed for the forecast loads. The design is made without reference to the existing

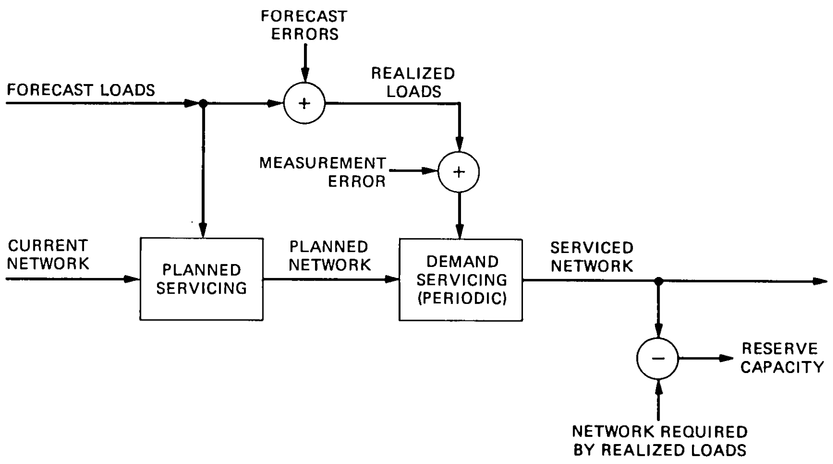


Fig. 4—Model of the planned and demand servicing process.

trunk group sizes. When the forecasting system calls for additional trunks on a group, the augments are usually implemented. If the forecasting system calls for fewer trunks, a disconnect policy is invoked to decide whether trunks should be disconnected, and, as discussed earlier, this policy reflects a degree of reluctance to disconnect trunks. The conventional method of planned servicing, which tries to guide the existing network towards an ideal network designed for the forecast loads, could be used in the DNHR network also. If there are substantial differences in the structure of the forecast network from one year to the next, conventional planned servicing, combined with the reluctance to disconnect trunks, might produce augmentation that could be avoided by better use of existing capacity.

4.1.2 Incremental planned servicing

Given the reluctance to disconnect trunks, it seems reasonable that planned servicing should design a network considering the trunking which is in place. This can be done using incremental planned servicing where, instead of designing an ideal network from the beginning, we design a minimum-cost augmentation of the existing network to meet the forecast loads. Slight modifications to the traffic routing and engineering blocks in the design algorithm acomplish this change, which allows us to use given initial link capacities as lower bounds on the designed link capacities. In effect, the algorithm takes into account the reluctance to remove trunks. This procedure limits the trunk augments to those required to meet the forecast loads and, thus, achieves a lower reserve capacity.

4.1.3 Generalized incremental planned servicing

Incremental planned servicing designs a minimum-cost augmentation to the existing network. We generalize this procedure to set minimum trunk group sizes and to allow trunk disconnects. This is done by deriving, for each group, a lower threshold and an upper threshold for its size, and using these thresholds to make initial adjustments to the group size (as described below) prior to incremental planned servicing.

The size thresholds for a group are based upon the forecast loads of the corresponding direct parcel, and are determined by choosing a minimum value $r_{\min}$ and a maximum value $r_{\max}$ for the ratio $r \triangleq$ (trunk group size)/(direct parcel load). The lower (reserve) threshold for the group size is based on the forecast peak load of the direct parcel in the next year and corresponds to the ratio $r_{\min}$. The upper (disconnect) threshold is based on the forecast peak load of the direct parcel over the next two years, with some allowance for forecast error, and it corresponds to the ratio $r_{\max}$.

The limits $r_{\min}$ and $r_{\max}$ were chosen by examining the range of values of the ratio in a typical DNHR network design. In general, the ratio has a smaller spread of values for large parcels than for small parcels. With large parcel loads, the corresponding group can be quite efficient carrying just the direct parcel; hence, its size to a large extent depends just on the direct parcel. For small parcel loads, however, the group size is less dependent on the direct parcel load and is more influenced by the alternate routing parcels carried on that group; hence, the ratio is expected to have a wider range of values.

Table I shows the limits for the ratio (trunk group size)/(direct parcel load), as a function of the direct parcel load.

The lower threshold $T_{\min}$ and upper threshold $T_{\max}$ for a trunk group are determined in terms of the forecast loads for the corresponding direct parcel:

Table I—Limits on $r = T/L$ = (trunk group size)/(direct parcel load)

Load L (erlangs)	$r_{\min}(L)$	$r_{\max}(L)$
0–5	0.3	4.5
5–10	0.4	4.5
10–25	0.5	3.0
25–50	0.7	3.0
50–100	0.95	2.5
>100	1.05	1.5

Let L_i = peak forecast load for the direct parcel in year i, $i = 1, 2$.
β_i = forecast uncertainty factor for year i, $i = 1, 2$, introduced to allow for probable error in the load forecast. The results presented were obtained with $\beta_1 = 1.15$, $\beta_2 = 1.3$ corresponding to 0.15 coefficient of variation in the forecast.

Then,

$$T_{\min} \triangleq r_{\min}\,(\beta_1 L_1) * \beta_1 L_1$$

$$T_{\max} \triangleq \max[r_{\max}(\beta_1 L_1) * \beta_1 L_1,\ r_{\max}(\beta_2 L_2) * \beta_2 L_2],$$

where $r_{\min}$ and $r_{\max}$ are the appropriate limits established for ratio (T/L), such as those in Table I.

With these lower and upper thresholds, we then define an initial size for each group, which depends on its current size as follows:

(*i*) If the current size of a group is between its lower and upper thresholds, its initial size equals its current size.

(*ii*) If the current size of the group is below its lower threshold, its initial size equals the lower threshold.

(*iii*) If the current size of the group is above its upper threshold, its initial size equals the upper threshold.

We use the initial network defined in this manner as the starting network for incremental design (i.e., minimum-cost augmentation) to arrive at the forecast network for each future year, as described in Section 4.2. Comparing the result with the current network, we determine the actual augments and disconnects that must be made to implement the forecast network. Under normal growth conditions, the current trunks are most often used in the initial network, not the upper and lower trunk limits, and the primary effect is to route traffic on the actual trunks in place and, thus, minimize rearrangments.

4.2 Planned servicing algorithm

As noted earlier, the unified algorithm (UA) for DNHR network design is an iterative procedure with four basic steps (Fig. 2): selection of cost-effective traffic paths, optimization of path flows, sizing the trunk groups (engineering) to correspond to the optimum flows, and updating of marginal link costs and optimum link blockings for the next iteration. The proposed procedures for planned and demand servicing involve modifications to the flow optimization and engineering routines of the UA to allow the existing link capacities to be used as lower bounds on the designed link capacities.

We now describe the modifications to the flow optimization and engineering procedures for use in planned servicing.

Let

L = number of links.

H = number of design hours.

$$\left.\begin{array}{rl} b_i^{\max} = & \text{maximum permitted blocking on link } i. \\ b_i^h = & \text{blocking of link } i \text{ in hour } h,\ h = 1, \cdots, H. \\ y_i^h = & \text{carried load on link } i \text{ in hour } h,\ h = 1, \cdots, H. \\ a_i = & \text{capacity of unaugmented link } i \text{, in carried load at blocking } b_i^{\max}. \\ \Delta a_i = & \text{capacity augmentation, in carried load, on link } i. \\ M_i = & \text{marginal cost of augmentation, in cost per erlang, on link } i. \end{array}\right\} i = 1, \cdots, L$$

4.2.1 Flow optimization

The object is to allocate the traffic flow of each hour among its admissible paths so as to minimize the cost of the required link capacity augmentations. On each link, for a given number of existing trunks and the maximum economic blocking determined from economic considerations,[3] there is a maximum load that can be carried on that link without augmentation; this is the unaugmented initial capacity of that link.

The flow optimization problem for planned servicing is now stated as a linear program in which the decision variables are the flow assignments and the augmentations Δa_k above the existing link capacities a_k (instead of total link capacities as in the design problem described in Ref. 1), and the cost to be minimized is the marginal cost of augmentation

$$\sum_{k=1}^{L} M_k \Delta a_k.$$

This formulation ensures that efficient use is made of existing link capacities, by means of routing changes if needed, before link augmentations are proposed.

4.2.2 Engineering

In engineering we are given the traffic routing and loads and find the needed augmentation to those groups which exceed their maximum permitted blockings. This is accomplished by the following iterative procedure:

(*i*) Begin with assumed link blockings $\hat{b}_i^h$ (e.g., the link blockings in the unaugmented network) subject to $\hat{b}_i^h \le b_i^{\max}$, $i = 1, \cdots, L$.

(*ii*) Calculate the corresponding carried link loads y_i^h, under the known routing and assumed link blockings.

(*iii*) If for all $h, y_i^h \le a_i$, the capacity of the unaugmented link at its maximum blocking, then the link needs no augmentation; if $y_i^h > a_i$, the required augmentation Δa_i is determined by engineering the link for load y_i^h at blocking $b_i^{\max}$.

(*iv*) From the link loads y_i^h and link sizes computed in (*ii*) and (*iii*), we recalculate all the link blockings b_i^h; if $|b_i^h - \hat{b}_i^h|$ is not sufficiently close to zero for all i in all hours h, redefine $\hat{b}_i^h = b_i^h$, $i = 1, \cdots, L$, $h = 1, \cdots, H$, and return to (*ii*).

The marginal link costs and optimum link blockings are, in general, determined for the forecast loads each year during planned servicing, although in some years their values might change little from the previous year.

4.3 Demand servicing methods

Between successive planned servicings, if the realized loads exceed forecast values and cause unacceptable blocking, then quick corrective action, called demand servicing, is needed. In the current hierarchical network, demand servicing is usually limited to trunk group augmentations. However, in the DNHR network, the basic routing patterns are time-variable, and hence, routing modifications can be used in demand servicing to reduce network augmentation. To the extent that routing changes can be substituted for the installation of trunks, rearrangements are also reduced.

Demand servicing consists of three steps:

(*i*) Detecting the need for demand servicing, i.e., determining whether or not all parcels are receiving adequate service,

(*ii*) If servicing is needed, then determining the best combination of routing changes and link augmentations that will restore the desired grade of service at minimum cost of augmentation, and

(*iii*) Implementing the routing changes and link augments. These steps are discussed in more detail below.

4.3.1 Detection of service problems

Point-to-point blocking measurements are needed in the DNHR network to determine the level of service being provided and, thus, to detect the existence of service problems. Because of measurement errors and day-to-day traffic variations, such blocking measurements will have an inherent statistical variability which must be allowed for by establishing acceptable bands for the measured blockings.

4.3.2 Demand servicing algorithm

The need here is for a simple procedure to determine the corrective action required; there is no attempt to redesign the whole network, or to disconnect trunks, if the network is found to be overprovided for the realized loads. We use the flow-optimization routine to determine the optimum traffic routing for the realized loads. Using this optimum routing, we use the engineering routine to determine the link augmentations required to limit link blockings to their maximum permitted values. If some parcel blockings remain higher than desired after this step, we invoke the blocking correction procedure discussed in Ref. 1 to correct the problem. Thus, a procedure similar to the incremental planned servicing algorithm is used to determine the required changes in demand servicing.

4.3.3 Implementing routing changes

Using demand servicing with routing changes, the network routing might change very frequently: possibly at every demand servicing interval. Manual administration of such frequent routing changes in the network would be unmanageable, and a substantial degree of automation will be required in implementing the routing changes. This suggests a network routing data base which would receive routing revisions from the output of demand servicing. With such a data base and an automatic update system, the administration of routing changes in demand servicing appears quite feasible.

4.4 Servicing results

The servicing model in Fig. 4 was used to simulate the servicing process on a 28-node network (Fig. 5) to determine the effectiveness of the proposed planned and demand servicing procedures. The network, starting from a design for the first year's forecast loads, was taken through 10 years of the servicing process, each iteration consisting of a forecast at the beginning of the year, followed by a demand servicing during the year. The forecast parcel loads grew at a 5-percent annual rate. To simulate forecast error, the realized parcel loads in each year were assumed to be normally distributed about the forecast loads, with a 15-percent coefficient of variation.

The following three schemes were compared:

Scheme A—Conventional planned servicing and demand servicing with routing changes.

Scheme B—Incremental planned servicing and demand servicing with routing changes.

Scheme C—Generalized incremental planned servicing and demand servicing with routing changes.

In schemes A and B, no disconnects were allowed in planned servicing in order to simulate complete reluctance to disconnect trunks. In all three schemes, no disconnects were allowed in demand servicing.

Figure 6 shows the evolution of network reserve capacity for the three servicing schemes, measured by the percentage difference in cost between the realized network and an ideal network designed for the realized loads. Figure 7 shows the cumulative demand servicing trunk augments for the three schemes as percentages of the number of trunks in the starting network. Figure 8 shows, for each scheme, the level of demand servicing in each year, as measured by the trunk augments in demand servicing as a percentage of trunks in the realized network.

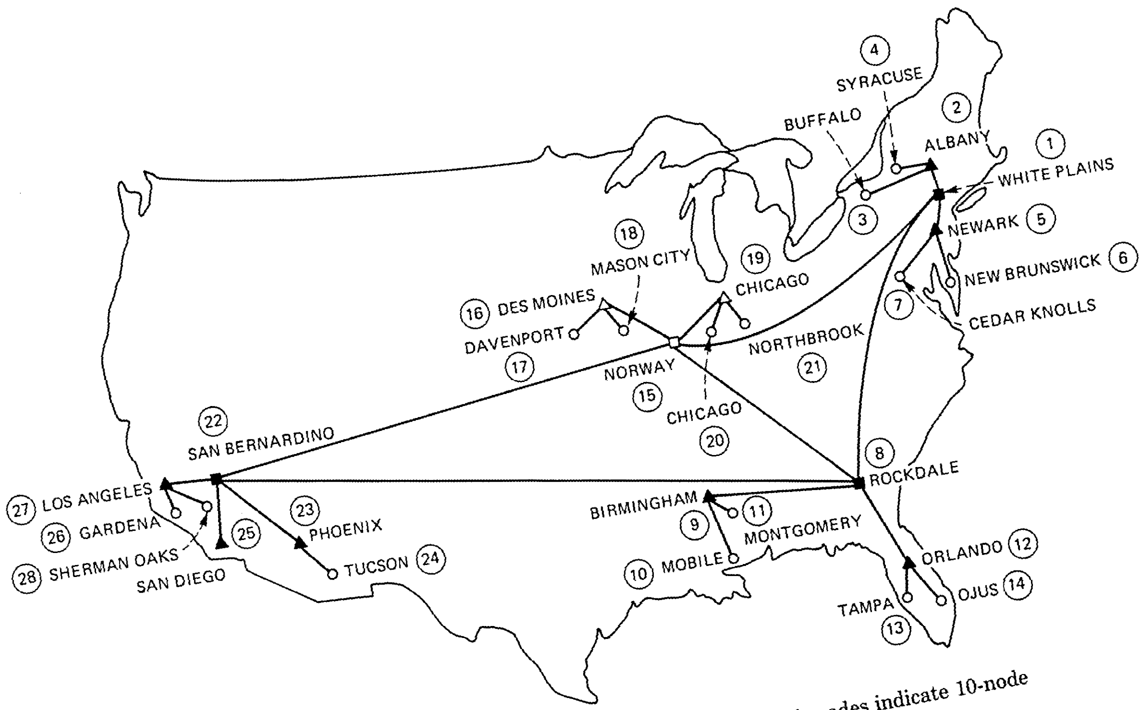


Fig. 5—Intercity network model for 28-node network. (Dark nodes indicate 10-node model.)

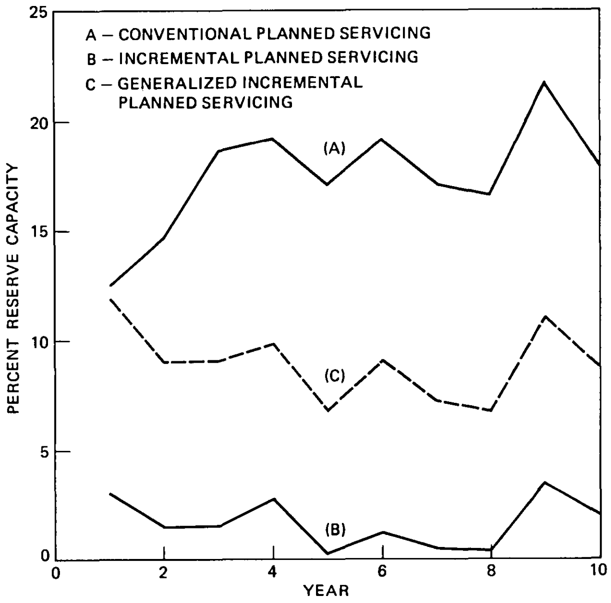


Fig. 6—Evolution of network reserve capacity.

We note from Figs. 6 and 7 that incremental planned servicing (scheme B), as expected, achieves a lower reserve capacity than conventional planned servicing (scheme A) but requires more demand servicing augments in response to forecast error. The generalized incremental planned servicing method (scheme C) falls between the other two both in reserve capacity and in demand servicing. Compared to conventional planned servicing, it achieves a striking reduction in reserve capacity for a modest increase in demand servicing.

Figure 8 shows that, in all three schemes, demand servicing rearrangements in each year are in the range of 1 to 4 percent of the trunks in the network, a level that is quite favorable in comparison with the demand servicing level in the current hierarchical network (estimated to be about 10 percent of the trunks in the network).

Figure 9 shows the cumulative *total* rearrangements (consisting of augments and/or disconnects in planned servicing and augments in demand servicing) for the three schemes as percentages of the number of trunks in the starting network. We note that when the total trunk changes occurring in planned and demand servicing are considered, incremental planned servicing (scheme B) produces the fewest rearrangements and conventional planned servicing the most, with generalized incremental planned servicing falling in between the other two. However, in general, more time is available for implementing planned servicing rearrangements than demand servicing rearrangements,

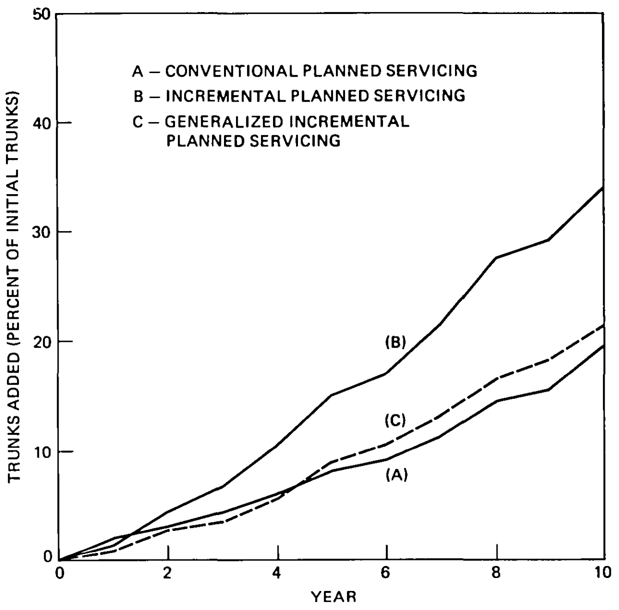


Fig. 7—Cumulative demand servicing rearrangements.

which are called for on short notice, to correct existing service problems. It is therefore likely that demand servicing rearrangements are more expensive than planned servicing rearrangements. Taking this into account, we may expect that the administrative cost of rearrangements is smaller with generalized incremental planned servicing than with just incremental planned servicing.

Figures 6 and 7 have pointed to the trade-off that exists between reserve capacity and demand servicing rearrangements, and generalized incremental planned servicing has been proposed as a method of securing the desired trade-off between these two aspects. For example, by multiplying the factors $r_{\min}$ in Table I by a factor $\alpha \geq 0$, we can parametrize the resulting levels of reserve capacity and demand servicing. The value $\alpha = 1$ corresponds to curve C in Figs. 6 to 9; $\alpha < 1$ results in lower reserve capacity and increased demand servicing, while $\alpha > 1$ leads to higher reserve capacity and reduced demand servicing.

Figure 10 is a curve of average reserve capacity versus the average level of demand servicing (average over 10 years) in scheme C, with α as the parameter. For comparison, the two points corresponding to schemes A and B, respectively, are also plotted. Point B is almost the same as the limiting case $\alpha = 0$, while point A lies above the curve, showing that a more favorable trade-off can be obtained with scheme C than with A. This curve is a quantitative expression of the trade-off between reserve capacity and demand servicing.

We conclude that generalized incremental planned servicing, combined with demand servicing with routing changes, is an efficient method of controlling reserve capacity and the level of demand servicing in the network. On the basis of the results presented in this paper, a reserve capacity level of about 7 to 10 percent appears suitable for the assumed 15-percent coefficient of variation of load forecast error. We have not presented a direct comparison between demand servicing with and without routing changes. Such a comparison has been made on a 10-node subset of the 28-node network (Fig. 5). The results show that routing changes in demand servicing reduce network cost by about 2 to 3 percent and demand servicing rearrangements by about 15 percent.

V. REAL-TIME ROUTING CONTROL

Planned servicing and demand servicing will account for known, systematic load variations including unforecasted demand in the planned routing patterns and trunk group sizes. The only routing decisions necessary in real time involve conditions that also become known in real time: day-to-day load variations, network failures, and network overloads.

The day-to-day component of load variation is not systematic and/ or easily predictable because it involves daily load shifts which are

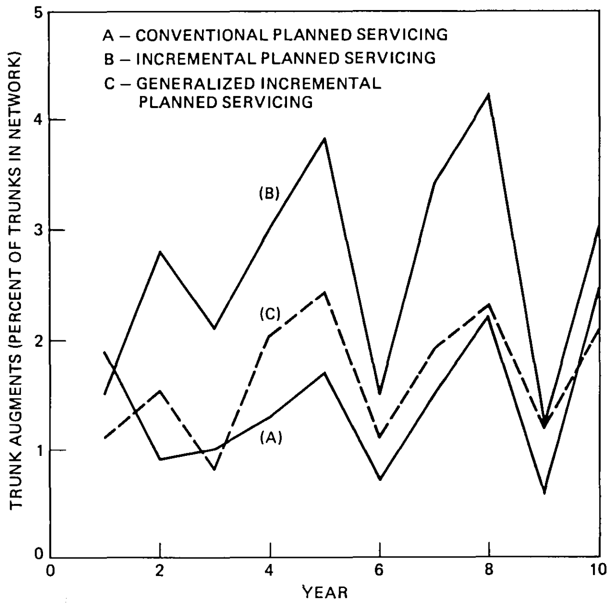


Fig. 8—Demand servicing level. Trunk augments in demand servicing as a percentage of trunks in network.

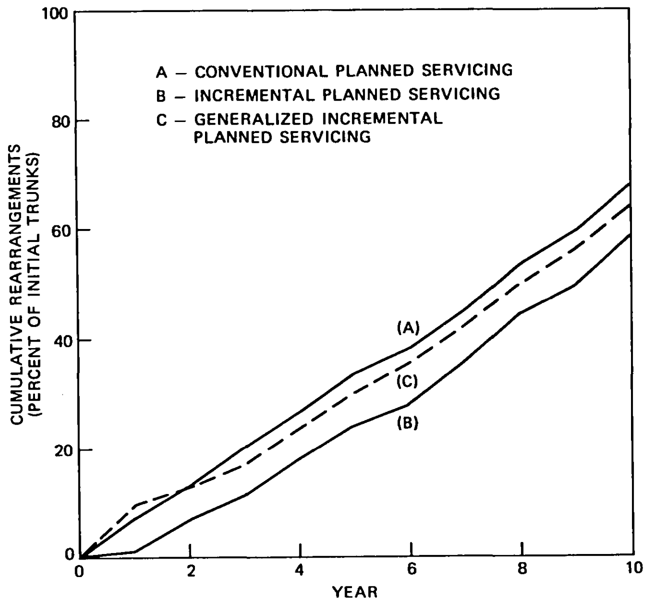


Fig. 9—Cumulative trunk rearrangements. Augments plus disconnects.

essentially random from one day to the next.* Reasonably accurate load patterns can be predicted months in advance; the unforecasted demands can then be identified and corrected over a period of a few weeks (as the loads develop), but daily load variations must ultimately be identified in real time.

The network design will size the network to accommodate all expected load patterns including day-to-day load variations. Sizing the network for day-to-day variations will guarantee that some capacity will stand idle at least some of the days. If planned routing patterns were totally preprogrammed, no advantage could be taken of temporarily idle network capacity to complete calls that might otherwise be blocked. For this reason, a method of extending routing patterns beyond the preprogrammed sequence to include real-time decisions was devised.

Real-time routing can be used to improve network service. Service improvement is significant even with relatively simple procedures—the improvement can also be equated to an equivalent trunk cost savings of about 2 to 3 percent or improved network service with a higher overall completion rate. Real-time dynamic routing should also

* It is known that some daily variations are systematic (e.g., Monday is usually higher than Tuesday). However, in the present environment, these known changes are ignored and lumped into the stochastic model.

improve network performance somewhat in the event of network failures, especially when some amount of reserve capacity is available for redirecting traffic flows from their usual patterns.

5.1 Real-time routing method

A relatively simple real-time procedure is investigated which is a natural extension of the two-link routing procedure being proposed. The method appends to each sequence of two-link paths, engineered by the design algorithm for the expected network load, additional two-link (real-time) paths to be used only after the normal sequence is exhausted and only when idle capacity is available.

Dynamic trunk reservation is used to help recognize idle network capacity. Access to trunks on a particular trunk group is allowed only after a specified number of trunks—the reservation level—is available. Reservation guarantees that capacity is truly idle and accessing it will produce minimal interference with normal traffic.

The selection of real-time paths for each point-to-point pair can be done as a natural extension of the design and servicing algorithms. These algorithms recognize noncoincidence factors and can identify groups that are expected to have slack capacity at a particular time. Causes of slack capacity include forecast errors, the disconnect policy, and reserve capacity from modular engineering. The candidate real-time paths would be selected off-line from the list of paths generated

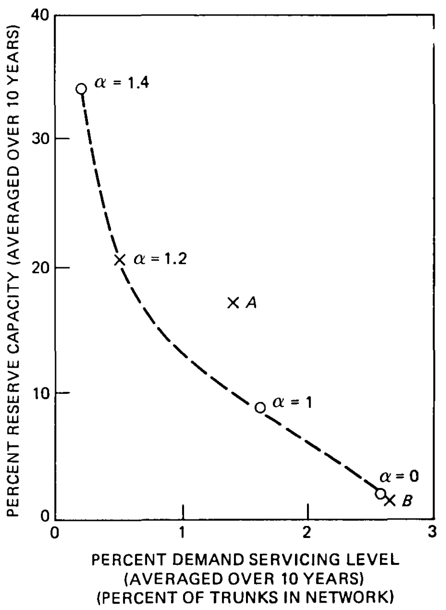


Fig. 10—Trade-off between reserve capacity and demand servicing level.

by the algorithms. This list contains a large number of potential paths to be used in forming the planned routes. The real-time paths are chosen from those paths not already used as part of the planned route. It should also be noted that the larger the allowed number of real-time paths, the lower the blocking for an individual point-to-point pair. However, there are administrative costs, storage limitations, and real-time penalties restricting the number of allowed paths.

Three means of routing calls among the real-time paths were studied: a sequential method identical to the planned routing patterns described in Section 2.1, and two cyclic routing methods in which the real-time calls are rotated among the real-time paths analogous to the current automatic-out-of-chain-routing (AOOCR) method (described in Section 5.2). It was found that the sequential method provided equal or better performance than the others and, hence, is preferred because of its ease of implementation.

5.2 Alternative methods of real-time routing

There are several possible methods for implementing real-time routing which use call-by-call routing decisions. Methods under active study are discussed in this section.

5.2.1 Automatic-out-of-chain routing

The No. 4 ESS provides an expansive control AOOCR.[10] With AOOCR, overflow from a final trunk group, which is presently routed to a reorder announcement or manually rerouted, is sent to an out-of-chain route where it will attempt to complete. Up to seven out-of-chain routes can be identified for each final group, and the No. 4 ESS will spread the overflow traffic uniformly over these routes, as capacity is available. This is accomplished by using a cyclic routing method which, for each call, tries the path following the previous path attempted. All out-of-chain traffic is accompanied by a CCIS traveling class mark so that it receives special treatment at the via-office to prevent shuttling and will be turned back unless the via-route has available capacity. If a call fails to find a free trunk leaving the via-node, the call is blocked at the originating office and the particular out-of-chain path is turned off from further attempts for a period of about 30 seconds.

5.2.2 Learning automata

Another decentralized routing scheme involves the use of learning automata.[11-13] A learning automaton is a machine (or an algorithm) whose actions are constantly modified by feedback from its environment. The updating procedures used to modify the actions of the different types of automata determine their learning characteristics. One example is the $L_{R\text{-}I}$ automaton (linear reward-inaction). In this scheme, if a particular routing choice, or action, gets a positive response from the environment (i.e., the call is completed), the probability of choosing this action for subsequent calls is increased. However, if a negative response is received, the action probabilities are not modified. A detailed mathematical model for the $L_{R\text{-}I}$ automaton can be found in Ref. 12.

Models for the sample mean (M) automaton and the linear reward-penalty ($L_{R\text{-}I}$) automaton have also been developed for possible use in the telephone network. Simulation studies with simplified networks show that these automata perform better than a fixed (hierarchical) routing strategy.[11,13]

5.2.3 Centralized real-time routing

A centralized routing system has been investigated by Bell Northern Research.[14] In this implementation, the selection of candidate paths at each switch is recalculated every two seconds. The path selection is done by a central routing processor, based on the busy-idle status of all trunks in the network. This system was field tested for four months on nine switching systems in Toronto. A computer analysis of actual call-by-call demand on the network showed that advanced routing provides more uniform and better service characteristics than the hierarchy during overloads.

The preceding have been examples of different techniques of real-time routing. However, the decentralized real-time routing method investigated here seems to be a practical method for large networks at the present time.

5.3 Simulation results

In this section, we summarize the results of a call-by-call simulation using the 28-node intercity network model (Fig. 5). The simulations selected were guided by results from small analytic models, and the simulation results were obtained using the 10 a.m. (EST) busy-hour load and routing.

The call-by-call simulation model assumed transparent nodes; that is, no queuing or blocking was modeled for the switching systems in the network. Poisson arrivals were used to model originating calls together with an exponential holding time distribution having a mean of five minutes. Day-to-day variations were modeled with a gamma distribution with a variance equal to $0.13a^{\phi}$, with $\phi = 1.5$ to model low daily variations. The parameter "a" represents the point-to-point offered load in erlangs. Reserve capacity was modeled by using a uniform distribution on the trunk group size centered about a 7-percent average reserve capacity and varied uniformly between 3 and 11 percent. It is taken as a typical level of reserve capacity for the dynamic routing network.

5.3.1 Results for day-to-day variations

Table II illustrates the performance of the real-time routing scheme in terms of three network performance criteria used in the simulation:

(*i*) Average network blocking—indicates the effectiveness of completing calls.

(*ii*) Maximum parcel blocking—indicates the interference of real-time calls with traffic using the planned route.

(*iii*) Crankbacks per originating call and machine attempts per originating call—indicate the switching and signaling effort to complete real-time calls. Total crankback attempts are counted; machine attempts include originating, terminating, crankback (counted as one full attempt), and tandem completing.

Notice that real-time routing has little impact on total machine attempts, and that network blocking with real-time routing is reduced by about 75 percent. Figure 11 illustrates the improvement in blocking performance with real-time routing as a function of reserve capacity. We see that its effectiveness increases as reserve capacity increases, which is expected. This improvement in network service will translate into greater revenues by providing a higher network completion rate.

5.3.2 Results for network overload and failure

Real-time routing should not degrade network performance under overload and failure conditions. The simulation results verify that real-time routing does not degrade average network blocking or individual parcel blockings under general and focused overload conditions. However, control is needed to limit the generation of crankback messages under these conditions.

A link failure of the Los Angeles, Ca.-Newark, N.J. (LSAN-NWRK) link (23 trunks) was simulated with the results shown in Table III.

These results also show that the average network blocking improves using real-time routing with a slight increase in switching effort. Comparing Tables II and III, we note that under a link failure the

Table II—Performance of real-time routing (low day-to-day variations—7 percent reserve capacity)

Real-time Routing Used?	Avg. Network Blocking	Max. Parcel Blocking	Crankbacks per Orig. Call	Attempts per Orig. Call
No	0.00249	0.017	0.0207	2.17
Yes	0.00058	0.009	0.0230	2.17

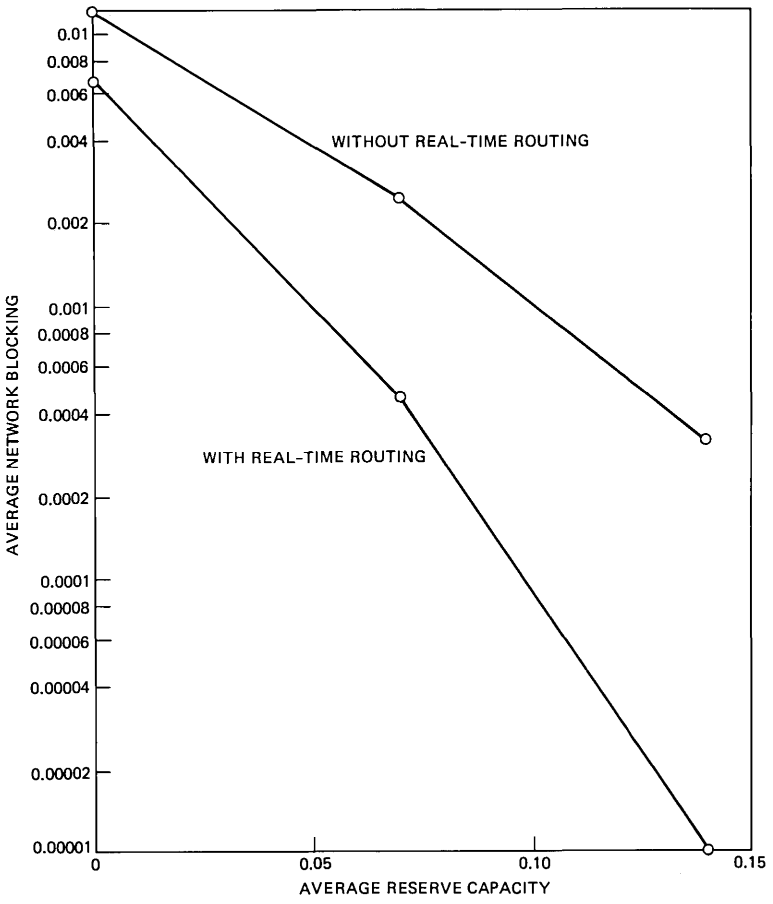


Fig. 11—Average network blocking versus average reserve capacity (low day-to-day variations).

average network blocking increases slightly, but that real-time routing maintains the maximum parcel blocking at the same level.

A node failure of the NWRK node was simulated with the results shown in Table IV. The high blocking parcel in this case was the ALBY-WHPL parcel. Newark, N.J. was normally a via point for this parcel. Additional real-time calls using the ALBY-WHPL link also contributed to the higher parcel blocking. However, it is significant that the overall network blocking improved when real-time routing was applied. All calls destined for NWRK overflowed all the planned and real-time paths causing a large number of real-time path attempts, plus crankback messages. Normally, automatic network management controls would cancel such attempts to alleviate this situation. Another interesting

Table III—Performance under link failure (LSAN-NWRK failure—7 percent reserve capacity)

Real-time Routing Used?	Avg. Network Blocking	Max. Parcel Blocking	Crankbacks per Orig. Call	Attempts per Orig. Call
No	0.00264	0.023	0.0258	2.17
Yes	0.00062	0.009	0.0291	2.18

Table IV—Performance under node failure (NWRK node failed—7 percent reserve capacity)

Real-time Routing Used?	Avg. Network Blocking*	Max. Parcel Blocking*	Crank-backs per Orig. Call	Attempts per Orig. Call
No	0.00819	0.082	0.183	2.19
Yes	0.00099	0.089	0.187	2.20

* Excluding traffic to the NWRK node.

phenomenon is that most parcels not normally using NWRK as a via point achieved better than normal service. This occurred because the trunk groups normally carrying NWRK traffic are relatively free (NWRK traffic cannot complete). Hence, other parcels can make use of these relatively lightly loaded groups to achieve better than normal service.

VI. ACKNOWLEDGMENTS

We thank A. H. Westreich who did the initial work on computer implementation of the servicing model. Thanks are also in order for L. T. Nguyen, who carried out many of the subsequent computer simulations.

REFERENCES

1. G. R. Ash, R. H. Cardwell, and R. P. Murray, "Design and Optimization of Networks with Dynamic Routing," B.S.T.J., this issue.
2. M. Eisenberg, "Engineering Traffic Networks for More than One Busy Hour," B.S.T.J., *56,* No. 1 (January 1977), pp. 1–20.
3. C. J. Truitt, "Traffic Engineering Techniques for Determining Trunk Requirements in Alternate Routed Networks," B.S.T.J., *33,* No. 2 (March, 1954), pp. 277–302.
4. B. Yaged, Jr., Long Range Planning for Communications Networks, Polytechnic Institute of Brooklyn, Ph.D Thesis, 1971.
5. J. E. Knepley, "Minimum Cost Design for Circuit Switched Networks," Technical Note Numbers 36–73, Defense Communications Engineering Center, System Engineering Facility, Reston, Virginia, July, 1973.
6. R. I. Wilkinson, "A Study of Load and Service Variations In Toll Alternate Route Systems," Proc. 2nd Int. Teletraffic Congress, The Hague, July, 1958, Document No. 29.
7. R. L. Franks et al., "A Model Relating Measurements and Forecast Errors to the Provisioning of Direct Final Trunk Groups," B.S.T.J., *58,* No. 2 (February, 1979), pp. 351–77.
8. D. W. Hill and S. R. Neal, "Traffic Capacity of a Probability-Engineered Trunk group," B.S.T.J., *55,* No. 7 (September, 1976), pp. 831–42.

9. C. R. Szelag, "Trunk Demand Servicing in the Presence of Measurement Uncertainty," B.S.T.J., *59,* No. 6 (July-August, 1980), pp. 845-60.
10. V. S. Mummert, "Network Management and its Implementation on the No. 4 ESS," Int. Switching Symp., Japan, 1976.
11. P. R. S. Kumar and K. S. Narendra, "Learning Algorithm Model for Routing in Telephone Networks," Systems and Information Sciences Report No. 7903, Yale University, May, 1979.
12. K. S. Narendra and M. A. L. Thathachar, "On the Behavior of a Learning Automaton in a Changing Environment with Application to Telephone Traffic Routing," Systems and Information Sciences Report No. 7803, Yale University, October, 1978.
13. D. M. McKenna and K. S. Narendra, "Simulation Study of Telephone Traffic Routing Using Learning Algorithms," Technical Report No. 7806, Yale University, 1978.
14. A. E. Bean and E. Szybicki, "Advanced Traffic Routing in Local Telephone Networks: Performance of Proposed Call Routing Algorithms," 9th Int. Teletraffic Cong., Torremolinos, Spain, October, 1979.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Improving the Quality of a Noisy Speech Signal

By M. M. SONDHI, C. E. SCHMIDT, and L. R. RABINER

(Manuscript received December 18, 1980)

In this paper we discuss the problem of reducing the noise level of a noisy speech signal. Several variants of the well-known class of "spectral subtraction" techniques are described. The basic implementation consists of a channel vocoder in which both the noise spectral level and the overall (signal + noise) spectral level are estimated in each channel, and the gain of each channel is adjusted on the basis of the relative noise level in that channel. Two improvements over previously known techniques have been studied. One is a noise level estimator based on a slowly varying, adaptive noise-level histogram. The other is a nonlinear smoother based on inter-channel continuity constraints for eliminating the so-called "musical tones" (i.e., narrow-band noise bursts of varying pitch). Informal listening indicates that for modest signal-to-noise ratios (greater than about 8 dB) substantial noise reduction is achieved with little degradation of the speech quality.

I. INTRODUCTION

The idea that a vocoder may be used to improve the quality of a noisy speech signal, has been around for about twenty years. To the best of our knowledge the first such proposal was made in 1960 by M. R. Schroeder.[1] The basic idea of this proposal can be explained with the help of Fig. 1, as follows:

Figure 1a shows a typical short-term magnitude spectrum of a voiced portion of a noisy speech signal. Let $S(\omega)$ denote the envelope of this spectrum. (Recall that the "channel gains" of a vocoder are estimates of this envelope at the center frequencies of the channels. The fine structure of the spectrum is attributed to the harmonics of the fundamental voice frequency.)

Figure 1b shows a "formant equalized" version, $\bar{S}(\omega)$, of the envelope. The peaks in S and $\bar{S}$ occur at the same frequencies but the peaks of $\bar{S}$ (unlike those of S) are all of the same amplitude.

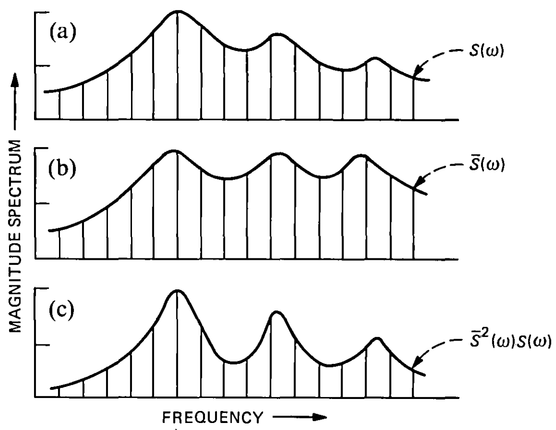


Fig. 1—Illustration of noise stripping by increasing the dynamic range between formant peaks and noise valleys. (a) Original spectral envelope and fine structure. (b) Formant-level equalized spectral envelope. (c) The product spectrum $\bar{S}^2(\omega)S(\omega)$ in which the ratios between formant peaks and valleys is larger than in the original spectrum.

The proposal is, essentially, to generate a signal with a fine structure as close as possible to that of the original speech signal, but with an envelope given by $\bar{S}^n S$, where n is some intetger, say, 1 or 2. Except for a scale factor, the spectral envelope of the resulting signal is the same as that of the original signal at the formant peaks, but is considerably reduced in the valleys. As shown in Fig. 1c this processing effectively reduces the overall noise level. Of course, the formant peaks also become sharper, i.e., the formant bandwidths get reduced.

Reference 1 describes two implementations of this idea: a frequency domain method in which the envelope is modified by modifying the channel gains of a self-excited channel vocoder, and a time domain method in which the same effect is achieved by repeated convolution.

In many practical cases of interest, the noise is additive and uncorrelated with the speech signal. In such a situation, if it were possible to estimate the spectral level of the noise as a function of frequency, then the noise reduction could be achieved in a somewhat different manner. Suppose the noisy speech is applied to the input of a channel vocoder (see Section II for a detailed description). Let the output of the kth channel be $y_k = s_k + n_k$, where s_k is in the speech signal and n_k the noise signal in that channel. Let N_k^2 be the average power of the noise and S_k^2 that of the speech signal. Then, assuming that the noise and speech are uncorrelated, the average power of the noisy speech is given by

$$Y_k^2 = S_k^2 + N_k^2 \tag{1}$$

Now Y_k^2 can be estimated directly from the output signal y_k. If an estimate of N_k^2 is available, as postulated, then $(Y_k^2 - N_k^2)^{1/2}$ provides an estimate of the magnitude of the signal alone in the kth channel. Thus, if the level of the channel signal is multiplied by the ratio of this estimated signal power to overall power, then a noise reduction is achieved.

In 1964, at the suggestion of M. R. Schroeder, this "spectral subtraction" idea was implemented as a BLODI language computer program by one of us (MMS) in collaboration with Sally Sievers.[2] Besides spectral subtraction, one other feature was incorporated into this implementation. It had been recently demonstrated that autocorrelation and cepstrum pitch extraction are quite accurate and reliable for noisy speech signals with signal-to-noise ratio (s/n) as low as 6 dB.[3,4] Such extractors provide a clean excitation signal even from a highly noisy speech signal. Therefore, the self-excitation described in Ref. 1 was replaced by a voiced-unvoiced (buzz-hiss) signal derived from an autocorrelation pitch extractor.

Although this implementation demonstrated the feasibility of the basic idea, the computer facilities available at that time did not allow a thorough investigation of the effects of changing various parameters and configurations. Also, since digital hardware was not yet readily available, it did not appear likely that such noise-stripping techniques would find application in the immediate future. For these reasons these techniques were not actively pursued at that time.

Since the mid-seventies, presumably due to the vastly improved digital technology and renewed military interest, noise-stripping has again attracted considerable attention. The renewed interest in this problem appears to have started in 1974, when Weiss et al. independently discovered the spectral subtraction method.[5] Except for the fact that the filter bank of the channel vocoder was replaced by short-term Fourier analysis, the implementation of Weiss et al. was quite similar to the one described above. During the past five or six years several studies have explored this and other methods for noise removal. Notable among these is the work of Boll, Berouti et al., and McAulay and Malpass.[6,7,8] A review of these and other studies is given in a recent paper by Lim and Oppenheim.[9]

In view of the current interest in noise removal, we have recently been experimenting with the spectral subtraction method by computer simulation. Subsequent sections of this paper describe the results of our experiments.

From the brief description given above, it is clear that spectral subtraction is expected to be useful only in cases when the noise is additive. With this constraint, there are basically two types of situations in which this method might find application:

(*i*) The speech may be produced in a noisy environment, e.g., in the cockpit of an airplane. In such a situation the spectrum of the noise is unknown a priori. This information must be estimated from the noisy speech signal itself, e.g., during intervals of silence between speech bursts. The algorithm for estimating the noise spectrum is, therefore, one of the most important parts of the simulations described later.

(*ii*) The speech itself may be generated in a quiet environment but might be transformed to a noisy signal because of the action of a coding device. Examples where such noise may be modelled as additive are pulse-code modulation (PCM) coders, and delta modulators whose step size is chosen such that granular noise predominates over the slope-overload noise. In such cases, both the level of the noise and its spectral composition might be known a priori. Use of this a priori information simplifies the system and improves its performance.

There is a third way in which noise may enter the communication channel additively. The speech signal may be generated in a quiet environment but the listener may be in a noisy environment. A message sent over the public address system at a busy railway station is such an example. In this case, the problem is to preprocess the speech signal in such a way that its intelligibility is least impaired by the noise. Some work on this problem has been reported in the literature;[10] however, we will not deal with this problem.

Before turning to a description of our simulations, it is worth emphasizing that we deliberately used the word "quality" rather than "intelligibility" in the title of this paper. Ideally, of course, one would like the intelligibility also to be increased. However, this is not absolutely essential. It is quite annoying and fatiguing to have to listen to a noisy speech signal for any length of time. Therefore, a device that reduces or eliminates the noise can be quite useful even if the cleaner signal is no more intelligible than the noisy one.

II. THE BASIC STRUCTURES

Two basic channel vocoder configurations for implementing spectral subtraction were simulated. For reasons that will become apparent from the following descriptions, we call these configurations self-excited and pitch-excited, respectively.

2.1 The self-excited configuration

A block diagram of the self-excited method of noise removal is shown in Fig. 2. The noisy speech, sampled 10,000 times per second is first passed through a bank of N equispaced bandpass filters that span the telephone channel bandwidth (approximately 200 to 3200 Hz). The processing of the output of the bandpass filter is identical for each

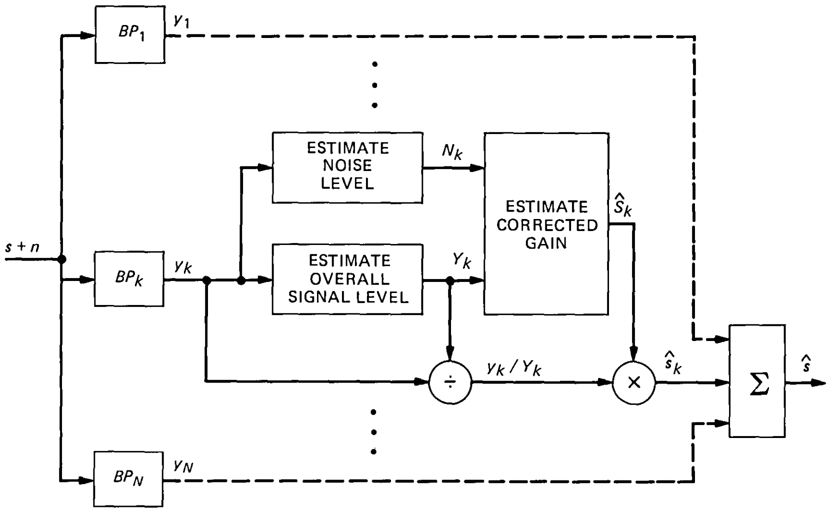


Fig. 2—Block diagram of the self-excited channel bank noise stripper consisting of a bank of N FIR bandpass filters with gain estimation and correction within each channel.

channel. In the kth channel, the following operations are performed on the output y_k:

(*i*) The level (magnitude) of the noisy speech signal, Y_k, is estimated.

(*ii*) In a parallel path the level of the noise, N_k, is estimated.

(*iii*) The estimates N_k and Y_k are used to derive an estimate $\hat{S}_k$ of the level of the uncorrupted speech signal in the kth channel.

(*iv*) The adjusted channel signal is computed by the relation

$$\hat{s}_k = y_k \frac{\hat{S}_k}{Y_k}. \tag{2}$$

Clearly $\hat{s}_k$ has the desired estimated magnitude $\hat{S}_k$. The sum $\hat{s} = \sum_{k=1}^{N} \hat{s}_k$ then provides the final processed output.

2.2 The pitch-excited configuration

A block diagram of the pitch-excited method is shown in Fig. 3. The estimates $\hat{S}_k$, $k = 1, 2, \cdots N$, are obtained exactly as in the case of the self-excited configuration. However, the adjusted channel signals are obtained differently.

(*i*) The noisy speech signal is first processed by a pitch extractor which also provides the voiced/unvoiced classification. The particular pitch extractor used is described in Ref. 11.

(*ii*) The output of the pitch extractor is used to provide a clean excitation signal which consists of a Gaussian noise during unvoiced

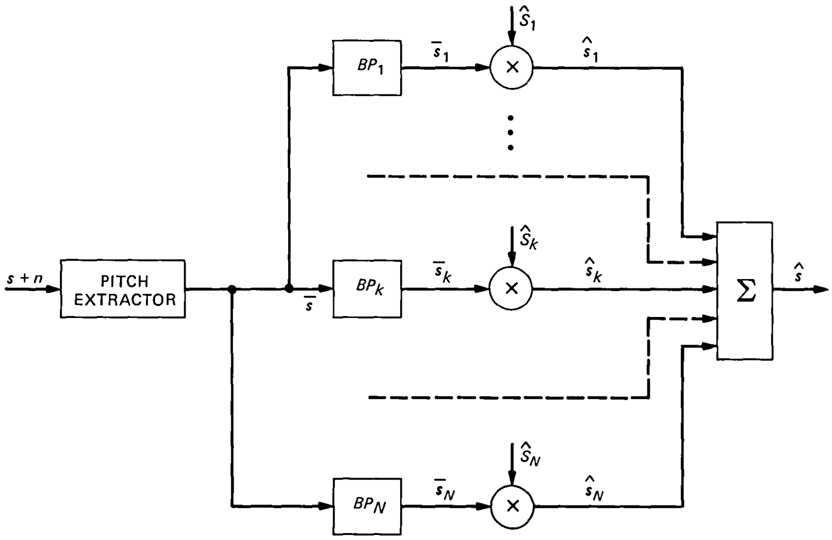


Fig. 3—Block diagram of the pitch-excited channel bank noise stripper in which a voiced/unvoiced excitation is used in place of the bandpass channel signals.

portions and a train of impulses at the pitch rate during voiced segments.

(*iii*) This clean excitation signal is passed through a bank of bandpass filters, identical to the ones shown in Fig. 2, to give channel signals, $\overline{s_k}$, which are approximately equal in magnitude.

(*iv*) The adjusted channel signal is computed as

$$\hat{s}_k = \overline{s_k} \cdot \hat{S}_k. \tag{3}$$

As before, $\hat{s}_k$ has the correct magnitude and, as before, the sum of these adjusted channel signals gives the final processed output.

As discussed in the next section, the estimates of $\hat{S}_k$ are computed every 0.01 s (i.e., 100 times a second). In our initial experiments the channel gains were held constant between estimates. In this case, the gain jumps in value every 0.01 s, producing annoying audible clicks. These clicks were eliminated by replacing each jump by a linear interpolation of the channel gains over 6 speech samples (i.e., over 0.6 ms).

III. ALTERNATIVE CONFIGURATIONS SIMULATED

Several modified versions of the basic configurations of Figs. 2 and 3 have been simulated, and several sentences processed with these simulations. The alternatives that we have studied in some detail are two choices for the number of channels; two methods of estimating

Y_k; two methods of estimating N_k; and two methods of estimating $\hat{S}_k$. These will now be described.

3.1 The filter bank

Two designs were simulated, each with equispaced filters. In one design 16 channels (200-Hz wide) were used, and in the other 32 channels (100-Hz wide). The filter responses and the sum of the responses for each design are shown in Fig. 4. (Each filter was a linear phase, finite impulse response (FIR) filter of duration 88 samples in the 16-channel filter bank and 176 samples in the 32-channel filter bank.)

3.2 Estimating Y_k

The two methods of estimating the magnitude, Y_k, of the noisy channel signal are shown in Fig. 5. Either $|y_k|$ or y_k^2 is low-pass-filtered to 30 Hz. In the second case, the square-root of the output of the low-pass filter is computed. The impulse and frequency responses of the low-pass filter [a 3rd order infinite impulse response (IIR) Bessel filter] are shown in Fig. 6.

The choice of bandwidth of the low-pass filter is governed by a compromise between the following two requirements: For accurate estimation of Y_k the averaging time should be as large as possible, i.e.,

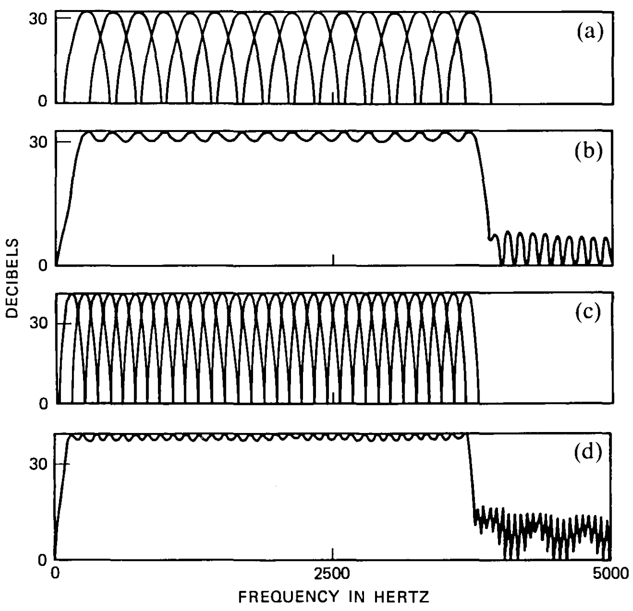


Fig. 4—(a) Frequency responses of individual filters of the 16-channel filter bank. (b) Composite responses for 16-channel filter bank. (c) Frequency responses of individual filters of the 32-channel filter bank. (d) Composite responses for 32-channel filter bank.

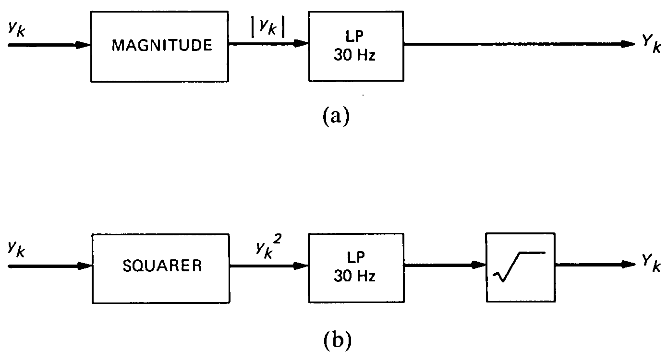


Fig. 5—Signal processing for estimating the overall signal level using either a magnitude (a) or a squaring (b) nonlinearity, followed by a low-pass filter. In the case of the squaring nonlinearity, the low-pass filter is followed by a square root box.

the filter bandwidth should be as small as possible. On the other hand, the spectrum of speech varies with time so the bandwidth should be as large as possible to track these variations. The usual compromise cut-off frequency in channel vocoders is about 30 Hz.

Note that the outputs of the low-pass filters need be sampled only 60 times/s. To allow for the roll-off of the filters, the sampling rate was chosen as 100/s. Somewhat surprisingly, a much higher sampling rate was found to degrade performance. We will explain this paradox in Section IV (The Musical Tones).

3.3 Estimating N_k

During intervals of silence in the speech, the input signal consists of noise alone. Therefore, one possible estimate for N_k is the smallest value attained by Y_k. However, because of statistical fluctuations, Y_k quite rapidly takes on an unrealistically low value. Therefore, this estimate is quite unsatisfactory. In order to avoid such problems with outliers, the method schematized in Figure 7 has been simulated.

As a first step, the magnitude of y_k is estimated by a procedure identical to that of Fig. 5, except that the low-pass filter has a cut-off frequency of 10 Hz instead of 30 Hz. (The impulse response of the 10-Hz filter is quite similar to that of the 30-Hz filter with the time axis scaled by a factor of 3.)

As before, the cut-off frequency of the low-pass filter should be chosen no larger than that necessary to follow the time-variations of the noise spectrum. Our choice of 10 Hz is an extremely conservative value. For most applications a cut-off frequency of 1 Hz or less should suffice.

Analogously to the estimation of Y_k, we have two ways of estimating

N_k, which differ only in the type of nonlinearity used. Figure 7 shows the front end of the alternate noise estimator that we have simulated.

Let $Z_k(n)$ be the estimates of the magnitude of y_k obtained by one of these methods, sampled every 0.01 s. Then the algorithm for finding the noise level is as follows:

(*i*) Store $Z_k(n)$, $n = 1, \cdots, Q$ in a buffer of size Q.

(*ii*) Find the smallest value such that the next higher value is within 6 dB of it. Call this smallest value MIN.

(*iii*) Make a histogram with 1-dB bins of all the values that lie in the range MIN to MAX = MIN + 15 dB.

(*vi*) Declare K times the magnitude corresponding to the peak of the histogram, as the noise level.

(*v*) Get next sample.

(*vi*) If this sample is greater than MAX, discard it and go to step (*v*).

(*vii*) If the sample is less than MAX replace the oldest sample in the buffer by the new sample and go to step (*ii*).

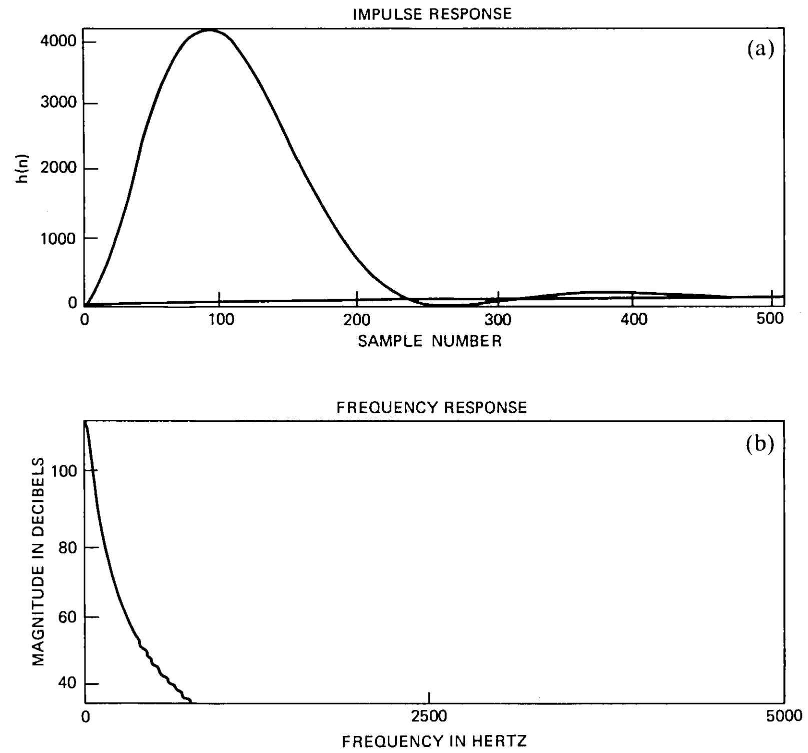


Fig. 6—Impulse response (a) and frequency response (b) of the 30-Hz, 3rd order, Bessel IIR filter used in estimating overall signal level.

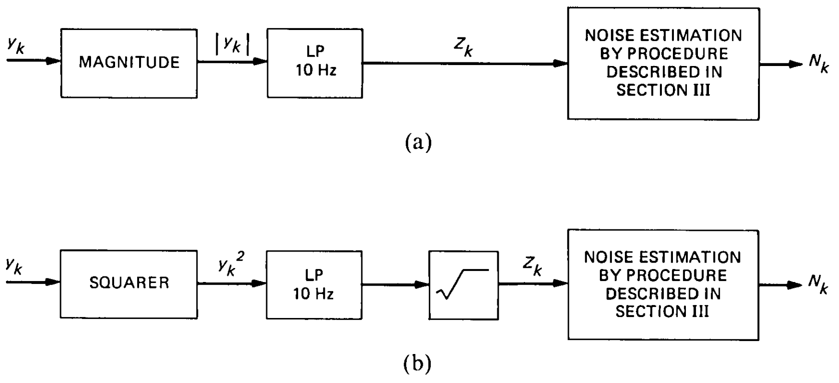

y_k
MAGNITUDE
$|y_k|$
LP 10 Hz
z_k
NOISE ESTIMATION BY PROCEDURE DESCRIBED IN SECTION III
N_k
(a)
y_k
SQUARER
y_k^2
LP 10 Hz
$\sqrt{\ }$
z_k
NOISE ESTIMATION BY PROCEDURE DESCRIBED IN SECTION III
N_k
(b)

Fig. 7—Signal processing for estimating noise level. In (a) and (b) the estimates of channel levels are obtained exactly as in (a) and (b) of Fig. 6, except that the low-pass filters have a bandwidth of 10 Hz instead of 30 Hz. The final step in both (a) and (b) is an adaptive noise estimation procedure based on a time-varying noise histogram.

After some experimentation, $Q = 100$ and $K = 3$ or 3.5 were found to be most satisfactory for the range of s/n's considered. All experiments to be described later were performed with these values of Q and K.

Careful considerations of the above algorithm should convince the reader that this procedure ignores occasional low values of Z_k; it guards against sudden increased in Z_k because of the onset of speech; and finally, it allows adaptation to a slowly varying noise level.

3.4 Estimating $\hat{S}_k$

As mentioned in the introduction, under the assumption that s_k and n_k are uncorrelated, $\hat{S}_k$ should be estimated as $\hat{S}_k = (Y_k^2 - N_k^2)^{1/2}$. However, there is statistical fluctuation because of the finite averaging time even if the assumption is strictly valid. Therefore, sometimes the estimated value of Y_k is less than that of N_k. In such cases, $\hat{S}_k$ is set to zero. Thus, our first procedure for estimating $\hat{S}_k$ is

$$\hat{S}_k = \sqrt{Y_k^2 - N_k^2}, \qquad Y_k > N_k \tag{4a}$$

$$= 0, \qquad Y_k \leq N_k. \tag{4b}$$

A second estimate that we have tried is

$$\hat{S}_k = Y_k - N_k, \qquad Y_k > N_k \tag{5a}$$

$$= 0, \qquad Y_k \leq N_k. \tag{5b}$$

IV. THE MUSICAL TONES

We have processed several speech signals through a variety of noise-stripping algorithms obtained by selecting from the alternatives listed above. The results will be discussed in detail in the next section.

However, one general observation that can be made is that although the noise can be eliminated even from severely noisy speech signals, it gets replaced by "musical tones." These are short bursts of more or less sinusoidal tones with varying pitch. The explanation of the origin of these tones is as follows: The algorithm for estimating $\hat{S}_k$ will, in general, set several consecutive channels to zero (in the valleys between formant peaks). If because of statistical fluctuation a single channel escapes elimination, (i.e., is above the noise threshold) it will appear as a narrow band signal much like a tone-burst with the center frequency of the channel. Every time such a tone-burst appears, its pitch will be determined by the particular isolated channel that gives rise to it. (At this point, the paradox mentioned in Section 3.2 can be explained. Consider a channel where the speech energy is very low, i.e., $Y_k \approx N_k$. If Y_k is oversampled, the number of times it crosses N_k increases and, therefore, the number of spurious noise bursts also increases.)

We have found one simple procedure to combat this phenomenon. Every time the channel gains $\hat{S}_k$ are updated, the new values are scanned across channels (i.e., the array $\hat{S}_k(n)$, $k = 1, \cdots N$ is examined at the time instant n). A nonzero value which is flanked by zero on both sides, is set equal to zero.

For male voices, this removal of isolated channels works extremely well. However, the method does not work well for high-pitched female voices when the noise level is high. The reason is that in the latter case there may be only one or two pitch harmonics in a formant peak. Thus, the noise stripping algorithm might create several isolated channels in formant regions as well. Therefore, removal of isolated channels removes a large part of the speech signal, along with the musical tones. We do not have a good method of dealing with this problem for high-pitched voices at high noise levels.

Suggestions for combatting these musical tones have also been made by Boll and Berouti et al.[6,7] We have compared our method to these other methods and find that except in the case of high-pitched voices at very low s/n our method performs better.

V. EXPERIMENTS

We have processed several sentences spoken by male and female speakers through noise-strippers obtained by selecting most of the possible combinations of alternatives listed in Section II. Uncorrelated Gaussian noise was added to provide the noisy test samples. The variance of the Gaussian distribution was selected so as to provide several s/n's in the range of about 4 to 16 dB. We have not conducted formal listening tests on the outputs. However, informal listening (mostly by the three authors of this report) allows us to draw the following general conclusions:

(*i*) The algorithm is capable of following slow variations of the noise spectrum. We tested this on noise with a flat spectrum but with a sudden jump of 6 dB in its amplitude. The algorithm attained the correct estimates of channel gains within 0.5 s.

(*ii*) The implementation with the 32-channel filter bank performs better than the one with the 16-channel filter bank.

(*iii*) In Fig. 5 the second alternative performs significantly better than the first, i.e., the square root of the average power is a better statistic to use than the average of the magnitude.

(*iv*) Power subtraction [eqs. (4a, 4b)] and spectral magnitude subtraction [eqs. (5a, 5b)] appear to work about equally well even at the lowest s/n (about 5 dB) that we tried.

(*v*) The factor K in the noise estimation procedure of Section III, should be set to about 3 or 3.5 for the range of s/n's considered in this paper.

(*vi*) For male voices, if isolated channels are eliminated as discussed in Section III, then pitch excitation and self excitation both work about equally well.

(*vii*) For female voices it is not possible to remove isolated channels at high noise levels (s/n's less than say 8 dB). In these situations, pitch excitation is superior to self excitation.

VI. CONCLUSION

We have described several algorithms based on spectral subtraction for removing noise from a noisy speech signal. Two noteworthy features of our simulations are the manner in which we estimate the noise level and the manner in which we deal with the narrow-band, time-varying noise bursts that commonly arise in spectral subtraction methods.

Our simulations were arranged to provide flexibility to allow us to test various modifications. However, it should be possible to realize the final preferred version of our algorithm in digital hardware that runs in real time.

The ultimate test of such a system is a large-scale statistical study of listeners' preference. We have not attempted such a study. However, on the basis of informal listening we can say that our method is quite successful in removing noise, and in most instances is superior to the other methods known to us.

REFERENCES

1. M. R. Schroeder, U.S. Patent No. 3,180,936 filed December 1960, issued April, 1965.

2. This simulation by M. M. Sondhi and S. Sievers is documented only in an unpublished internal report. The procedure is described briefly in a review paper by M. R. Schroeder and A. M. Noll, Paper A21, 5th Int. Congress on Acoust., Liege, Belgium, 1965. The device was also patented by M. R. Schroeder, U.S. Patent No. 3,403,224, filed May 1965, issued September, 1968.
3. M. M. Sondhi, "New Methods of Pitch Extraction," IEEE Trans. Audio, *AU-6,* No. 2 (June 1968), pp. 262–6.
4. A. M. Noll, "Cepstrum Pitch Determination," J. Acoust. Soc. Am, *41,* No. 2 (February 1967), pp. 293-309.
5. M. R. Weiss, E. Aschkenasy, and T. W. Parsons, "Processing Speech Signals to Attenuate Interference," IEEE Symp. on Speech Recognition, Pittsburgh, April 1974, Contributed Papers, pp. 292-3.
6. S. F. Boll, "Suppression of Acoustic Noise in Speech Using Spectral Subtraction," IEEE Trans. Acoust. Speech and Sig. Process., *ASSP-29,* No. 2 (April 1979), pp. 113–20.
7. M. Berouti, R. Schwartz, and J. Makhoul, "Enhancement of Speech Corrupted by Acoustic Noise," Proc. Int. Conf. Acoust. Speech, and Sig. Process. (April 1979), pp. 208–11.
8. R. J. McAulay and M. L. Malpass, "Speech Enhancement Using a Soft-Decision Maximum Likelihood Noise Suppression Filter," Tech. Note 1979-31, MIT Lincoln Lab., Lexington, Ma., June 1979.
9. J. S. Lim and A. V. Oppenheim, "Enhancement and Bandwidth Compression of Noisy Speech," Proc. IEEE, *67,* No. 12 (December 1979), pp. 1586–604.
10. I. B. Thomas and R. J. Niederjohn, "Enhancement of Speech Intelligibility at High Noise Levels by Filtering and Clipping," J. Aud. Eng. Soc., *16,* No. 10 (October 1968), pp. 412–15.
11. R. A. Gillman, "A Fast Frequency Domain Pitch Algorithm," J. Acoust. Soc. Am., *58,* Supplement No. 1 (Fall 1975), p. S62.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Cutoff Calls and Telephone Equipment Reliability

By M. TORTORELLA

(Manuscript received March 31, 1981)

It is often difficult or expensive to measure cutoff calls, which are usually caused by failures and malfunctions in some component of the telephone network. Therefore, it is desirable to have an indirect method for estimating the number of cutoff calls caused by equipment failures in a switching system or facility. This paper discusses a mathematical model that can be used to determine the cutoff call rate in a network component as a function of the failure modes and failure rates in the component, and the call holding time distribution. It includes a discussion of a paradigm for developing reliability objectives that directly reflect service as it is seen by end users. The mathematical model, an M/M/c/c queuing system with server failures, is described. A strong law of large numbers and a central limit theorem for the number of cutoff calls—accumulated either according to the number of failures or over time—are developed. An example from a switching system is given to show how these results are applied in specific cases.

I. INTRODUCTION AND SUMMARY

The purpose of this paper is to describe a mathematical model for the rate of cutoff calls caused by failures and malfunctions in telephone equipment. The cutoff call behavior of almost any piece of telephone equipment that serves callers can be analyzed using this technique, but the primary applications we have in mind are large integrated systems containing many components, such as switching systems and transmission systems (trunk groups). The model relates the rate of cutoff calls produced by failures in the equipment and its subsystems to the failure modes in the equipment, their severity and frequency of occurrence, and the call-holding-time distribution.

The interaction of telephone call requests with service equipment has often been successfully described using queuing models. Therefore, it seems reasonable that a study of the effects of equipment failures on the calls in a telephone system should be feasible within the context of the classical queuing models of telephony. This is the approach adopted here, with the additional feature that the servers may be unreliable and subject to failures of a kind that cause the customer (if any), in service at a position whose server fails, to be dropped from the system at the time of the failure. Forys and Messerli have previously studied trunk groups containing unreliable servers.[1] Their interest was in characterizing the effect on arriving calls of one or more short-holding-time (hence, very likely to be malfunctioning) trunks in the group, whereas here the interest is primarily in the effects of unreliable servers that may fail singly or together in groups, on customers who are already in service.

The paper is divided into five sections. Section II contains a general discussion of reliability objectives as they apply to telephone equipment, and the paradigm for developing reliability objectives that directly reflect service as it is seen by the customer. We observe that the critical step that has been lacking is the ability to translate equipment reliability into rates of occurrence and duration of customer-perceivable problems, such as cutoff calls and network connection failures, that are produced by failures and outages.

In Section III, the structure of the mathematical models to be used is described. The basic structure is one of a queuing system with server failures, and, using this structure, the probability that a call in the system will be cut off is determined. The way one describes mathematically the system organization and failure modes is also covered in this section. The probability of cutoff can be computed under quite general conditions on the arrival process, the service times, and the queue discipline, because it depends only on what happens after the customer enters service.

Section IV describes a more specialized queuing model, in the context of which certain limit laws for the cumulative number of cutoff calls can be obtained. This is the M/M/c blocking system with server failures, and both a strong law and a central limit theorem are obtained. The eventual use of these limit laws, as the basis for constructing statistical tests for determining compliance with objectives, is also briefly discussed. Section V is devoted to the single-server case, and explicit calculation of all parameters of interest.

Finally, Section VI gives an example of the application of this theory to the estimation of cutoff call rates in a toll switching system. It is important to be able to do this kind of analysis because one may wish to predict cutoff call performance for a system that is still being designed. This technique is then an example of an indirect, albeit approximate, method of estimating a cutoff call rate for which no satisfactory direct method may be available.

Two appendices contain all proofs and other mathematical details that, otherwise placed, would interfere with the flow of the text.

II. RELIABILITY OBJECTIVES AND CUSTOMER SERVICE

2.1 General

It is currently recognized that the most desirable way to specify performance and service objectives for telephone network equipment is to use, in addition to economic information, considerations of how the operation of this equipment affects service as it is seen by the customer. In order to do this for reliability objectives, we need to realize that customers do not perceive outages, failures, and malfunctions as such. They are aware of them only insofar as they cause service problems detectable by users who generally are not aware of the internal operations of the telephone network. To achieve the goal of determining equipment reliability objectives based on customer needs and expectations, then, the following steps are required:

(*i*) Determine the customer-perceivable service effects of the reliability problems to be controlled.

(*ii*) Determine the quantitative relationships between the frequency and duration of reliability problems in the system or equipment and the rates of occurrence and duration of the service effects found in the first step.

(*iii*) Use these relationships to translate the customer service objectives for the system, which control the customer-perceivable effects stemming from reliability problems, into internal reliability objectives.

This paper focuses on the second step for a particular service effect: cutoff calls.

2.2 Service effects

From the customer's point of view, the primary service effects of failures and malfunctions are cutoff calls, ineffective attempts (network connection failures), isolation (line and toll), and transmission impairments (excessive loss, noise, etc.). Cutoff calls will be discussed at length below. Ineffective attempts, or network connection failures, can be caused by failures and malfunctions because the unavailability of a portion of the telephone network increases the network's blocking probability during the time this portion of the network is out of service. If the failed equipment is a customer's loop, or a part of the local central office that disables the customer's line functions, causing a customer to be unable to communicate with the local central office, the customer experiences *line isolation* for the duration of the failure.

If the failed equipment is a toll-connecting trunk group from a customer's local central office, the customer experiences *toll isolation,* meaning that toll calls to or from certain areas cannot be placed or received.

Transmission impairments can be caused by malfunctions such as equipment operating outside tolerances. These phenomena are well-understood and measurement plans are in place to return relevant information about transmission problems to maintenance forces so that abnormal conditions may be corrected. These will not be discussed further.

The rate of network connection failures and the duration of isolations are determined primarily by the duration of the outage. Thus, analysis of these service problems is helpful in determining reliability objectives and maintenance policies to limit outage duration. We will see below that the rate of cutoff calls is primarily driven by the rate of failures, so that analysis of cutoff calls is useful mainly in determining objectives for frequency of occurrence of outages. Of course, a comprehensive strategy for reliability management should deal with these complementary facets of equipment reliability in a unified way, and maintenance (service restoration and equipment repair) policies play an important part here. An objective for frequency of occurrence of failures, together with a maintenance policy, implies a certain total outage time for the equipment. Similarly, an objective for total outage time, together with a service restoration and equipment repair strategy, limits the number of times outages may occur. Although this paper deals only with cutoff calls and frequency of occurrence of outages, it should be borne in mind that a unified approach to reliability objectives, combining considerations not only of cutoff calls and outage frequency but also of network connection failures and outage duration, is most desirable.

2.3 Types of failures included

2.3.1 Causes of cutoff calls

A cutoff call is a connected (stable) call that has been terminated other than by an on-hook by either party. The event of termination is sometimes referred to as a *cutoff,* for short (as is a call that is so affected). The terminology is intended to connote an unintentional, unexpected interruption. International Telegraph and Telephone Consultative Committee (CCITT) terminology refers to a cutoff-causing failure in a switching system as a "premature release malfunction in an exchange."

Cutoff calls are caused by equipment failures (including recovery actions), and other external factors, such as radio fades and in-band talkoff (simulation of the 2600-Hz supervisory signal by a signal emitted by one of the parties). The termination takes place at the instant the failure or other event begins, so the rate of cutoffs is influenced primarily by the rate of failures (this is demonstrated in eq. (7)). Cutoffs are related to reliability, then, just as ineffective attempts or network connection failures are related to availability. To determine the rate of cutoff calls seen by a telephone user, the cutoff call performances of individual switching and transmission systems are combined in a network model. A suitable model is one for the reliability of a series system consisting of switching systems and trunk groups.

2.3.2 Scope of the model

The reliability problems covered by the model are those of failure and repair of entire systems and parts of systems, and those failures and malfunctions that may not completely disable a system or subsystem, but that cut off calls when they occur. In the first case, systems and subsystems will be considered to be either operating properly and fully available for use, or not operating at all and unavailable. Cutoff calls caused by improper operation, or operation outside tolerances, of a system or subsystem can also be treated. The key notion is that any event that causes cutoff calls when it occurs can be called a "failure" for purposes of this discussion. The model can accommodate many different "failure" modes, as long as the occurrence times and severities of these events can be characterized sufficiently well that failure processes and cutoff impacts (Section 3.2) can be assigned. In particular, the model could in principle include such events as radio fades and in-band talkoff as "failure modes." However, in studying cutoff calls as related to equipment reliability, this is not recommended, because these are external events, not caused by an equipment failure or malfunction which could be controlled by preventive or corrective action by the telephone company.

As for causes of failure, for the model there is no restriction on the cause of the failure or malfunction. All that is required is that one be able to list the kinds of events that cause cutoffs, and describe probabilistically the times between incidents for each kind of event. The scope of this work encompasses all failures which lead to cutoff calls, regardless of cause, including hardware (component failure), software and firmware faults, human intervention errors, office database errors, and so on.

2.4 Uses of the mathematical model

This model finds three primary applications in system analysis and design. First, it can be used to make the translation which allows system cutoff call objectives to determine reliability objectives and maintenance policies for switching and transmission systems. Reliability objectives should not be viewed as ends in themselves, but only as means by which objectives for those aspects of customer service that are affected by reliability problems can be met. Second, they have value as predictive tools. System designers can use the probability of cutoff as a figure of merit for hypothetical system designs, architectures, and reliability characteristics. Systems that have not yet been constructed can be compared for this aspect of service quality, and this comparison can be a factor in deciding among competing designs, for example. Its third major use is to provide a framework within which to perform statistical tests, based on observed cutoff call rates, to see whether objectives are being met. In systems where cutoff calls are not measured, the models enable inferences to be made about the cutoff call rate based on other kinds of data, such as reliability records of equipment failures and malfunctions. Since cutoff calls are often difficult or expensive to measure in a given system, these techniques provide another, perhaps more attractive, means of understanding this important service problem.

III. MODEL DESCRIPTION AND PROBABILITY OF CUTOFF

In this section, we discuss the structure of the mathematical model for cutoff calls and reliability of telephone equipment. It starts with an outline-like guide to the sequence of results which make up the mathematical model. As an aid to seeing where the details fit into the overall scheme, this guide can be referred to while reading the remainder of the paper. A queuing model with server failures is covered, as is the organization of the servers and failure modes. Physical interpretation is given, and some probabilistic insights are added to help clarify the ideas. Finally, the probability that a call that has been accepted by the system will be cut off is computed.

3.1 Outline of results

3.1.1 Relation of probability of cutoff to equipment reliability

The first important result obtained is in Section 3.6, where the probability that a call that has been accepted by the system will be cut off is computed. This probability can be thought of as a figure of merit for the system in question, and can be computed under weak assumptions about the arrival process, the holding times, and the interfailure times. However, the probability of cutoff, by itself, is not enough to give a good understanding of how a system will behave with respect to cutting off calls. In particular, there are two important questions on which knowing the probability of cutoff alone sheds no light. First, does the observed cutoff call rate have any relation to the probability of cutoff? Second, what is the structure of the stochastic process which counts the number of calls cut off in a time interval? How much variability can be expected in such a count, for example?

3.1.2 Measurements and consistent estimation of the probability of cutoff

Section IV is devoted to an exploration of these questions for a more specialized system, the M/M/c/c queue with server failures. In answer to the first question, Corollaries 5 and 6 show that the observed cutoff call rate converges to the probability of cutoff as given by eq. (8). This means that, in this case, measurements can be relied upon to consistently estimate the probability of cutoff, which may be controlled by an objective. Also, when a prediction about the cutoff probability in a new system is made, it can reasonably be expected that the cutoff call rate shown by the system in operation will approach the predicted value (subject, of course, to the quality of the inputs to the prediction).

3.1.3 Asymptotic distribution of the number of cutoff calls

In answer to the second question, Theorems 7 and 9 show that the number of cutoff calls is, when suitably normalized, asymptotically normally distributed. The asymptotic variance of the number of cutoff calls [Theorem 8(b)], together with the asymptotic normality, suggests the variability to be expected in the observed (normalized) number of cutoff calls: about 63 percent of observations fall within one standard deviation of the mean, etc. Finally, the asymptotic distribution of the number of cutoff calls could be used as the basis for a statistical test for determining whether the objective is being met, although this is not accomplished in this paper.

3.2 Mathematical description of cutoff call model

The equipment will be modeled as a c-server queuing system. Calls (requests for service) arrive at the system at times $\tau_1, \tau_2, \cdots$. Denote by τ_n' the time that the nth arrival enters service. If this is a blocking system and all servers are occupied at time τ_n, the nth arrival never enters service, and for later convenience, τ_n' will be taken to be $-\infty$ in this case. Throughout Section III the arrival process may be any arbitrary point process. Each call has associated with it a (nonnegative) holding time that it wishes to spend using the resources of the system. It is assumed that a single call occupies only a single server in the system during its entire holding time (this will be important later in discussion of the organization of the failure modes). The holding times are denoted by $Y_1, Y_2, \cdots$, and are taken to be mutually independent and identically distributed, and independent of the arrival process.

So far, we have just described an ordinary queuing model. The additional feature that distinguishes the models including equipment failure is that the servers may be unreliable. That is, at certain (random) times, all the servers, or certain groups of servers, may cease serving the customers at their positions, and the affected customers will be forced to depart prematurely from the system at these times. Adopting the natural physical terminology for the mathematical model, these customers will be said to have been "cut off." Suppose that there are m different failure modes in the system. That is, there are m different ways in which various groups of servers (and possibly all servers) can fail in such a way as to cause cutoffs at the instant the failure begins. Any particular server may be affected by many failure modes, and many different configurations of failed servers may be included in a single failure mode. For example, suppose a switching system having 1200 terminations (lines and trunks) is made up of ten identical units, each serving 120 terminations. Then this system has a failure mode at 120 servers (terminations)—this would not be counted as ten separate failure modes if all these units had the same failure characteristics. With each failure mode, associate a renewal process listing the times at which failures of this type occur. These m processes will be called "failure processes." Let F^i be the distribution of the interrenewal times for the ith process, and let λ_i be the reciprocal of the mean time between renewals, $\lambda_i^{-1} = \int_0^\infty x \, dF^i(x)$. Let the epochs in the ith failure process be denoted by $S_1^i, S_2^i, \cdots$. It is assumed that these failure processes are mutually independent and independent of the arrival- and holding-time processes. The latter independence assumption is reasonable when the arrivals have no prior knowledge about the state of the system at the time of arrival.

Also associated with the ith failure mode is a number p_i between zero and one. The quantity p_i represents the probability that a call in the system will be cut off when a failure of type i occurs, and is called the *cutoff impact* of failure mode i. The severity of a failure of type i is indicated by p_i. If $p_i = 1$ then the ith failure mode is an entire system failure, and, with probability one, all calls in service are cut off when such a failure occurs. If, on the other hand, p_i is close to zero, then this describes a minor failure, and fewer calls will be cut off when such a failure occurs. We will take $p_i \neq 0$ for every i since a failure mode with cutoff impact zero can be ignored.

3.3 Correspondence with physical situation

Imagine a call using the resources of some telephone system (for definiteness, say a switching system), in either the setup phase or the conversation (stable) phase. Many elements of the system are used to provide and maintain the conversation path that is the electrical connection from one side (incoming or originating) of the system to the other (outgoing or terminating). Failure of some of these elements may cause the call to be dropped from the system without an on-hook by either party. In the queuing model, it is not these elements that are thought of as the servers. Rather, a single call is thought of as occupying a single server, such as a pair of terminations or a path through a system, which may be subject to being disabled by the failure of some of these elements. From this point of view, any particular server may be affected by several failure modes.

3.4 Probabilistic interpretation

Before turning to the computation of the probability that a call that has been accepted by the system will be cut off, the following probabilistic heuristics are offered as an aid to clarifying the idea of the model.

The event that a call in the system is cut off can be conceptualized as a realization of a competition process. Suppose a call having holding time Y enters the system at time t. At the entrance time t, m clocks are set running, with the ith clock's running time having the distribution of the excess lifetime of the time between failures for the ith failure process at time t. If the holding time Y expires before any of the clocks run down, no failures occur and, hence, no cutoff can occur. If one of the clocks runs down first (say the jth one), a biased coin ($P\{\text{heads}\} = p_j$) is tossed. If the coin comes up heads, the call is cut off, and the experiment stops for this call. If the coin comes up tails, the call is not cut off, and the experiment continues, with the jth clock now running according to the distribution F^j. For this call, the experiment stops either when it has been cut off or when it departs normally from the system.

The computation, which is performed in the next section, follows this description by first determining the probability of no cutoff and then subtracting from one.

3.5 Probability of cutoff

With this section, we begin following the outline of Section 3.1. The sequence of results and their proofs is simply a mathematical translation of the description given in Section 3.4. Lemma 1, while of independent interest, is used here only in establishing the main result of this section, which is Theorem 2.

Lemma 1: *Let* $\{N(t)\colon t \geq 0\}$ *be a renewal counting process with interrenewal time distribution F. Then for* t, $y \geq 0$ *and* $k \geq 1$, *the probability that there are k renewals in the interval* $[t, t + y]$ *is given by*

$$\int_0^t g_k(t - s, y)dM_0(s), \tag{1}$$

where

$$g_k(u, y) = \int_u^{u+y} [F_{k-1}(u + y - x) - F_k(u + y - x)]dF(x), \qquad (2)$$

with F_k the k-fold convolution of F with itself, F_0 equals V, the standard right-continuous unit step function with jump at the origin, and M_0 the augmented renewal function for the process. For $k = 0$, the probability that there are no renewals in this interval is given by $\int_0^t [1 - F(t + y - s)]dM_0(s)$.

Theorem 2: Let $\bar{M}_0^i$ be the augmented renewal function for the defective distribution $(1 - p_i)F^i$,

$$\bar{M}_0^i(x) = \sum_{k=0}^{\infty} (1 - p_i)^k F_k^i(x), \qquad (3)$$

and let

$$\bar{g}_i(u, y) = 1 - F^i(u) - p_i \int_u^{u+y} \bar{M}_0^i(u + y - x)dF^i(x). \qquad (4)$$

Then the probability that a call entering the system at time t is cut off is given by

$$1 - \int_0^{\infty} \left[\prod_{i=1}^{m} \int_0^t \bar{g}_i(t - s, y)d\bar{M}_0^i(s)\right]dH(y). \qquad (5)$$

In the limit as t approaches infinity, this becomes

$$1 - \int_0^{\infty} \prod_{i=1}^{m} \left[1 - \lambda_i p_i \int_0^{\infty} \bar{g}_i(u, y)du\right]dH(y). \qquad (6)$$

If the arrival process is independent of the remaining queuing and failure processes, the probability that the nth call will be cut off, given that it enters the system, can be computed by integrating eq. (5) against the distribution of τ_n'. In case all the failure processes are stationary Poisson processes, the probability of cutoff is constant and does not depend on the entrance time of the call.

Corollary 3: Suppose $F^i(x) = 1 - e^{-\lambda_i x}$ for $i = 1, \cdots, m$. Then every call in the system has probability of cutoff given by

$$1 - \int_0^{\infty} \exp\left(-\sum_{i=1}^{m} \lambda_i p_i y\right)dH(y). \qquad (7)$$

If, in addition, the call-holding-time distribution is exponential, $H(y) = 1 - e^{-\nu y}$, the probability of cutoff reduces to

$$\theta = \frac{\sum_{i=1}^{m} \lambda_i p_i}{\nu + \sum_{i=1}^{m} \lambda_i p_i}. \qquad (8)$$

These are obtained by appropriate substitution in eq. (5).

3.6 Discussion

The probability that a call already in the system will be cut off has been computed for a queuing system with unreliable servers. The arrival process and queue discipline may be arbitrary; this is a reflection of the fact that the event of cutoff depends only on what happens after the call enters the system. The limiting argument used to establish eq. (6) can be carried out even if the arrival process depends on the service time process (as in systems with state-dependent arrival rates), although the probability that the nth call will be cut off is more difficult to compute in this case. We have assumed the service times are independent and identically distributed. This could be relaxed, but for most ordinary message telephone service applications it does not seem necessary to introduce this complication. As can be seen from eq. (7), great simplification results if it can be assumed that the failure processes are stationary Poisson processes. In practice, this assumption has often been used because, in studying large systems from a great distance, data that would enable one to characterize the failure processes in the system in more detail are often not available. When the conditions that obtain in the physical situation are difficult to identify exactly, it may not be possible to determine the information needed to make successful application of a more general model.

IV. A MARKOV MODEL AND SOME LIMIT LAWS

4.1 Introduction

In Section IV we deal, for a more specific queuing system, with the second two items in the outline in Section 3.1. There are many ways to particularize the general considerations discussed in Section III, depending on the underlying queuing model. For purposes of estimation of cutoff call rates in telephone systems, certainly it is desirable to allow the most general model possible. This might be a transient analysis of a queue in which, in addition to the exogenous arrivals, there may be feedback and retrials by rejected and cutoff customers, and general service and interfailure times. Unfortunately, analytic treatment of such a complicated model is not within reach. The asymptotic analysis of such general queues, even with perfectly reliable servers, is accomplished only approximately in many cases.

Here, instead, we will study about the simplest of stochastic models for this situation, the M/M/c/c queue with stationary Poisson failure processes. This decision results from informal consideration of the tradeoff between realism of description on one hand and possibility of successful execution of analysis on the other. Even in this simple case, there are many interesting difficulties. For example, solving numerically the Chapman-Kolmogorov equations (Appendix A) for the invariant distribution of the embedded chain (Section 4.3) is likely to be easier than obtaining qualitative insight through analytic solution of these equations. No representation is hereby made that the Markovian assumptions are particularly accurate in representing reality, or that the asymptotic results obtained well describe transient behavior. Nevertheless, the assumptions are not such gross distortions of the physical situation that they render such models useless, and the study of simpler models has several important virtues to recommend it. Solutions can be obtained, the general features of the underlying situation remain visible without the technical details that sometimes obscure the main ideas, directions for the generalizations that are likely to be successful on more complicated models are suggested, and, last but not least, results can be checked against data to determine if more general models are required. The Markovian model to be described has been successfully used in the switching systems area, and predictions made from it have shown reasonable agreement with data. This is not to say that further refinements of these models would not be valuable. Such refinements would be interesting and useful advances in the state of the art.

4.2 Specifications and notation

In the M/M/c blocking system, let α be the arrival rate, ν be the service rate, and let $\{A(t) : t \geq 0\}$ denote the arrival process. The m failure processes are all stationary Poisson processes with rates $\lambda_1, \cdots, \lambda_m$, all positive. (In the example in Section 3.2, the failure rate for the 120-termination *failure mode* would be ten times the failure rate of a single 120-termination *unit*.) The system will be assumed to recover instantaneously from failures, so that the only effect that a failure has is to cause some of the calls in the system to depart prematurely, before the completion of their intended holding times. Failures, therefore, have no effect on calls that are not already in the system. For example, they do not cause an increase in the blocking probability of the system. Clearly this is only an approximation to the true situation, but it seems to produce acceptable results, for several reasons. First, in practical cases, the ratio of average outage time to mean time between failures is usually small; here this small number has been replaced by zero. Secondly, in this approximation the total number of cutoff calls tends to be overestimated because more calls are accepted into the system than would be if the failure durations were positive. This means that more calls are exposed to the possibility of being cut off. Again, if the times between failures are long compared to the outage times, the cutoff call rate (number of cutoff calls divided by number of arrivals or number of accepted calls) will not be badly distorted by this approximation.

The failure processes interact with the queuing processes in the following way. Let $B(t)$ denote the number of busy servers at time t, $t \geq 0$, including the effects of failures (as below), and let $C(t, r)$ be the number of busy servers at time t in an ordinary (no server failures) M/M/c/c system when there are r in the system at time 0. Then, whenever a failure of type i occurs, the probability that a call in the system will be cut off is p_i, and the cutting-off events for each of the calls in the system at that time are assumed to be mutually independent, as are the cutting-off events corresponding to different failure times. (Simultaneous failures occur with probability zero since the distributions of the interfailure times are all continuous.) This models a situation in which the calls in service at any time are more or less regularly spread out over the servers in the system, and all parts of the system subject to a given failure mode are equally vulnerable. The independence, on the other hand, is invoked to reflect the fact that this regular distribution obtains only perhaps in a very broad, average way, and at any given failure epoch, the server occupancy might be quite irregular. At each epoch in each failure process, then, the number of calls cut off is a binomial random variable with parameters given by the number of busy servers at that epoch and the cutoff impact of that failure mode. That is, at time S_n^i, if $B(S_n^i) = k$, the number of calls cut off is binomially distributed with parameters k and p_i. Sometimes many of the calls in the system will be carried by the unit (group of servers) experiencing the failure; sometimes proportionately fewer calls will be carried on this unit. The binomial model provides an approximate description of this situation. This is a compromise between a very detailed model that keeps track of individual server busy and idle times and the individual identities and times of failure of server groups, and a deterministic model having the number of cutoffs at S_n^i equal to $p_i B(S_n^i)$, which is unrealistic for being too regular.

4.3 The embedded Markov chain

As defined, $B(t)$ is a pure jump process; even with cutoffs caused by failures accounted for, all sample paths can be assumed to be continuous from the right. Pool the failure processes and denote the resulting stationary Poisson process by $\{S_1, S_2, \cdots\}$. Define $B_n = B(S_n^-)$ $(n = 1, 2, \cdots)$; B_n is the number of busy servers just before the nth failure of any kind. The sequence $\{B_n: n = 1, 2, \cdots\}$ is a Markov chain, called the *embedded chain*, with state space equal to $\{0, 1, \cdots, c\}$. The survivors in the system at time S_n^+ have the same exponentially distributed service times as new arrivals do, and their number is determined only from B_n. The number of arrivals in $[S_n, S_{n+1}]$ is independent of the number of arrivals before S_n. Note that the strong Markov property is not required of the arrival process, for while the failure epochs are random times, they are not determined by the arrival process because of the assumed independence.

4.4 Properties of the embedded chain

Let W_n denote the number of calls cut off by the failure that occurs at time S_n. Then, for each n, the conditional distribution of W_n, given B_n, is a mixture of binomials:

$$P\{W_n = w \mid B_n = b\} = \frac{1}{\lambda} \sum_{i=1}^{m} \lambda_i \binom{b}{w} p_i^w (1 - p_i)^{b-w},$$

$$b = 0, \cdots, c;\ w = 0, \cdots, b. \quad (9)$$

Here λ_i/λ is the probability that the nth event in the pooled process comes from failure process i $(\lambda = \lambda_1 + \cdots + \lambda_m)$. Denote the right-hand side of eq. (9) by q_{bw}.

Finally, note that the W_n's are conditionally independent, given the B_n's, because of the independence of the cutting-off events corresponding to different failure times. That is,

$$P\{W_{i_1} = w_1, \cdots, W_{i_n} = w_n \mid B_{i_1} = b_1, \cdots, B_{i_n} = b_n\}$$

$$= \prod_{k=1}^{n} P\{W_{i_k} = w_k \mid B_{i_k} = b_k\} \quad (10)$$

for all positive integers $n, i_1, \cdots, i_n$.

The properties of the B_n-process can be most readily obtained from the fundamental representation

$$B_{n+1} = C(S_{n+1} - S_n, B_n - W_n),$$

where the equality is equality in distribution. That is, the number of busy servers at (just before) S_{n+1} has the same distribution as the number of busy servers in an ordinary (no server failures) M/M/c/c system running for time $S_{n+1} - S_n$ with $B_n - W_n$ (the number of survivors in the system at time S_n^+) calls in the system at time zero. It has already been observed that $\{B_n: n = 1, 2, \cdots\}$ is a Markov chain; straightforward conditioning arguments and appeal to the independence of the failure and queuing processes establish that its transition probabilities are given by

$$P\{B_{n+1} = j \mid B_n = i\}$$

$$= \sum_{k=0}^{i} \sum_{r=1}^{m} \lambda_r \binom{i}{k} p_r^{i-k}(1-p_r)^k \int_0^\infty P\{C(x,k) = j\} e^{-\lambda x} dx. \quad (11)$$

These are independent of n, so the chain has stationary transition probabilities. Denote them by p_{ij}. We remark that if $p_r = 1$ for every r, these reduce to

$$p_{ij} = \int_0^\infty P\{C(x,0) = j\} \lambda e^{-\lambda x} dx,$$

so that $\{B_n\}$ are mutually independent in this case. Also, if the failure processes are not stationary Poisson, but are, say, renewal, then the p_{ij} are still well-defined, although they take a different form. In particular, they then depend on n, and while $\{B_n\}$ is still a Markov process, it does not have stationary transition probabilities. Some of the following results (particularly those about recurrence) continue to hold in this case, but limit laws are harder to obtain.

Riordan gives the distribution of $C(x, k)$:[2]

$$P\{C(x,k) = j\} = \frac{\rho^j}{j!} \left(\sum_{i=0}^{c} \frac{\rho^i}{i!} \right)^{-1}$$

$$+ \frac{c!}{j!} \rho^{c-k} \sum_{i=1}^{c} \frac{D_k(r_i) D_j(r_i)}{r_i D_c(r_i) D_c'(r_i + 1)} e^{r_i \nu x}, \quad (12)$$

where $\rho = \alpha/\nu$, the D_n are related to the Poisson-Charlier polynomials c_n (Ref. 3) by $D_n(s) = \rho^n c_n(-s)$, and $r_1, \cdots, r_c$ are the roots of $D_c(s + 1)$. These roots are all real and negative so that the $e^{r_i \nu x}$ all vanish as $x \to \infty$, and the $P\{C(x, k) = j\}$ approach the well-known Erlang equilibrium probabilities, independent of k. Equation (12) shows that $P\{C(x, k) = j\}$ is an analytic function of x that is not identically zero, so that its zeros, if any, are isolated. Thus, there is a set of positive measure in $[0, \infty[$ on which $P\{C(x, k) = j\} > 0$. This means that $\int_0^\infty P\{C(x, k) = j\} \exp(-\lambda x) dx$ is positive for every j and k, and so $p_{ij} > 0$ for every i and j. This positivity shows the $\{B_n\}$ chain to be irreducible and aperiodic. Since the chain is finite, all states are positive recurrent (Ref. 4, Section I.XV.6).

4.5 The induced Markov chain

The two-dimensional process $\{(B_n, W_n) : n = 1, 2, \cdots\}$ is again a Markov chain whose transition probabilities are given by $r_{s_i s_j} = q_{b_j w_j} p_{b_i b_j}$, where $s_i = (b_i, w_i)$. That is,

$$P\{(B_{n+1}, W_{n+1}) = s_j \mid (B_n, W_n) = s_i\} = r_{s_i s_j} = q_{b_j w_j} p_{b_i b_j}.$$

Use is made here of eq. (10). This chain will be called the *induced chain*.

It is desirable for the induced chain to inherit the properties of the embedded chain discussed in Section 4.4. To obtain this, it would be sufficient to have $q_{ij} > 0$ for all i and j. From eq. (9), this is satisfied, unless $p_i = 1$ for every i. The case $p_i = 1$ for every i is a trivial special case of what is to follow, because then $W_n = B_n$ with probability one, for every n. Also, for large systems with many failure modes, this case is of little interest. For these reasons, we will suppose that there is at least one i for which $p_i < 1$. Under this condition, $r_{s_i s_j} > 0$ for every i and j, and since the induced chain is also finite (its state space is $\{(b, w): b = 0, 1, \cdots, c, w = 0, 1, \cdots, b\}$), it is irreducible, aperiodic, and positive recurrent, just as the embedded chain was.

4.6 Stationarity

Since service objectives represent long term goals for system operation, it is appropriate to compare the equilibrium features of the model against the service objectives.

Since both the embedded and induced chains are positive recurrent, they are both ergodic. The embedded chain has an invariant distribution $\{u_k: k = 0, \cdots, c\}$ given by

$$u_k = \lim_{n\to\infty} p_{ik}^{(n)},$$

independent of i. As usual, the parenthesized superscript indicates the n-step transition probability. Furthermore, $u_k > 0$ for each k, $\sum_{k=0}^{c} u_k = 1$, and $u_k = \sum_{i=0}^{c} u_i p_{ik}$ (Ref. 4, Section I.XV.7). To say that the system has been in operation for a long time can be expressed by taking $\{u_0, \cdots, u_c\}$ to be the distribution of the number of busy servers at time zero. With this choice of initial distribution, $\{B_n\}$ becomes a strictly stationary process.

The induced chain also has an invariant distribution, denoted by $\{v_{(0,0)}, \cdots, v_{(c,c)}\}$. It is easy to see that $v_{(b,w)}$ is given by $v_{(b,w)} = q_{bw}u_b$, $b = 0, \cdots, c$; $w = 0, \cdots, b$. The induced chain can also be made strictly stationary by taking its initial distribution to be its invariant distribution.

4.7 A strong law of large numbers

The quantity of basic interest in this study is the cumulative number of cutoff calls, $\chi_n = W_1 + \cdots + W_n$. This section is devoted to describing a strong law of large numbers for χ_n and some of its ramifications. This addresses the second item in the outline of Section 3.1.

In general, $\{W_n: n = 1, 2, \cdots\}$ is not a Markov process. However, it can be written as a functional of the induced chain. The appropriate functional to choose is π_2, the projection onto the second coordinate: $W_n = \pi_2(B_n, W_n)$ for each n. π_2 is clearly a measurable function on the σ-field of the induced chain, and so the limit theorems of Sections V.5 and V.7 of Ref. 5 may be applied to $\{W_n\}$.

Let $Z(t)$ be the number of calls accepted by the system in $[0, t]$,

$$Z(t) = \sum_{n=0}^{\infty} V(t - \tau_n'),$$

and put $Z_n = Z(S_n^-)$.

Theorem 4: *χ_n/n converges with probability one to $\theta \lim_{n\to\infty} (EZ_n/n)$.*

Corollary 5: *χ_n/Z_n converges to θ in expectation and with probability one.*

The proofs of these results can be found in Appendix B.

Now this is not quite what is required for applications. Generally, one does not count either carried calls or cutoff calls indexed by the times of failure $S_1, S_2, \cdots$. Rather, what one does is keep a running count of these items indexed by a continuous time parameter. Accordingly, let $\chi(t)$ denote the total number of calls cut off in $[0, t]$; one has

$$\chi(t) = \sum_{n=1}^{\infty} W_n V(t - S_n) = \chi_{\max\{n : S_n \le t\}}.$$

Corollary 6: *$\chi(t)/Z(t)$ converges to θ in expectation and with probability one.*

Applications of these results have been discussed in Section 2.1. In a stable Markovian environment, Corollary 6 says that the natural estimator of the probability of cutoff in the system, namely the cutoff call rate, is strongly consistent. The implication for measurement is that for systems in operation, measurements can be relied upon to estimate the underlying cutoff call rate that is characteristic of the system. The extension of these results to other than Markovian queues would provide even better approximations when the environment can be more precisely specified. The implication for system design is that once it is configured with certain failure modes, etc., its cutoff call rate, in the appropriate environment, will be as predicted, subject to sets of probability zero and the quality of the failure rate predictions.

Before turning to central limit theorems, a partial indication of the rate of approach to steady state will be given.[6] For this purpose, assume that $c = \infty$ (so that all arriving calls immediately enter service) and that $p_i = 1$ for all i (so that every time a failure occurs, all calls in the system are cut off). Then it can be shown that

$$E\left(\frac{\chi(t)}{Z(t)}\right) = E\left(\frac{\chi(t)}{A(t)}\right) = \frac{\lambda}{\lambda + \nu}(1 - e^{-\lambda t})\left[1 - \frac{1 - e^{-(\lambda+\nu)t}}{(\lambda + \nu)t}\right]. \qquad (13)$$

Eq. (13) can be used to estimate relative errors after different times. Let $R(t) = \left(\frac{\lambda}{\lambda + \nu}\right)^{-1}\left[\frac{\lambda}{\lambda + \nu} - E\left(\frac{\chi(t)}{A(t)}\right)\right]$; then $100R(t)$ is the percentage error in $E\left(\frac{\chi(t)}{A(t)}\right)$ as an estimate of θ after t time units. Using eq. (13),

$$R(t) = e^{-\lambda t} + \frac{1}{(\lambda + \nu)t}(1 - e^{-\lambda t})(1 - e^{-(\lambda+\nu)t}). \qquad (14)$$

Measuring time in minutes, with $\lambda = 0.003$ (about three failures per day) and $\nu = 0.166$ (six-minute average call holding time), the percentage errors, from eq. (14), are 85 after one hour, 35 after six hours, 12 after 12 hours, and 2 after 24 hours.

4.8 A central limit theorem

The existence of an asymptotic normal approximation for the cumulative number of cutoff calls makes the construction of statistical tests easier. In this section, we discuss these approximations in discrete and continuous time. This addresses the third item in the outline of Section 3.1.

The central limit theorem for χ_n follows directly from the central limit theorem for functionals defined on a Markov chain, for example, see Theorem V.7.5 in Ref. 5.

Theorem 7: *There are positive numbers μ and σ for which*

$$\lim_{n\to\infty} P\left\{\frac{\chi_n - \mu n}{\sigma\sqrt{n}} \le x\right\} = \Phi(x),$$

where $\Phi(x)$ is the standard normal integral.

This requires little discussion: the condition (D_0) and the moment condition of theorem V.7.5 of Ref. 5 are satisfied because the induced chain is finite and positive recurrent. The interesting results are the values of the centering and scale parameters. It is easy to see that

$$\mu = EW_1 = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i EB_1 = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i \sum_{b=0}^{c} b u_b = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i \sum_{b=0}^{c} \frac{b}{m_{bb}}, \qquad (15)$$

where m_{ab} is the mean first passage time from state a to state b in the embedded chain. (If $a = b$, this is a mean recurrence time).

Theorem 8(*a*): *The asymptotic variance of the partial sums of the B_k's is*

$$\lim_{n\to\infty}\frac{1}{n}\operatorname{Var}\left(\sum_{k=1}^{n} B_k\right) = \sum_{a=0}^{c}\sum_{b=0}^{c}\frac{ab}{m_{aa}m_{bb}}\left(\frac{m_{bb}^{(2)}}{m_{bb}} - 2m_{ab}\right) + \sum_{b=0}^{c}\frac{b^2}{m_{bb}}, \tag{16}$$

where $m_{bb}^{(2)}$ is the second moment of the recurrence time for state b in the embedded chain.

Theorem 8(*b*): *The scale constant in the central limit theorem is*

$$\sigma^2 = \sum_{i=1}^{m}\frac{\lambda_i}{\lambda}p_i(1-p_i)\sum_{b=0}^{c}\frac{b}{m_{bb}} + \sum_{i=1}^{m}\frac{\lambda_i}{\lambda}p_i^2\sum_{b=0}^{c}\frac{b^2}{m_{bb}}$$
$$+\left(\sum_{i=1}^{m}\frac{\lambda_i}{\lambda}p_i\right)^2\sum_{a=0}^{c}\sum_{b=0}^{c}\frac{ab}{m_{aa}m_{bb}}\left(\frac{m_{bb}^{(2)}}{m_{bb}} - 2m_{ab}\right). \tag{17}$$

We have written the centering and scale parameters in terms of the moments of the recurrence times for the embedded chain. These can be found by solving for the invariant distribution of the embedded chain (Appendix A). The mean recurrence times are then just the reciprocals of the elements of the invariant distribution, and the second moments can be obtained from the first moments by using Theorem I.11.7 of Ref. 7. The mean first passage times m_{ij} can be found by solving another system of linear equations, for example see Theorem 6–7A of Ref. 8. For even moderate values of c, it appears that the wisest thing to do in applications is to solve the system of eqs. (25) numerically. The single-server case is treated explicitly in the next section, and it can be seen that even in this case, the computations are extensive.

In continuous time, the central limit theorem looks slightly different. This is because counting the number of cutoffs according to the number of failures, rather than over time, introduces a random time transformation with scale λ.

Theorem 9: *The distribution of the normalized cumulative number of cutoff calls over time,*

$$\frac{\chi(t) - \lambda\mu t}{\sigma\sqrt{\lambda t}}, \tag{18}$$

converges weakly to the cumulative distribution function (cdf) of a normal random variable having mean zero and variance $1 + \mu^2/\sigma^2$.

V. THE SINGLE-SERVER SYSTEM

In this section we discuss in detail the results of the previous sections as they apply to the single-server system. We will explicitly solve for the invariant distributions and, thereby, be able to represent the parameters of the limit laws in terms of the arrival, service, and failure rates.

If there is only a single server, we will suppose that there is only one failure mode, of rate λ and cutoff impact p. Certainly if all failures are complete failures, $p = 1$. We can allow $p < 1$ to account for malfunctions which may only sometimes cut off calls. Other failure modes with other severities could be allowed. Solving the Chapman-Kolmogorov system of eqs. (25) and making use of Theorem 10 we obtain

$$u_0 = r_0 = \frac{p\lambda + \nu}{p\lambda + \nu + \alpha}, \quad u_1 = r_1 = \frac{\alpha}{p\lambda + \nu + \alpha}. \tag{19}$$

From eqs. (11) and (12) we obtain the transition probabilities

$$\begin{aligned} p_{00} &= \frac{\nu + \lambda}{\lambda + \nu + \alpha}, \\ p_{01} &= \frac{\alpha}{\lambda + \nu + \alpha}, \\ p_{10} &= \frac{\nu + p\lambda}{\lambda + \nu + \alpha}, \\ p_{11} &= \frac{(1-p)\lambda + \alpha}{\lambda + \nu + \alpha}. \end{aligned} \tag{20}$$

The mean first passage and recurrence times can then be obtained as indicated in Section 4.8:

$$\begin{aligned} m_{00} &= \frac{p\lambda + \nu + \alpha}{p\lambda + \nu}, \\ m_{01} &= \frac{\lambda + \nu + \alpha}{\alpha}, \\ m_{10} &= \frac{\lambda + \nu + \alpha}{p\lambda + \nu}, \\ m_{11} &= \frac{p\lambda + \nu + \alpha}{\alpha}. \end{aligned} \tag{21}$$

Let σ_{11}^2 be the variance of the recurrence time for state 1. By using Theorem I.11.7 of Ref. 7, we obtain

$$\sigma_{11}^2 = \frac{(p\lambda + \nu)((2-p)\lambda + \nu + \alpha)}{\alpha^2}. \tag{22}$$

Since σ^2 from Theorem 8(b) reduces to $p\sigma_{11}^2/m_{11}^3$ in case $c = 1$, we obtain

$$\sigma^2 = \frac{p\alpha(p\lambda + \nu)((2-p)\lambda + \nu + \alpha)}{(p\lambda + \nu + \alpha)^3}. \tag{23}$$

This is the scale constant for Theorem 7. The centering constant is

$$\mu = \frac{p\alpha}{p\lambda + \nu + \alpha}.$$

VI. APPLICATIONS

These results have been applied at Bell Laboratories to predict the cutoff call performance of certain toll switching systems, and to evaluate reliability objectives for these systems on the basis of a determination of whether the cutoff call objective for the system can be met if these reliability objectives are followed.

In one such example, a system terminating 22,000 trunks was considered. Thirteen failure modes that were significant for cutoff calls in the system were identified. Table I lists, for each failure mode, the size of the unit failing or the number of terminations affected by the failure, the failure rate expressed as a mean number of failures per year, and the cutoff impact. For most of the failure modes, there was more than one type of unit or subsystem of the given size. The failure rates of all the units or subsystems of a single size were added together to obtain the failure rate for that failure mode. This is done because we are going to assume uniform distribution of calls over terminations, as discussed in Section 4.2. If more precise information on the distribution of calls over terminations or location of failed units is available, it may be more reasonable not to pool, but to carry individual information, as appropriate.

Every stable call in the system must occupy two terminations, one incoming and one outgoing. For a particular call, the failed unit or subsystem may be on the incoming side of the switch, the outgoing side, or both, or neither. Then the estimation of the cutoff impact of a failure mode is like a problem in sampling without replacement in which one counts the number of paths through the switch that contain the failed unit or subsystem. If the total number of terminations on the switch is N and the number of terminations affected by a failure of type i is n_i, then the cutoff impact for failure mode i is

Table I—Failure modes, frequencies, and cutoff impacts for example in Section VI

Failure Mode	Terminations Affected	Failures per Year	Cutoff Impact
1	22,000	0.248	1.0
2	5,500	0.195	0.438
3	4,080	0.077	0.337
4	2,040	0.0004	0.177
5	1,920	0.355	0.167
6	840	0.482	0.075
7	512	10.819	0.046
8	128	66.667	0.012
9	120	0.263	0.011
10	32	22.727	0.003
11	16	20.0	0.0015
12	8	217.391	0.0007
13	1	1030.0	0.0001

$$p_i = \frac{n_i(2N - n_i - 1)}{N(N-1)}. \tag{24}$$

Using eq. (8) we find that in a Markovian environment, the probability that a call entering the system will be cut off because of one of these failures is 0.24×10^{-4}, when the mean call-holding time is six minutes. Based on this, it was concluded that a sufficient margin of safety existed to ensure that the system's cutoff call objective would be met, even after allowing for possible errors in the specification of failure modes and rates, and other possibilities that could not be accounted for in the analysis.

VII. ACKNOWLEDGMENT

I would like to thank J. J. Kulzer and R. F. Serfozo for helpful discussions. I would also like to thank N. A. Marlow, who showed me how to obtain Theorem 8(a), and the referee, who suggested some simplifications in Theorem 4 and its corollaries.

APPENDIX A

The Invariant Distributions in Discrete and Continuous Time

As pointed out in Section 4.8, the centering and scale constants for the strong law and the central limit theorem are all written in terms of the mean first passage and recurrence times for the $\{B_n\}$ process. It appears from eqs. (11) and (12) that use of theorems I.7.1 and I.6.1 of Ref. 7 to find the invariant distribution of $\{B_n\}$ will require significant effort. In this appendix, we will derive the Chapman-Kolmogorov equations for the $\{B(t)\}$ process. Finding the invariant distribution of the $\{B(t)\}$ process by solving these equations is easier than solving for the invariant distribution of the discrete-time process using the transition probabilities in eq. (11). It is also a more attractive procedure numerically, because the matrix of coefficients is upper triangular with only a single nonzero subdiagonal, consisting of all α's. Finally, these results are tied together by Theorem 10, which indicates that these two invariant distributions are identical.

Let $r_n(t) = P\{B(t) = n\}$. Then for $h \geq 0$, we can write $r_n(t+h) = \sum_{k=0}^{c} P\{B(t+h) = n \mid B(t) = k, S_j \notin [t, t+h], \forall j\} P\{B(t) = k, S_j \notin [t, t+h], \forall j\} + \sum_{k=0}^{c} P\{B(t+h) = n \mid B(t) = k, S_j \in [t, t+h], \exists j\}$ $P\{B(t) = k, S_j \in [t, t+h], \exists j\}$.
To simplify the following display, in the first sum, all terms involving both an arrival and a departure in $[t, t+h]$ have an h^2 in them, and so can be left off. Similarly, in the second sum, because of the λh that will appear in front, all terms involving either an arrival or a departure can be left off. We obtain, omitting terms $o(h)$ or higher,

$$r_0(t+h) = (1-\lambda h)(1-\alpha h)[r_0(t) + \nu h r_1(t)] + \lambda h \sum_{k=0}^{c} q_{kk} r_k(t),$$

$$r_n(t+h) = (1-\lambda h)(1-\alpha h)[(1-n\nu h)r_n(t) + (n+1)\nu h r_{n+1}(t)]$$
$$+ (1-\lambda h)\alpha h r_{n-1}(t) + \lambda h \sum_{k=n}^{c} q_{k,k-n} r_k(t), \ 1 \le n \le c-1,$$

$$r_c(t+h) = (1-\lambda h)[(1-c\nu h)r_c(t) + \alpha h r_{c-1}(t)] + \lambda h(1-c\nu h)q_{c0} r_c(t).$$

Collecting terms, simplifying, dividing by h, and letting $h \to 0$, we obtain

$$r_0'(t) = -\alpha r_0(t) + \nu r_1(t) - \lambda[r_0(t) - \sum_{k=0}^{c} q_{kk} r_k(t)],$$

$$r_n'(t) = -(\alpha + n\nu) r_n(t) + \alpha r_{n-1}(t) + (n+1)\nu r_{n+1}(t)$$
$$- \lambda[r_n(t) - \sum_{k=n}^{c} q_{k,k-n} r_k(t)], \ 1 \le n \le c-1,$$

$$r_c'(t) = -c\nu r_c(t) + \alpha r_{c-1}(t) - \lambda[r_c(t) - q_{c0}\, r_c(t)].$$

In equilibrium, we look for solutions with $r_j = \lim_{t\to\infty} P\{B(t) = j\}$ and $\lim_{t\to\infty} r_j'(t) = 0$. Then these equations become

$$0 = -(\alpha+\lambda) r_0 + \nu r_1 + \lambda \sum_{k=0}^{c} q_{kk} r_k$$

$$0 = \alpha r_{n-1} - (\alpha + n\nu + \lambda) r_n + (n+1)\nu r_{n+1}$$
$$+ \lambda \sum_{k=n}^{c} q_{k,k-n} r_k, \ 1 \le n \le c-1 \tag{25}$$

$$0 = \alpha r_{c-1} - (c\nu + \lambda - \lambda q_{c0}) r_c$$

$$1 = \sum_{k=0}^{c} r_k,$$

where the condition that $\{r_0, \cdots, r_c\}$ be a probability distribution has been added. These are the equations used to solve for the invariant distribution of the continuous-time process. Writing $\rho = \alpha/\nu$, $\lambda' = \lambda/\nu$, and $\mathbf{r} = (r_0, \cdots, r_c)^T$, the first $c+1$ equations can be written in matrix form as

$$[M(\rho) + \lambda' Q]\mathbf{r} = 0,$$

where $M(\rho)$ is the standard matrix for the M/M/c/c birth-death process (Ref. 9, Section 2.1), and

$$Q = \begin{pmatrix} 0 & q_{11} & q_{22} & \cdot & \cdot & q_{cc} \\ 0 & q_{10}-1 & q_{21} & \cdot & \cdot & q_{c,c-1} \\ 0 & 0 & q_{20}-1 & \cdot & \cdot & q_{c,c-2} \\ \cdot & \cdot & 0 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 0 & \cdot & q_{c1} \\ 0 & 0 & 0 & 0 & 0 & q_{c0}-1 \end{pmatrix}.$$

The equations in this form show clearly that when $\lambda = 0$, we recover the ordinary M/M/c/c system, as expected. The $M(\rho)$ matrix is tridiagonal and Q is upper triangular, leading to the attractive form for numerical work mentioned above.

It remains to show that the two invariant distributions, for continuous time and for discrete time, are identical.

Theorem 10: $r_j = u_j, j = 0, \cdots, c$.

Proof. Define $B^*(t) = \sum_{n=1}^{\infty} B_n I(S_n \le t < S_{n+1})$, where I denotes the indicator function. Since $\{S_n\}$ is a Poisson process, $B^*(t)$ is a Markov process which will be thought of as a semi-Markov process embedded in the continuous-time busy server process. The distribution of the time between transitions in this process is exponential(λ), regardless of the starting state, and so the expected time to the next transition, starting from state i, is $1/\lambda$ for every i. From Section 6.3(ii) of Ref. 10 we obtain that $\lim_{t\to\infty} P\{B^*(t) = j \mid B^*(0) = i\} = u_j$ for $j = 0, \cdots, c$. Next, the distribution of the time from an arbitrary epoch back to the most recent failure is also exponential(λ), so that using Section 6.3(iv) of Ref. 10, we obtain

$$r_j = \sum_{i=0}^{c} u_i \int_0^{\infty} P\{B(S_n + t) = j \mid B_n = i\}\lambda e^{-\lambda t} dt,$$

regardless of the value of n because of the stationarity of $\{B_n\}$. For t with $S_n + t < S_{n+1}$, one gets $B(S_n + t) = j$ by having k survivors in the system at time S_n^+ and letting the M/M/c/c system evolve from there ($k = 0, 1, \cdots, i$). This has probability $\sum_{k=0}^{i} q_{i,i-k} P\{C(t, k) = j\}$, so

$$r_j = \sum_{i=0}^{c} u_i \sum_{k=0}^{i} q_{i,i-k} \int_0^{\infty} P\{C(t, k) = j\}\lambda e^{-\lambda t} dt = \sum_{i=0}^{c} u_i p_{ij} = u_j. \quad \blacksquare$$

APPENDIX B

Proofs

In this appendix, we provide proofs for Lemma 1, Theorems 2 and 4, Corollaries 5 and 6, and Theorems 8 and 9. The blot symbol ■ signifies the end of a proof.

Proof of Lemma 1: For $t, y > 0$ and $k \ge 2$, begin by writing

$$P\{N(t+y)-N(t)=k\} = \sum_{j=0}^{\infty} P\{N(t+y)=k+j,\, N(t)=j\}$$

$$= \sum_{j=0}^{\infty} P\{S_{k+j} \le t+y < S_{k+j+1},\, S_j \le t < S_{j+1}\},$$

where the interrenewal times are $X_1, X_2, \cdots, S_n = X_1 + \cdots + X_n$, and $S_0 = 0$. Now condition on $S_j = s$, $X_{j+1} = x$, and $X_{j+2} + \cdots + X_{j+k} = u$. Using the independence and identical distribution of the interrenewal times, together with some algebraic simplification, leads to eqs. (1) and (2). The sum and integral can be interchanged because of the uniform convergence of the renewal function on compact intervals. The proof is similar for the cases $k = 0$ and 1. ■

Proof of Theorem 2: Let $N_i(t)$ be the renewal counting process for failure mode i, and let T_n stand for the event that the nth arriving call is accepted into the system and survives to the end of its intended holding time without being cut off. Then there is a version of $P\{T_n | Y_n = y, \tau_n' = t\}$ that is given by

$$\sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} P\{T_n | N_i(t+y) - N_i(t) = k_i,\, i = 1, \cdots, m\}$$

$$\cdot P\{N_i(t+y) - N_i(t) = k_i,\, i = 1, \cdots, m\}$$

$$= \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} \prod_{i=1}^{m} (1-p_i)^{k_i} P\{N_i(t+y) - N_i(t) = k_i\}$$

$$= \sum_{k_1=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} \prod_{i=1}^{m} (1-p_i)^{k_i} \int_0^t g_{k_i}^i(t-s, y)\, dM_0^i(s)$$

$$= \prod_{i=1}^{m} \sum_{k=0}^{\infty} (1-p_i)^{k} \int_0^t g_{k}^i(t-s, y)\, dM_0^i(s),$$

where the superscript i on the g indicates the function from eq. (2) which belongs to failure mode i. Now insert the expression for g_k^i from Lemma 1, simplify, and use eq. (4). This leads to the desired conditional probability's being given by

$$\prod_{i=1}^{m} \int_0^t \bar{g}_i(t-s, y)\, dM_0^i(s).$$

Equation (5) is then obtained by unconditioning on the holding-time distribution and subtracting from one to obtain the probability of cutoff. To obtain eq. (6), first observe that since $\bar{g}_i$ is directly Riemann integrable, the basic renewal theorem (Ref. 4, Section II.XI.1) applies, yielding

$$\lim_{t\to\infty}\int_0^t \bar{g}_i(t-s,y)dM_0^i(s) = \lambda_i\int_0^\infty \bar{g}_i(u,y)du$$

$$= 1 - \lambda_i p_i \int_0^\infty \int_u^{u+y} \bar{M}_0^i(u+y-x)dF^i(x)du.$$

The Lebesgue bounded convergence theorem then allows the interchange of limit with the integrals in eq. (5), yielding eq. (6). ■

Proof of Theorem 4: The existence of the a.s. limit as $n \to \infty$ of χ_n/n follows from standard results about regenerative processes. These results, in a Markov chain setting (e.g., Theorem I.15.2 of Ref. 7), show that χ_n/n converges w.p. 1 to

$$\sum_{b=0}^{c}\sum_{w=0}^{b} wv_{(b,w)},$$

which, upon reversing order of summation, is seen to be equal to EW_1 since $\{W_n\}$ is a stationary process. Note that one also has $EW_1 = \lim_{n\to\infty} E\chi_n/n$. To complete the proof, straightforward calculations show that $E\chi_n = \theta EZ_n$, and that $\lim_{n\to\infty} EZ_n/n$ exists. ■

Proof of Corollary 5: Write $\chi_n/Z_n = (\chi_n/n)(n/Z_n)$ to obtain the result. ■

Proof of Corollary 6: Use theorem 8.1 of Ref. 12. ■

Proof of Theorem 8(*a*): Let $V_j(n)$ denote the number of visits to state j in the first n transitions of the embedded chain. Then

$$\sum_{k=1}^{n} B_k = \sum_{j=0}^{c} jV_j(n),$$

and it follows that

$$B_1 = \sum_{j=0}^{c} jV_j(1) \quad \text{and} \quad B_k = \sum_{j=0}^{c} j[V_j(k) - V_j(k-1)], \quad k \geq 2. \tag{26}$$

Using Lemma 7.3 of Ref. 5 and the stationarity of the embedded chain, our first step is

$$\lim_{n\to\infty}\frac{1}{n}\operatorname{Var}\left(\sum_{k=1}^{n} B_k\right) = \operatorname{Var} B_1 + 2\sum_{k=2}^{\infty}[EB_1B_k - (EB_1)^2]. \tag{27}$$

The variance of B_1 is easily seen to be

$$\operatorname{Var} B_1 = \sum_{b=0}^{c}\frac{b^2}{m_{bb}} - \sum_{a=0}^{c}\sum_{b=0}^{c}\frac{ab}{m_{aa}m_{bb}}. \tag{28}$$

For the second term, use the representation in eq. (26), exchange order of summation, and sum by parts to obtain

$$\sum_{k=2}^{\infty}[EB_1B_k - (EB_1)^2]$$

$$= \sum_{i=0}^{c} \sum_{j=0}^{c} ij \lim_{R\to\infty} \left(E[V_i(1)V_j(R) - V_i(1)V_j(1)] - \frac{R-1}{m_{ii}m_{jj}} \right). \tag{29}$$

To simplify this, observe that $EV_i(1)V_j(1) = EV_i(1)^2$ when $j = i$ and is zero otherwise. Also, $P\{V_i(1) = x\} = 1 - u_i$ for $x = 0$, it equals u_i for $x = 1$, and is zero for $x \geq 2$, so that $EV_i(1)^2 = u_i = 1/m_{ii}$. Equation (29) becomes

$$\sum_{i=0}^{c} \sum_{j=0}^{c} ij \lim_{R\to\infty} \left[EV_i(1)V_j(R) - \frac{R-1}{m_{ii}m_{jj}} \right] - \sum_{i=0}^{c} \frac{i^2}{m_{ii}}. \tag{30}$$

Further simplifying, observe that

$$\begin{aligned} EV_i(1)V_j(R) &= E(V_j(R) \mid V_i(1) = 1)P\{V_i(1) = 1\} \\ &= E(V_j(R) \mid B_1 = i)u_i, \end{aligned}$$

so that the limit to be evaluated in eq. (30) becomes, after factoring out the common term $1/m_{ii}$,

$$\lim_{R\to\infty} \left[E(V_j(R) \mid B_1 = i) - \frac{R-1}{m_{jj}} \right]. \tag{31}$$

Now, letting I stand for the indicator function, $V_j(R) = \sum_{n=1}^{R} I(B_n = j)$, so that $E(V_j(R) \mid B_1 = i) = \sum_{n=0}^{R-1} p_{ij}^{(n)}$. When $j = i$ we obtain immediately, using Theorem I.6.5 of Ref. 7, that the limit in eq. (31) is given by

$$\frac{m_{ii}^{(2)} + m_{ii}}{2m_{ii}^2}.$$

When $j \neq i$, add and subtract $\sum_{n=0}^{R-1} p_{jj}^{(n)}$ in eq. (31), and use Theorem I.11.4 together with Theorem I.6.5 of Ref. 7 to obtain that the limit in eq. (31) is given, in this case, by

$$\frac{m_{jj}^{(2)} + m_{jj}}{2m_{jj}^2} - \frac{m_{ij}}{m_{jj}}.$$

We obtain, finally,

$$2 \sum_{k=2}^{\infty} [EB_1B_k - (EB_1)^2] = \sum_{i=0}^{c} \sum_{j=0}^{c} \frac{ij}{m_{ii}} \left(\frac{m_{jj}^{(2)} + m_{jj}}{m_{jj}^2} - \frac{2m_{ij}}{m_{jj}} \right). \tag{32}$$

Combining eqs. (27), (28), and (32) yields eq. (16), as was to be proved. ■

Proof of Theorem 8(*b*): From Theorem 7.5 of Ref. 5, the scale constant for the central limit theorem for the induced chain is the asymptotic variance of χ_n. We have

$$\operatorname{Var} \chi_n = E \sum_{k=1}^{n} W_k^2 + 2E \sum_{k=1}^{n} \sum_{j<k} W_jW_k - \left(E \sum_{k=1}^{n} W_k \right)^2.$$

The last term on the right is equal to

$$-\left(\sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i\right)^2 \left(E \sum_{k=1}^{n} B_k\right)^2.$$

Using the conditional independence (eq. (10)), one shows that

$$EW_jW_k = \left(\sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i\right)^2 EB_jB_k \quad \text{for} \quad j \neq k,$$

and using eq. (9), one shows that

$$EW_k^2 = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i(1 - p_i)EB_k + \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i^2 EB_k^2.$$

Combining these and simplifying leads to

$$\frac{1}{n} \operatorname{Var} \chi_n = \sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i(1 - p_i) \sum_{b=0}^{c} \frac{b}{m_{bb}}$$
$$+ \left[\sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i^2 - \left(\sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i\right)^2\right] \sum_{b=0}^{c} \frac{b^2}{m_{bb}} + \frac{1}{n} \left(\sum_{i=1}^{m} \frac{\lambda_i}{\lambda} p_i\right)^2 \operatorname{Var}\left(\sum_{k=1}^{n} B_k\right).$$

The remainder of the proof consists in using Theorem 8(a) followed by algebraic manipulation. ■

Proof of Theorem 9: Begin by writing

$$\frac{\chi_n - \mu\lambda S_n}{\sigma\sqrt{n}} = \frac{\chi_n - \mu n}{\sigma\sqrt{n}} - \frac{\mu}{\sigma} \frac{S_n - n/\lambda}{\sqrt{n}/\lambda}. \tag{33}$$

By Theorem 7, the distribution of the first term on the right converges to the standard normal cdf. The distribution of the second term converges to the cdf of a normal random variable having mean zero and variance μ^2/σ^2. We will show that for each n, these two terms are independent.

The stochastic process $B^*(t)$ defined in the proof of Theorem 10 is a Markov pure jump process, and $X_1 = S_1$ and B_1 are independent because of the independence of the failure process and the arrival and service time processes (or use Theorem 15.28 of Ref. 11). Since the σ-field of the W's is contained in that of the B's, $S_1 = X_1$ and $\chi_1 = W_1$ are independent too. By Proposition 15.27 of Ref. 11, S_1 is a Markov time for the process, so that the process $B_1^*(t) = B^*(t + S_1)$ for $t \geq 0$ is a Markov process whose initial distribution is $P\{B_1 = b\}$, $b = 0, \cdots, c$. But because of the stationarity of $\{B_n\}$, $P\{B_1 = b\} = u_b$, $b = 0, \cdots, c$, so that $B_1^*(t)$ and $B^*(t)$ are equivalent processes. Hence, X_2 and B_2 are independent, and so are X_2 and W_2, from which it follows that S_2 and χ_2 are independent. The result for S_n and χ_n follows by induction.

It follows that the limit of the distribution of the quantity in eq. (33)

is the cdf of a normal random variable having mean zero and variance $1 + \mu^2/\sigma^2$. Now apply Theorem 8.1 of Ref. 12 to obtain the final result. The sufficient condition of that theorem is satisfied, because, using the notation of Ref. 12, $M^*(n) \leq W_{n+1} + \mu\lambda X_{n+1}$ with probability one. ■

REFERENCES

1. L. J. Forys and E. Messerli, "Analysis of Trunk Groups Containing Short-Holding-Time Trunks," B.S.T.J., *54,* No. 6 (July–August 1975), pp. 1127–53.
2. J. Riordan, *Stochastic Service Systems,* New York: John Wiley, 1962.
3. D. L. Jagerman, "Some Properties of the Erlang Loss Function," B.S.T.J., *53,* No. 3 (March 1974), pp. 525–51.
4. W. Feller, *An Introduction to Probability Theory and Its Applications,* Vols. I and II, New York: John Wiley, 1950.
5. J. L. Doob, *Stochastic Processes,* New York: John Wiley, 1953.
6. N. A. Marlow, private communication.
7. K. L. Chung, *Markov Chains with Stationary Transition Probabilities,* New York: Springer Verlag, 1967.
8. E. Parzen, *Stochastic Processes,* San Francisco: Holden-Day, 1962.
9. L. Kosten, *Stochastic Theory of Service Systems,* Oxford: Pergamon Press, 1973.
10. D. Gross and C. M. Harris, *Fundamentals of Queueing Theory,* New York: John Wiley, 1974.
11. L. Breiman, *Probability,* Reading, Mass.: Addison-Wesley, 1968.
12. R. F. Serfozo, "Functional Limit Theorems for Stochastic Processes Based on Embedded Processes," Adv. Appl. Prob., 7 (1975), pp. 123–9.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

A New Approach to High-Capacity Digital Mobile Radio

By P. S. HENRY and B. S. GLANCE

(Manuscript received March 19, 1981)

Space diversity (adaptive phased-array antennas) is an effective weapon against the cochannel interference encountered in cellular mobile radio systems. High-order diversity, and hence, strong interference suppression, can be achieved with modest hardware complexity by using time-division retransmission. With this technique, which is especially well-suited to digital modulation methods, the adaptive signal processing required for space diversity can be performed at just one end of the communication link, namely, the base station. At the other end (the mobile unit) only a single-element antenna is needed. Moreover, the use of coherent phase-shift keying in such a system allows simple RF circuity, because the adaptive processing is done at baseband. In the context of cellular mobile radio, the combination of space diversity, time-division retransmission and 120-degree corner illumination of each cell can yield a reliable communication channel even in the presence of intercell interference, Rayleigh fading (both flat and frequency-selective), and shadow fading. The use of these techniques allows approximately 130 two-way channels per cell (at 32 kb/s each) to be accommodated in the 40-MHz bandwidth of the 850-MHz mobile radio band.

I. INTRODUCTION

We present an outline for the design of a cellular digital mobile radio system suitable for telephone service in urban areas. This system serves two purposes: (*i*) it demonstrates that a digital system with a capacity comparable to that of existing analog designs[1,2] is a realistic possibility, and (*ii*) it provides a framework for taking advantage of future advances in digital signal processing, especially speech coding. In this paper, we assume that adequate speech quality can be achieved at 32 kb/s using adaptive-differential-pulse-code modulation

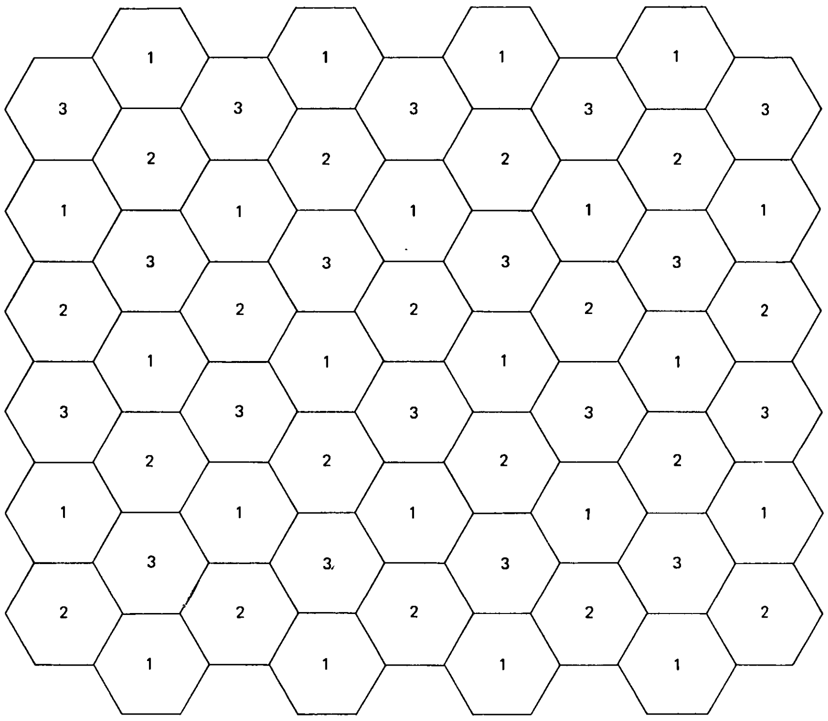


Fig. 1—A hexagonal-cell layout for a mobile radio system. The number in each cell refers to the channel set assigned to it.

(ADPCM).* Over the next few years economical coders in the 10 to 16 kb/s range should become available.[5]

We consider a service area covered with hexagonal cells with radius (center-to-corner) typically one mile, as shown in Fig. 1.[6] A communication link between a base station and a mobile is established on an assigned channel (frequency band) chosen from the channel set available to that cell. To get high capacity, the same channel set may be re-used in several cells, provided they are widely enough separated so that the mutual cochannel interference is tolerable. The intervening cells, of course, must use different frequencies. To accomplish this, the total bandwidth used by the system is divided into several channel sets, each with an equal number of channels. Each cell is assigned one set (indicated by the numbers in the cells of Fig. 1) according to a plan that maximizes the distance between re-uses of any given set. The greater the number of channel sets, the greater the distance (and hence isolation) between cochannel cells. On the other hand, a large number

* In addition to ADPCM, adaptive-delta modulation at 40 kb/s is an attractive candidate for reduced-bit-rate voice transmission.[3,4]

of sets means relatively few channels per cell and, hence, low system capacity. We will see that one of the strengths of the system described below is that it requires the use of just three channel sets; analog systems use 7–15.[1,2]

In the following discussion, we use elementary models to characterize the phenomena which limit mobile radio communication. We then describe hardware for dealing with these phenomena, and calculate system performance. Our investigation is simplified by two assumptions:

(*i*) Cochannel interference is the sole source of additive signal degradation. (Thermal noise is negligible.)

(*ii*) The interference at any point in the system, being the incoherent sum of contributions from many interferers, is equivalent to stationary Gaussian noise. In effect, we are assuming that the shadow and Rayleigh fading (Section II) of the total interference is negligible compared with the fading of the signal.[7,8]

II. AVAILABLE SIGNAL-TO-INTERFERENCE RATIO

Mobile radio reception in urban areas is characterized by large fluctuations in received signal power P as a mobile travels along a street. This variability can be modeled as the product of three factors:[8]

$$P(\mathbf{r}) = |\mathbf{r}|^{-n} \cdot S(\mathbf{r}) \cdot R^2(\mathbf{r}), \tag{1}$$

where $\mathbf{r}$ is the position vector denoting the location of the mobile relative to the base station. The first factor on the right represents the general reduction in signal strength as a mobile recedes from the base station. In free space, of course, $n = 2$, but in an urban environment, n is in the range of 3 to 4.

The second factor, $S(\mathbf{r})$, represents shadow fading, which is primarily the result of blockage because of large objects, such as buildings and hills. Measurements of S in several cities indicate that it is approximately a log-normal random variable: values of S measured in dB display a normal distribution with mean value zero and standard deviation σ in the range of 6 to 10 dB.

The third factor in eq. 1 represents Rayleigh fading, a phenomenon caused by the random addition of signals arriving at an antenna via multiple paths. The amplitude of the received envelope, R, may be modeled as a random variable with a probability density function

$$p(R) = 2R exp(-R^2). \tag{2}$$

The mean-squared value of R, corresponding to average signal power, is unity. In general, R varies with both vehicle location and signal frequency. For the time being, we neglect the frequency dependence.

A detailed view of a group of cells from Fig. 1 is shown in Fig. 2. To reduce cochannel interference, each cell is covered by three base stations located on alternate cell corners.[9] At any time, only one of these stations (usually the one receiving the strongest signal) serves a given mobile.

We estimate radio system performance by calculating worst-case signal-to-interference ratios (SIRs) for base-to-mobile (B→M) and mobile-to-base (M→B) transmission. The average SIR is defined to be the ratio of signal power to total interference power, based on an $|\mathbf{r}|^{-n}$ propagation law and averaged over shadow and Rayleigh fading. In the B→M direction, the worst-case SIR occurs when: (*i*) the desired mobile is in a cell corner between two base stations, and (*ii*) every cochannel cell is served by a base station whose antenna pattern covers this mobile. The resulting average SIRs for $n = 3$ and $n = 4$ propagation laws are

$$\left.\begin{array}{l}\mathrm{SIR}(n = 3) \approx 8 \text{ dB} \\ \mathrm{SIR}(n = 4) \approx 13.5 \text{ dB}\end{array}\right\} \text{B}\rightarrow\text{M}. \tag{3}$$

In the M→B direction, the worst case occurs when the desired mobile is in a cell corner between two base stations and the interfering mobiles are as close as possible to the base station being interfered with. The average SIRs in this case are

$$\left.\begin{array}{l}\mathrm{SIR}(n = 3) \approx 7.5 \text{ dB} \\ \mathrm{SIR}(n = 4) \approx 12.5 \text{ dB}\end{array}\right\} \text{M}\rightarrow\text{B}. \tag{4}$$

The corner-excited cells of Fig. 2 are effective in reducing the

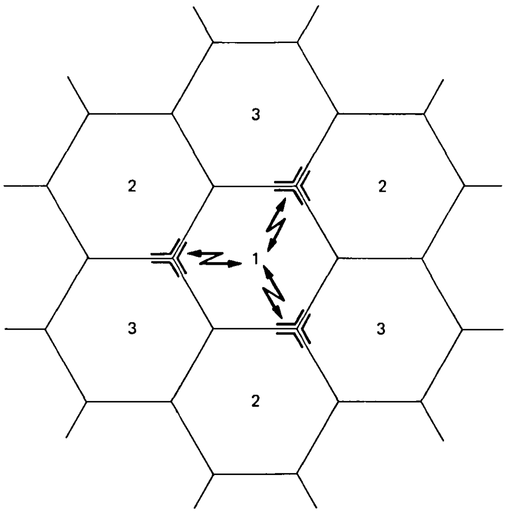


Fig. 2—Corner-excited cells. Each cell is served by three base stations equipped with 120-degree sectoral antennas.

performance degradation caused by shadow fading, because it is feasible to switch from one base station to another as shadowing conditions vary. Since the shadow fading on the paths to the three base stations is uncorrelated, the probability of a simultaneous outage on all three paths is far less than the probability of an outage on any one of them.[10] The performance of a corner-excited cell in the presence of shadow fading (σ = 8 dB) may be summarized as follows: over the entire area of the cell, the worst-case SIRs of eqs. 3 and 4 are exceeded at least 90 percent of the time. Stated differently, if a mobile communication system is able to operate satisfactorily with the SIRs of eqs. 3 and 4, then shadow fading will cause less than 10 percent of the cell area to have unsatisfactory service. (We consider 90 percent coverage to be a reasonable service objective.)[9] We, therefore, take the SIR values in eqs. 3 and 4 to be reasonable estimates of the available SIR in our radio system.

III. SPACE DIVERSITY

In a Rayleigh-fading environment the performance of conventional digital radio systems with SIR $<$ 10 dB (eqs. 3 and 4) is very poor; binary coherent phase shift keying (CPSK), for example, has an error probability greater than 10^{-2}. The use of space diversity, however, greatly improves the situation and allows acceptable error rates ($\leq 10^{-3}$) to be attained.[7] In a space-diversity system, multiple antennas are used and the independently fading signals received on each branch are combined coherently. This process gives two benefits: it increases the output SIR because the signal contributions from the branches are added in phase, while the interference components are added randomly, and it smooths out the fluctuations in the output signal because all branches are unlikely to fade simultaneously. With binary CPSK and optimal (maximal-ratio) diversity,[7,11] the SIR at each branch required to achieve 10^{-3} error rate is 11, 7, and 4 dB for 2-, 3- and 4-branch systems, respectively. Comparing these values with eqs. 3 and 4, we see that 3-branch diversity is theoretically adequate for $n = 3$ propagation and 2-branch for $n = 4$. To allow reasonable margins for nonideal equipment, and also to simplify B$\rightarrow$M transmission (See Section V), we will assume 4-branch diversity for $n = 3$, and 3-branch for $n = 4$.

In conventional space diversity systems, arrays of multiple antennas are required at both the base station and mobile. However, a technique known as time-division retransmission[12] allows the advantages of space diversity to be obtained with an array only at the base station and just a single antenna at the mobile. In such a system, all the adaptive signal processing is done at the base station where its cost can be shared among many users. The equipment on the mobile is kept simple.

Communication with time-division retransmission requires time-sharing of a single channel by both directions of transmission. In the M→B direction, the antennas at the base station are cophased to achieve SIR enhancement; during B→M transmission, the excitation of each base station antenna is adjusted so that the separate contributions all arrive in phase at the mobile.

The operation of time-division retransmission can be understood from Fig. 3. Let the signals arriving at the base station be $\cos(\omega_c t + \theta_1)$ and $\cos(\omega_c t + \theta_2)$, where ω_c is the carrier frequency and θ_1 and θ_2 are phases measured relative to some arbitrary reference. If these signals are phase-shifted by $-\theta_1$ and $-\theta_2$, they will be brought to a common phase; simple addition at this point will be equivalent to coherent combining. For transmission back to the mobile, each antenna is excited using the conjugate (negative) of the received phase. These excitation phases exactly compensate for the different phase

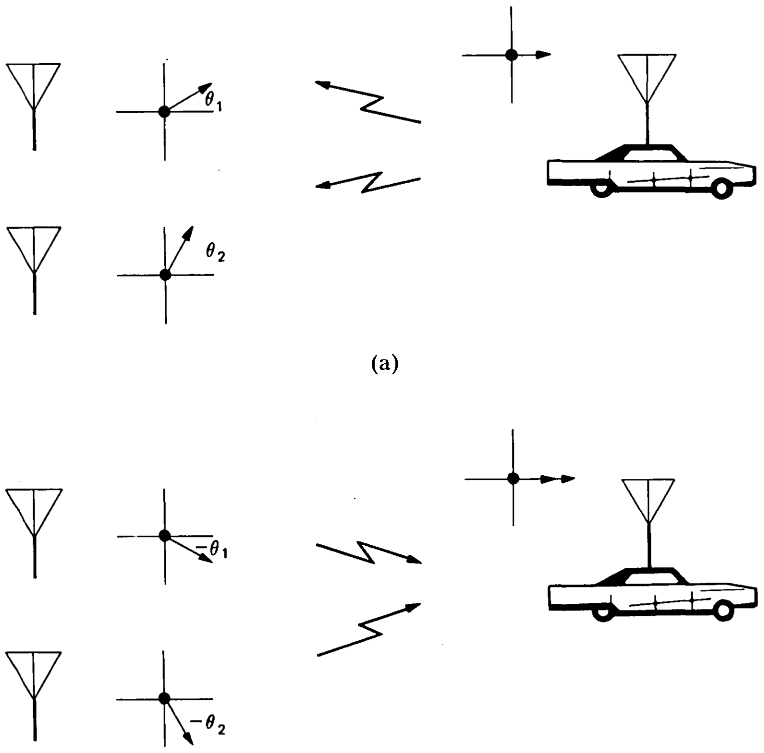


Fig. 3—Time-division retransmission. (a) Signals received from the mobile are coherently combined at the base station. (b) Signals transmitted from the base station are phased so as to interfere constructively at the mobile.

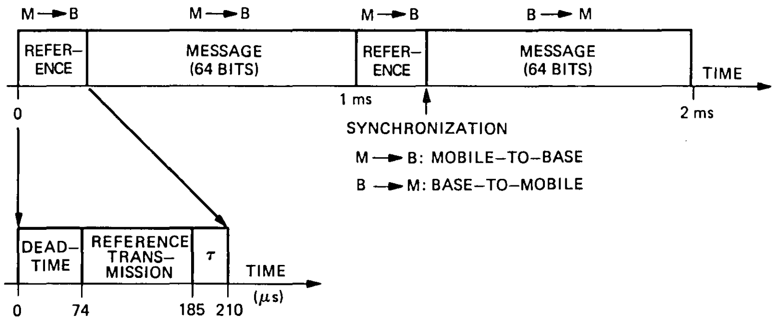


Fig. 4—A signaling frame for time-division retransmission. Both directions of transmission time-share a single channel.

delays experienced by the radiated signals, so that they all add up in phase at the mobile.

In addition to adjusting the phases of the base station antenna branches, the ideal combiner also adjusts their weights. For equal power Gaussian interference at each branch, the best net SIR is achieved with a maximal-ratio combiner, in which each branch contributes at its output a signal amplitude proportional to its received signal power.

From the foregoing description, it is apparent that there are two fundamental operations that must be performed by the base station receiver: identification of the desired signal, and coherent combination of the antenna branches. In Section IV, we describe hardware for implementing these functions.

IV. BASE STATION RECEIVER

The signaling frame for a 32-kb/s, 2-way voice channel is shown in Fig. 4. The basic cycle time of 2 ms consists of 2 message intervals, when speech is transmitted, and 2 reference intervals, during which the base station diversity combiner is updated. The 1-ms repetition period for reference transmission was chosen to be rapid enough to ensure that propagation conditions remain relatively constant during the message interval. (See Section VI.) To achieve 32 kb/s in each direction, 64 bits must be sent in each 790-μs message interval, implying a symbol rate of 81 kbaud. Depending on the details of pulse-shaping and filtering, this rate requires 80–120 kHz of bandwidth with binary phase shift keying (PSK) modulation. Thus, the number of channels per cell that can be served in the 40-MHz bandwidth of the 850-MHz mobile radio band is approximately 133 (40 MHz/3·100 kHz).

All the base stations are assumed to be synchronized at the instant indicated in Fig. 4. (A synchronization accuracy of ±1 μs can be achieved with pseudo-noise transponder techniques or direct time broadcast by satellite.[13,14]) A mobile establishes its timing from signals received from its base station; mobiles are not in synchronism with each other because of the distance-dependent propagation time across the cell.

The signal-processing circuitry for one branch of the base station receiver is shown in Fig. 5. We will see that when the demodulated signals from each branch are added as shown in the figure, the result is equivalent to maximal ratio combining. The operation of the receiver (somewhat oversimplified for the time being) is as follows. Within the

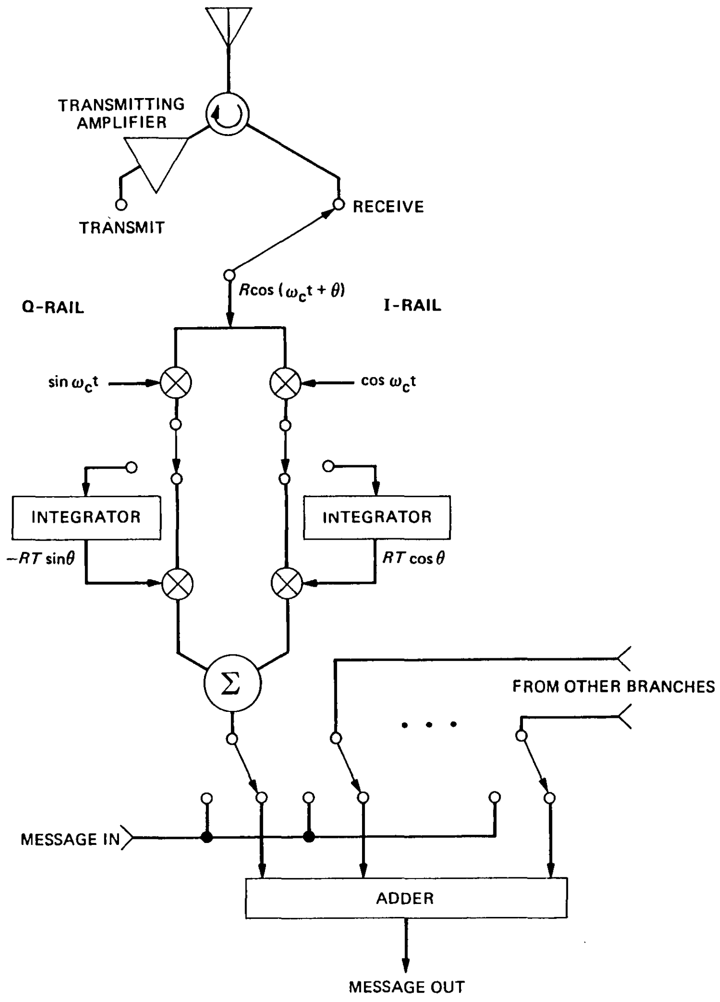


Fig. 5—Base station diversity receiver. The addition of the outputs from the several branches produces optimal (maximal-ratio) combining. The switches are in a position to receive the message-interval transmission from the mobile.

reference transmission interval the carrier burst of duration T (111 μs in Fig. 4) received at an antenna has the form $R\cos(\omega_c t + \theta)$, where R is the Rayleigh amplitude and θ an unknown phase. (Both R and θ, though functions of time, are essentially constant during the reference interval.) After down-conversion the signals on the I and Q rails are $R\cos\theta$ and $-R\sin\theta$, respectively. These signals are integrated to produce reference coefficients $TR\cos\theta$ and $-TR\sin\theta$, which will be used for subsequent message demodulation. During message transmission to the base station, the kth message bit and accompanying interference may be written as

$$a_k R\cos(\omega_c t + \theta) + I_c\cos\omega_c t + I_s\sin\omega_c t,$$

where $a_k = \pm 1$ represents the transmitted bit, and I_c and I_s are Gaussian variables with zero mean and variance s^2. After down-conversion and multiplication by the previously determined reference coefficients, the I and Q signals become

$$a_k TR^2\cos^2\theta + I_c TR\cos\theta$$

and

$$a_k TR^2\sin^2\theta - I_s TR\sin\theta.$$

The sum of these terms gives a demodulated signal $a_k TR^2$ and a mean-square "noise" $T^2R^2s^2$, resulting in a s/n of R^2/s^2; this is precisely the performance that would have been achieved with conventional coherent demodulation. The output signals from all branches are in phase (independent of θ), and each has a magnitude proportional to R^2, the received signal power; thus, the simple addition of these outputs is equivalent to maximal-ratio combination. Observe that the adaptive signal processing is all performed at baseband, where it is amenable to digital implementation. Processing at RF is minimal.

The successful operation of the base station receiver requires a clean reference signal from the mobile, relatively uncontaminated by interference. The primary effect of such interference is to cause suboptimal combining of the various antenna branches, leading to reduced signal output power. This reduction is tolerable (less than a few tenths of a dB) for reference SIR greater than about 14 dB. However, if all mobiles were to transmit reference bursts at their carrier frequencies, the reference SIR (eq. 4) would be less than 14 dB and, hence, unacceptably low. This problem is solved using a frequency-offset reference transmission scheme based on the seven-cell cluster shown in Fig. 6. Each cell in a cluster is assigned an offset frequency, designated by the seven subscripts "a" through "g," which is a multiple (0, ± 1, ± 2, ± 3) of a low frequency $\Omega = 2\pi/T$, where T is the duration of the reference-signal transmission.

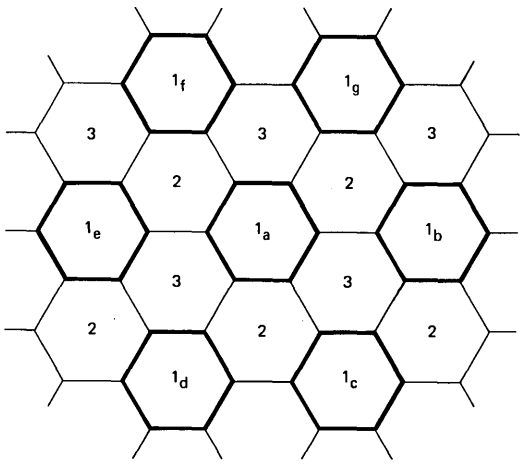


Fig. 6—A seven-cell cluster for offset-frequency reference transmission. The use of orthogonal reference signals allows interference from the first ring of interferers (subscripts "*b*" through "*g*") to be completely suppressed.

During the reference interval, the transmitting frequency of a mobile is shifted from the carrier frequency ω_c by the offset assigned to that cell. At the base station, the local oscillator is shifted by the same amount, and the reference coefficient is generated by the integrator as described earlier. We will show that the reference coefficient so obtained is the same as if no offset had been used at the mobile or base station, provided the fading on the M→B path is flat (nonfrequency-selective).

The use of different reference frequencies by mobiles sharing the same channel allows the base station to select the desired signal and suppress the interference. In effect, the re-use factor for reference transmission is 21, even though it is only 3 for message transmission. The choice of $\Omega = 2\pi/T$ allows the various reference signals to be orthogonal; unwanted signals do not contribute to the integrator output. In the present system, $T \approx 111$ μs ($\Omega = 2\pi \cdot 9$kHz), which is a compromise between excessive bandwidth (T small) and excessive time allocated to reference transmission (T large).

The reference signal may be generated using a single-sideband modulator to shift the carrier frequency by the desired offset. To obtain offsets at the mobile and base station with the required phase relationship (see below), the offsets can be generated from the appropriate harmonic (9, 18, 27) of the 1-kHz reference clock which controls the timing of the reference bursts.

We now show that in a flat-fading environment, the reference coefficient is independent of the offset frequency. Flat fading means that the envelope delay (the derivative of phase with respect to frequency) is independent of frequency. Let the reference clock on the mobile be cos $\omega_r t$, and let its mth harmonic be used to generate the desired offset, so that the transmitted reference signal is $\cos(\omega_c t + m\omega_r t)$. The received signal (apart from a frequency-independent scale factor) is $\cos(\omega_c t + \theta + m\omega_r(t - t_d))$, where t_d is the envelope delay and θ is the unknown phase angle used in Section III. The base station reference clock, being locked to the envelope of the received signal, may be written $\cos \omega_r(t - t_d)$. The offset local oscillator is then $\cos(\omega_c t + m\omega_r(t - t_d))$. Down-conversion of the received reference signal by this local oscillator yields $\cos \theta$, independent of offset. We comment in Section VI on the degradation caused by nonflat fading.

Rejection of an interfering reference signal requires integration over its entire duration. Since interference comes from distant cells, some excess integration time must be allocated to cover the associated propagation delay. For a cell radius of 1 mile, 25 μs is adequate to allow complete integration of reference signals from the first ring of interferers in Fig. 6.

The reference scheme described above is implemented by dividing the 210-μs reference interval into three zones, as shown in Fig. 4:

(*i*) A dead time of 6 symbol intervals ($\approx$74 μs) following message transmission to let signals from distant cells "quiet down."*

(*ii*) Nine symbol periods ($\approx$111 μs) for actual reference transmission.

(*iii*) An excess integration time τ of 2 symbol periods ($\approx$25 μs).

The reference SIR in this scheme is 21 dB for $n = 3$ (inverse-cube propagation) and 28 dB for $n = 4$. These values are comfortably above the 14-dB requirement mentioned earlier.

V. BASE-TO-MOBILE TRANSMISSION

For transmission back to the mobile, the circuit shown in Fig. 5 is used with the signal flow along the rails reversed. The required phase conjugation is accomplished by inverting the sign of the Q-rail reference coefficient. This procedure gives the same SIR at the mobile as if all the transmitted power were radiated from a single antenna and maximal-ratio diversity (with the same number of branches as at the base station) were used at the mobile.[12]

The receiver on the mobile can be very simple if differential phase shift keying (DPSK) is used in the B$\rightarrow$M direction. The SIR requirements for this type of modulation, though greater than those of CPSK,[15] are met by the system. For inverse-cube propagation ($n = 3$) with 4-

* In some situations, e.g., a locale with hilly terrain, a longer dead time may be necessary in order to eliminate interference from distant cells.

branch diversity, the required SIR is 7 dB, 1 dB less than available (eq. 3). For $n = 4$ with 3 branches, the requirement is 9 dB, leaving a 4.5 dB margin.

VI. IMPAIRMENTS

In the preceding discussion, we considered some fundamental obstacles to mobile radio communication and proposed a system design to deal with them. As additional impairments are considered, a more refined design emerges. Two impairments that seem particularly important will be discussed very briefly in this section: the time dependence of the reference coefficients, and the error in these coefficients caused by frequency-selective fading.

(*i*) *Time Dependence.* The reference coefficients, since they are determined at 1-ms intervals, do not precisely correspond to the propagation conditions existing during message transmission. The consequent system degradation may be estimated by modeling the reference coefficients as samples of a narrow-band Gaussian process with sample-to-sample correlation of $\rho(\tau) = J_0(\omega_c v\tau/c)$, where J_0 is the zero-order Bessel function, v is the vehicle speed and τ is the sampling interval.[16] The greatest error occurs at the end of a message interval when the reference coefficient is 1-ms "old"; the mean-squared fractional error between the "true" and "available" coefficients is $E^2 = 2(1 - \rho(\tau))$. At a carrier frequency of 850 MHz and a vehicle speed of 55 mph, $E^2 \approx 0.1$. This degradation is equivalent to an SIR during reference transmission of ~10 dB, which is unacceptably low. (See Section IV.) The problem can be largely eliminated by using a simple two-point linear predictor to estimate the reference coefficient during message transmission.[17] In this case, the mean-squared fractional error is $E^2 \approx (\omega_c v\tau/c)^4/8 \approx .005$, corresponding to an effective reference SIR of 23 dB.

(*ii*) *Frequency-Selective Fading.* When frequency-selective fading is significant, the envelope delay is no longer independent of frequency. This leads to errors in the reference coefficients determined by the frequency-offset technique (Section IV). The mean-squared error associated with this degradation is $E^2 \approx (\Delta\omega)^2 \cdot \mu_2$, where $\Delta\omega$ is the frequency offset and μ_2 is the second central moment of the path-delay distribution.[18] Let us assume that the multipath characteristics on the three links between the mobile and the corner base stations are statistically independent. In a typical urban location, the probability of finding $\sqrt{\mu_2} > 2$ μs on any link is ~0.2, so the probability of finding $\sqrt{\mu_2} > 2$ μs on all three links is less than 1 percent. (In effect, we are using base-station diversity to combat frequency-selective fading in much the same way that it is used against shadow fading. Since the outages caused by these phenomena are small and nearly uncorrelated,[19] the net system outage will be approximately the sum of the two, or 11 percent.) We, therefore, use 2 μs as a reasonable value for $\sqrt{\mu_2}$, and find a mean-squared reference error due to frequency-selective fading of $E^2 \approx (2\pi \cdot 27\ kHz \cdot 2\ \mu s)^2 \approx 0.1$, which is not acceptable. A large improvement is obtained when reference transmissions are made alternately on two frequencies, one above and one below the carrier; interpolation can then be used to estimate the desired reference coefficient. The mean-squared error depends on the product of the two frequency offsets, so the best pairings are (+27, −9), (+18, −18), (+9, −27), where the numbers denote offset frequencies in kHz. The resulting error is $E^2 \approx \frac{1}{4}(2\pi \cdot 18\ \text{kHz})^4 \cdot \mu_4$, where μ_4 is the fourth central moment of the path-delay distribution. For an exponential distribution $\mu_4 = 9\mu_2^2$, so $E^2 \approx .006$, corresponding to a reference SIR of 22 dB.[20]

The errors caused by time dependence of the reference coefficients and frequency-selective fading degrade the reference SIR computed in Section IV. Since these errors arise from independent sources, they add incoherently, and result in net reference SIR's of 17 dB and 19 dB for the $n = 3$ and $n = 4$ cases, respectively. These values are safely above the 14-dB requirement mentioned in Section IV.

VII. SUMMARY

In the preceding discussion, we developed an outline for a high-capacity cellular digital mobile radio system. To mitigate the effects of shadow fading, the plan uses three-corner excitation of each cell. Rayleigh fading and cochannel interference are combatted using space diversity; an array of 3 or 4 elements provides adequate performance. Time-division retransmission is an attractive way to implement space diversity on two-way channels; it allows all the adaptive signal processing to be performed at the base station. Moreover, the use of CPSK modulation permits this processing to be done at baseband, thereby minimizing the complexity of the RF hardware. To provide clean reference signals for the base-station diversity combiner, a frequency-offset transmission scheme is used. The impairments associated with this technique, though not negligible, are acceptably small.

REFERENCES

1. S. T. Kamata, M. Sakamato, and K. Fukuzumi, "800 MHz Band Land Mobile Telephone Radio System," Rev. Electron. Commun. Labs, *25* (November–December 1977), pp. 1157–71.
2. W. R. Young, "Advanced Mobile Phone Service: Introduction, Background and Objectives," B.S.T.J., *58*, No. 1 (January 1979), pp. 1–14.
3. J. L. Flanagan, et al., "Speech Coding," IEEE Trans. Commun., *COM-27* (April 1979), pp. 710–37.
4. R. J. Canniff, "Signal Processing in SLC-40, a 40-Channel Rural Subscriber Carrier," Conf. Rec. Int. Commun. Conf., San Francisco, June 16–18, 1975, Paper No. 40C.
5. B. G. Haskell and R. Steele, "Audio and Video Bit-Rate Reduction," Proc. IEEE, 69 (February 1981), pp. 252–62.

6. W. C. Jakes, Jr., Ed., *Microwave Mobile Communications*, New York: John Wiley, 1974, Section 7.2.
7. Y. S. Yeh and D. O. Reudink, "Efficient Spectrum Utilization for Mobile Radio Systems Using Space Diversity," IEE Conf. on Radio Spectrum Conservation Techniques, London, July 7–9, 1980.
8. W. C. Jakes, op. cit., Chapters 1 and 2.
9. V. H. MacDonald, "Advanced Mobile Phone Service: The Cellular Concept," B.S.T.J., *58*, No. 1 (January 1979), pp. 15–41.
10. W. C. Jakes, op. cit., Section 5.5.
11. W. C. Jakes, op. cit., Section 5.2.
12. W. C. Jakes, op. cit., Section 6.5.
13. R. C. Dixon, Ed., *Spread Spectrum Techniques*, New York: IEEE Press, 1976, Part VIII.
14. R. L. Easton et al., "Dissemination of Time and Frequency by Satellite," Proc. IEEE, 64 (October 1976), pp. 1482–93.
15. M. Schwartz, W. R. Bennett, and S. Stein, *Communication Systems and Techniques*, New York: McGraw Hill, 1966, Chapter 10.
16. R. H. Clarke, "A Statistical Theory of Mobile Radio Reception," B.S.T.J., *47*, No. 6 (July–August 1968), pp. 957–1000.
17. B. M. Oliver, "Efficient Coding," B.S.T.J., *31*, No. 3 (July 1952), pp. 724–50.
18. D. C. Cox and R. P. Leck, "Correlation Bandwidth and Delay Spread Multipath Propagation Statistics for 910-MHz Urban Mobile Radio Channels," IEEE Trans. Comm., *COM-23* (November 1975), pp. 1271–80.
19. D. C. Cox, "Multipath Delay Spread and Path Loss Correlation for 910-MHz Urban Mobile Radio Propagation," IEEE Trans. Veh. Tech., *VT-26* (November 1977), pp. 340–4.
20. W. C. Jakes, op. cit., Section 1.5.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Least-Squares Algorithms for Adaptive Equalizers

By M. S. MUELLER

(Manuscript received March 17, 1981)

Least-squares algorithms are the fastest converging algorithms for adaptive signal processors, such as adaptive equalizers. The Kalman, fast Kalman, and adaptive lattice algorithms using a least-squares cost function are investigated and extended to complex, fractionally spaced equalizers. It is shown that, for a typical telephone channel, these algorithms converge roughly three times as fast as the conventional stochastic-gradient technique. We analyze and compute the computational complexities and demonstrate that the fast Kalman algorithm is the most efficient in terms of overall performance.

I. INTRODUCTION

Adaptive channel equalization is a widespread technique used in most high-speed digital data modems. Generally, a transversal filter with adjustable coefficients is used as the equalizer. It can be adjusted adaptively to compensate for the undesired intersymbol interference introduced by the channel.

A large number of equalizer adjustment algorithms are conceivable, depending on the cost function. The currently prevailing technique is the so-called stochastic gradient algorithm. In the past years, three new rapidly converging algorithms were published, namely, the Kalman,[1] fast Kalman,[2] and adaptive lattice[3-7] algorithms. Here, we consider algorithms which minimize the sum-of-error-squares cost function. Because these least-squares algorithms make better use of all the past available information than the stochastic gradient algorithms, their start-up is faster.[8]

Originally the Kalman, fast Kalman, and adaptive lattice algorithms for equalizer update procedures were published for real-valued signals. In this paper, we present extensions of these algorithms to complex-valued signals which facilitate the analysis of quadrature-amplitude-modulated (QAM) data transmission formats. We also extend the algorithms to include fractionally spaced equalizers.[9] An important characteristic of each algorithm is its computational complexity which we analyze for the least-squares, as well as for the stochastic gradient algorithms. Simulation results of the equalizer start-up using least-squares adjustment algorithms are presented for fractionally and symbol-spaced equalizers. Quadrature-amplitude-modulated and real-life voice-grade transmission channels are used for this study.

II. THE LEAST-SQUARES ALGORITHMS

In this section, we describe extensions of the Kalman, fast Kalman, and adaptive lattice adjustment algorithms for the coefficient adjustment of complex, fractionally spaced equalizers. It is assumed that equalizer output values are computed once for each symbol interval T, where T denotes the time interval between successive data values in the transmitter. The fractionally spaced equalizer is assumed to operate on T/p-spaced complex samples of the received signal.

Let $\xi(n)$ denote the complex p—dimensional vector of the new signal samples entering the fractionally spaced equalizer at time nT. Denote the M dimensional complex signal vector at time nT containing all signal samples over the past N time instances ($M = Np$) by

$$x(n)^* = [\xi(n)^*, \xi(n-1)^*, \cdots \xi(n-N+1)^*].\dagger \qquad (1)$$

Then the output of the fractionally spaced equalizer is written as

$$y(n) = c(n-1)^* x(n), \qquad (2)$$

where $c(n-1)$ is the M dimensional coefficient vector which was last updated at the previous time instant $n-1$. The desired data value at this instant is $d(n)$. Therefore, an output error

$$e(n) = d(n) - y(n) \qquad (3)$$

results.

The objective of the least-squares algorithms is to determine the coefficient vector $c(n)$ which minimizes the weighted sum of all squared errors as if it were used over all the past received signal vectors, i.e., $c(n)$ minimizes

$$\sum_{k=0}^{n} \lambda^{n-k} |d(k) - c(n)^* x(k)|^2. \qquad (4)$$

Setting the derivative of eq. (4) with respect to $c(n)$ to zero yields the discrete-time, Wiener-Hopf equation

† The * in eq. (1) denotes conjugate complex scalars and conjugate complex transposed vectors (matrices).

$$A(n)c(n) = v(n), \tag{5}$$

where

$$A(n) = \sum_{k=0}^{n} \lambda^{n-k} x(n)x(n)^* + \lambda^n \delta I_{MM} = \lambda A(n-1) + x(n)x(n)^*, \tag{6}$$

and

$$v(n) = \sum_{k=0}^{n} \lambda^{n-k} d(n)^* x(n) = \lambda v(n-1) + x(n)d(n)^*. \tag{7}$$

A small positive definite matrix δI_{MM} is included to ensure positive definiteness of $A(n)$ for all n. For $\lambda = 1$, $\delta = 0$ and large n, $1/n$ A(n) is an estimate of the channel correlation matrix, $1/n$ v(n) is an estimate of the cross correlation vector between the desired and the received signal. For $\lambda = 1$, all past information is weighted equally in calculating an updated coefficient vector; for $\lambda < 1$ the past is attenuated geometrically. Consequently, the present has a larger influence on the update than the past. This is a desired feature if time-varying channels are involved.

Since eqs. (6) and (7) can be written recursively, the updated coefficient vector can be calculated recursively as follows, cf. Appendix A:

$$c(n) = c(n-1) + g(n)e(n)^*, \tag{8}$$

where $g(n)$ is the Kalman gain defined as

$$g(n) = A(n)^{-1}x(n). \tag{9}$$

The Kalman, the fast Kalman, and the adaptive lattice algorithms all minimize the same cost function.[4] The difference is in the manner and the complexity with which it is achieved.

The remaining part of this section contains a brief discussion of the Kalman and the fast Kalman algorithms. The adaptive lattice algorithm is discussed in more detail; its derivation is given in Appendix A. Emphasis is placed on the signal transformation which is performed by the lattice structure. This signal transformation permits the evaluation of equalizers of increasing order in a computationally efficient way. The three algorithms are given in Appendix B in a form suitable for numerical evaluation.

2.1 The Kalman algorithm

The Kalman algorithm makes use of the recursive definition of $A(n)$ in eq. (6) and iteratively computes and stores its inverse $A(n)^{-1}$. The equalizer coefficient vector is then updated according to eqs. (8) and (9) at each iteration.

While the Kalman algorithm assures rapid equalizer start-up, it has the disadvantage of requiring matrix operations. Therefore, the number of calculations is proportional to M^2 and grows very fast with increasing M.

2.2 *The fast Kalman algorithm*

Ljung et al.[10] succeeded in formulating an equivalent algorithm with reduced complexity, where the number of operations is proportional to M. This algorithm was applied to the adaptive equalizer in Ref. 2. The algorithm exploits the fact that only p new signal samples enter the signal vector $x(n)$, p samples are discarded, and the remaining are just shifted. This is accomplished by means of $p \times M$ dimensional forward and backward predictors for the new and discarded values. Recurrence equations for these predictors and, finally, for the Kalman gain vector can be derived based on this. At most, $p \times M$ matrices have to be iterated—the chief reason for the reduced complexity.

2.3 *The least-squares adaptive lattice algorithm*

Recently the adaptive lattice algorithm for a least-squares cost function, originally published by Morf et al.,[11] was extended to equalizer update applications by Satorius and Pack[3] and Shichor.[4] Its application to the decision feedback equalizer is reported by Shensa.[6] Here, a further extension to the complex fractionally spaced equalizer is presented. A short form of this was published by Lim and Mueller.[7]

In the adaptive lattice structure, the equalizer coefficients operate on a transformed signal vector

$$\tilde{x}(n) = L(n-1)x(n), \tag{10}$$

where the transformation matrix is a lower triangular matrix formed by the backward prediction coefficients $c_m^b(n)$ of order m, $m = 1 \cdots N-1$, i.e.,

$$L(n) = \begin{bmatrix} I & 0 \cdots 0 \\ -c_1^b(n)^* & I\,0 \cdots \\ -c_2^b(n)^* & I\,0 \cdot\cdot \\ \cdot & \cdots \\ \cdot & I\,0 \\ -c_{N-1}^b(n)^* & I \end{bmatrix}. \tag{11}$$

The backward predictor $c_m^b(n)$ of order m is a $mp \times p$ dimensional matrix satisfying

$$A(n)\begin{bmatrix} -c_m^b(n) \\ \\ I \\ 0 \\ \cdot \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ 0 \\ \epsilon^b(m, n) \\ \times \\ \cdot \\ \times \end{bmatrix}. \tag{12}$$

In eq. 12, $\epsilon^b(m, n)$ is the $p \times p$ dimensional backward prediction error residual of order m. For a detailed discussion of the backward predictor and the backward prediction error residual refer to Appendix A. In eqs. (11) and (12), I denotes a $p \times p$ dimensional identity matrix. The crosses in eq. (12) denote some unspecified elements which are of no further interest at this time. It follows from eqs. (11) and (12) that $A(n)L(n)^*$ is a lower block triangular matrix. Then it follows that $L(n)A(n)L(n)^*$ is lower block triangular, because it is the product of two lower block triangular matrices. On the other hand, $L(n)A(n)L(n)^*$ is Hermitian because $A(n)$ is Hermitian according to its definition in eq. (6). Therefore, $L(n)A(n)L(n)^*$ has to be a block diagonal matrix. Its diagonal element at the mth position is $\epsilon^b(m-1, n)$, i.e.,

$$L(n)A(n)L(n)^* = \text{diag}\,[\epsilon^b(m-1, n)]. \tag{13}$$

Hence, $L(n)$ diagonalizes $A(n)$.

We assume that all $\epsilon^b(m-1, n)$ are invertible and so we invert eq. (13). After premultiplying the result by $L(n)^*$ and postmultiplying it with $L(n)$ we obtain

$$A(n)^{-1} = L(n)^*\text{diag}\,[\epsilon^b(m-1, n)^{-1}]L(n). \tag{14}$$

Since it is desired that the adaptive lattice equalizer perform identically as the other least-squares equalizers, it follows that all the equalizer's output signals have to be equal, i.e.,

$$y(n) = c(n-1)^*x(n) = \tilde{c}(n-1)^*\tilde{x}(n). \tag{15}$$

From eqs. (10) and (15), it follows that the transformed coefficient vector $\tilde{c}(n)$ has to satisfy

$$\tilde{c}(n) = L(n)^{-*}c(n). \tag{16}$$

The matrix $L(n)^{-*}$, denotes the conjugate transposed inverse of $L(n)$. Upon substituting eqs. (5) and (14) into eq. (16), we obtain

$$\tilde{c}(n) = \text{diag}\,[\epsilon^b(m-1, n)^{-1}]L(n)v(n). \tag{17}$$

This suggests that the transformed coefficient vector is easily obtainable from the transformed correlation vector. The equalizer output of order N can be written as

$$y(n) = \sum_{m=1}^{N} z(m, n-1)^* \epsilon^b(m-1, n-1)^{-1} e^b(m-1, n), \qquad (18)$$

where we defined

$$\tilde{x}(n) = \begin{bmatrix} e^b(0, n) \\ e^b(1, n) \\ \cdot \\ \cdot \\ \cdot \\ e^b(N-1, n) \end{bmatrix}, \text{ and } L(n)v(n) = \begin{bmatrix} z(1, n) \\ z(2, n) \\ \cdot \\ \cdot \\ \cdot \\ z(N, n) \end{bmatrix}. \qquad (19)$$

Note that the elements of $\tilde{x}(n)$ are the backward prediction errors of order 0 to $N-1$. Therefore, the transformed equalizer operates on the backward prediction errors of order 0 to $N-1$.

Equations (10), (11), and (17) to (19) define the adaptive lattice equalizer as a transform of the ordinary transversal equalizer. An interesting property of this transform is due to the fact that $L(n)$ is a lower triangular matrix. This makes it possible to increase the dimension of the transformed signal vector and of the equalizer in a rather simple way, i.e., only one new p-vector is added to the vectors of order $m-1$ to form the vectors of order m. The already existing elements are unchanged. Accordingly, the equalizer output can be computed order-recursively in a very efficient way.

The time update algorithm of the lattice structure makes consequent use of the above-described order recursions. In addition to the backward predictor, the forward predictor and its error residual are iterated. Only the prediction errors and the prediction error residuals of order zero are updated in time. Then, using these elements as an anchor, the prediction errors and prediction error residuals of higher order are obtained recursively. It turns out that the predictions themselves are not needed. Finally, a time update of the elements $z(m, n)$ of the transformed vector $v(n)$ is obtained.

This scheme also makes use of all previously received data. Theoretically, its performance should be identical to the Kalman and the fast Kalman algorithms. Since there are no matrices involved, storage requirements and numbers of multiplications increase linearly with the equalizer length, though faster than in the fast Kalman algorithm.

A detailed derivation of the least-squares adaptive lattice equalizer algorithm is given in Appendix A. There, we adopt a notation which allowed us to describe the equalizer, the backward and the forward predictors as special cases of a general least-squares problem. In Appendix B, we list the adaptive lattice algorithm together with the two other least-squares algorithms in a form suitable for sequential execution.

III. COMPLEXITY

The number of multiplications (divisions are counted as multiplications) per iteration and the required precision are the dominant factors determining the complexity of real-time algorithms.

The effect of limited precision was investigated by T. L. Lim and the author in earlier work on that subject. There are no major differences between the three algorithms. With floating-point arithmetic, the requirements for the mantissa are from 11 to 12 bits for symbol-spaced equalizers and from 13 to 15 bits for $T/2$-spaced equalizers.

Table I gives the number of multiplications for the three least-squares algorithms and for the stochastic-gradient algorithm which is obtained when in eq. (8); $g(n)$, is replaced by a scalar. Results for both symbol- and $T/2$-spaced equalizers are given. The numbers for exponential weighting are included for the three least-squares algorithms.

The gradient algorithm requires the smallest number of multiplications, followed by the fast Kalman, the adaptive-lattice, and the Kalman algorithms. The gradient algorithm, of course, requires twice as many multiplications for the $T/2$ equalizer than for the symbol-spaced equalizer. For the Kalman algorithms, this factor is about four and for the adaptive-lattice, it is about five.

The fast Kalman algorithm has the lowest complexity of all least-squares algorithms; it requires about five times as many multiplications as the gradient algorithms for symbol-spaced equalizers and ten times as many for $T/2$-spaced equalizers. The adaptive-lattice algorithm requires more multiplications than the fast Kalman algorithm, especially for $T/2$-spaced equalizers. This is mainly because of the large number of matrix operations which is reflected in the large coefficient of p^3. However, it was pointed out in Refs. 3 and 4 that it offers a

Table I—Number of complex multiplications for equalizer spanning N symbol intervals with p samples per interval

		# Multiplications	$N = 31$ $p = 1$	$N = 31$ $p = 2$
Gradient		$2Np$	63	127
Kalman	$\lambda \neq 1$	$2N^2p^2 + 5Np$	2015	7998
	$\lambda = 1$	$Np(Np + 1)/2$ less than for $\lambda \neq 1$	1519	6045
Fast Kalman	$\lambda \neq 1$	$N(p^3 + 6p) + \frac{5}{3}p^3 + 2p^2 + \frac{4}{3}p$	316	1202
	$\lambda = 1$	$p(p + 1)/2$ less than for $\lambda \neq 1$	315	1199
Adaptive lattice	$\lambda \neq 1$	$N\left(\frac{13}{3}p^3 + 7p^2 + \frac{11}{3}p\right) - 4p^3 - 5p^2 - 2p$	454	2046
	$\lambda = 1$	$\left(N - \frac{1}{2}\right)p(p + 1)$ less than for $\lambda \neq 1$	393	1863

unique feature: the number of equalizer taps can be increased according to the actual need for the particular channel involved. Since for real-time applications the computational power for the longest required equalizer has to be provided, this cannot be regarded as an advantage and does not justify the considerably higher complexity for modem applications.

The Kalman algorithm requires the largest number of multiplications. Since it offers no additional features when compared with the fast Kalman algorithm, the latter is preferred for equalizer implementations.

IV. SIMULATED COMMUNICATION SYSTEM

Figure 1 shows the simulated system, where the transmitter is assumed to have a raised-cosine shaped transfer function with 12 percent excess bandwidth. Quadrature amplitude modulation with a symbol rate of 2400 baud and 2 bits per symbol is used. The carrier frequency is placed at 1700 Hz. The data symbols in the in-phase branch are taken from a binary pseudo random noise sequence (PRNS). The same sequence, shifted and reversed in time, is used in the quadrature branch.

Various channel transfer functions were considered. Figure 2 shows the transfer function of a channel which barely meets the requirements for basic conditioning of private lines. The eigenvalue spread of the autocorrelation matrix for symbol-spaced samples equals 9.8. The equivalent baseband impulse response of the combined transmitter and channel is used to generate the input data for the equalizer. Gaussian noise of specified power is added.

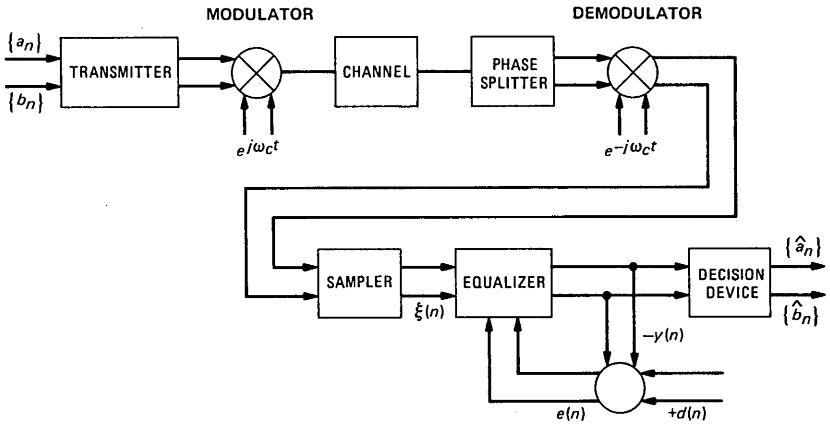


Fig. 1—Simulated transmission system.

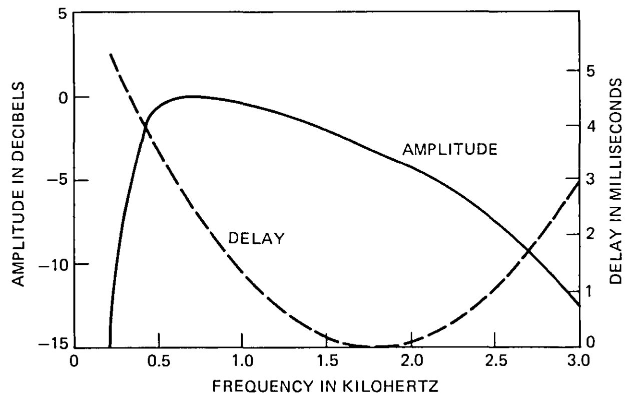


Fig. 2—Channel transfer function.

V. INITIAL CONVERGENCE

The initial convergence of the squared error at the output of the equalizer was determined for the Kalman, fast Kalman, and adaptive lattice equalizer structures. The behavior of the gradient algorithm, as well as of a fixed transversal filter with optimal coefficients were simulated for comparison purposes. Single precision floating-point arithmetic is used throughout, i.e., the mantissa is represented by 24 bits. The s/n is 25 dB and all equalizer coefficients are initially set to zero. A PRNS with a period of 127 symbols is used for the data symbols, and ten simulation runs with different starting points with respect to the PRNS are averaged. The resulting curve is smoothed with an exponential weighting factor of 0.9 to obtain the results shown in Fig. 3.

Figure 3 shows the results for the channel depicted in Fig. 2. The sampling phase is chosen to be 25 percent of a symbol interval away from the optimal sampling phase. Figure 3a corresponds to a 31-tap, symbol-spaced equalizer and Figure 3b to a $T/2$-spaced equalizer also spanning 31-symbol intervals. The behavior of the Kalman and fast Kalman algorithms has been observed to be identical [the difference in the output mean squared error (mse) is always smaller than 0.01 dB]. Therefore, the Kalman algorithm is not included on the plots.

The optimal fixed equalizer attains an output mse of 23.1. dB normalized to the signal level. For the symbol-spaced equalizer, about 125 iterations are required to converge to a normalized mse of 20 dB. For the $T/2$-spaced equalizer, all least-squares algorithms converge in about 150 iterations. The gradient algorithm requires about 400 iterations to converge to the same level. Very similar results were obtained for a channel with amplitude distortion as shown in Fig. 2 but with no

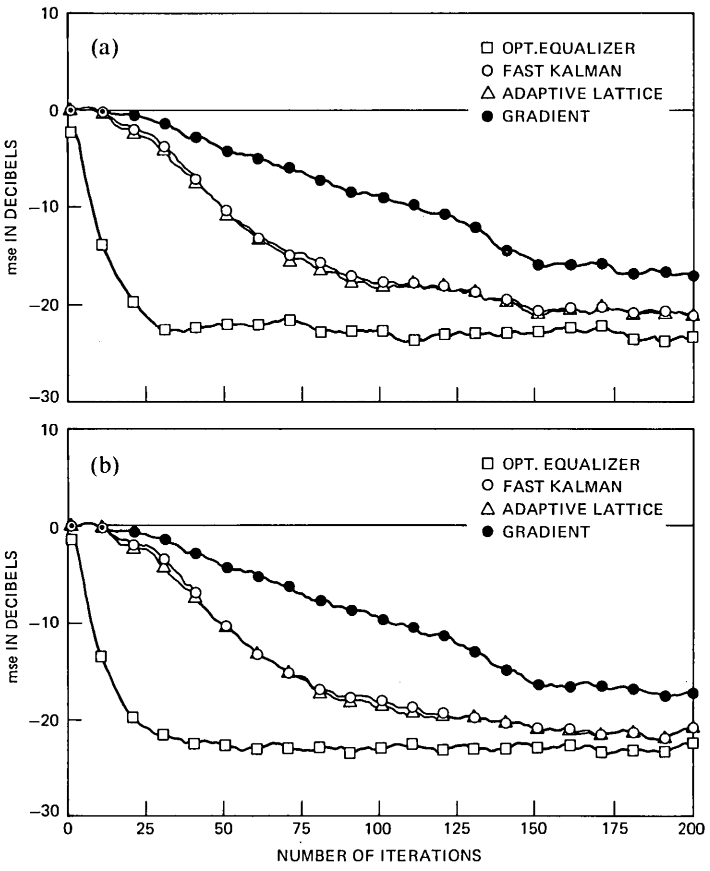


Fig. 3—(a) Convergence of symbol-spaced equalizer. $N = 31$, s/n $= 25$ dB. (b) Convergence of $T/2$-spaced equalizer. $N = 31, p = 2$, s/n $= 25$ dB.

phase distortion. If an ideal channel (no amplitude and no phase distortion) is used, the convergence time is reduced by about 35 percent.

These results indicate that, for realistic telephone channels, the least-squares algorithms behave very similarly and converge about three times faster than the stochastic gradient algorithm. Furthermore, it is found that the least-squares algorithms can be implemented successfully for both the symbol-spaced and $T/2$-spaced complex equalizers. Notice that the adaptive lattice algorithm requires a high number of matrix inversions per iteration if $p > 1$, which is often susceptible to numerical instabilities. However, our simulations did not uncover any stability problems.

The inclusion of an exponential weighting factor in the sum-of-squares cost function was proposed in Refs. 2 and 3 to allow for the tracking of time varying parameters or channels. When this was included in our simulations, and single precision floating-point arithmetic was used, an unstable behavior of the fast Kalman algorithm resulted. Double precision arithmetic (i.e., 56-bits for the mantissa) was found to eliminate the instability. The Kalman and the adaptive-lattice algorithms did not show this instability.

VI. CONCLUSION

The Kalman, fast Kalman, and adaptive-lattice algorithms are the fastest known methods for the training of equalizers. In particular, it was found that they require only about a third as many iterations as the gradient algorithm to converge to within 3 dB of the optimal mse. For a $T/2$-spaced equalizer and a worst-case channel, the equalizer start-up requires about 150 iterations.

The fast Kalman algorithm possesses the lowest complexity of these schemes. It requires about ten times as many multiplications per iteration for a $T/2$-spaced equalizer as the stochastic-gradient algorithm.

The adaptive-lattice algorithm requires more multiplications per update but has the advantage of being able to increase the equalizer length adaptively when needed. This is an advantage for off-line or batch processing but not for real-time applications.

The Kalman algorithm possesses the highest complexity and offers no advantage over the two other schemes. Therefore, it is not recommended for implementation.

VII. ACKNOWLEDGMENT

I would like to thank T. L. Lim for his contributions on this subject.

APPENDIX A

Derivation of the Least-Squares Algorithms

Let $\xi(n)$ be a p-dimensional complex vector denoting the new elements in the pm dimensional signal vector $x_m(n)$

$$x_m(n) = [\xi(n)^*, \cdots \xi(n-m+1)^*]^*. \qquad (20)$$

Let $c_m^s(n)$ be a complex pm dimensional coefficient vector, which denotes the equalizer coefficients if $s = e$, and let $c_m^s(n)$ be a complex $pm \times m$ dimensional matrix which stands for the forward predictor if $s = f$ and backward predictor if $s = b$.

Then the output of the equalizer, the forward and the backward predictors can generally be expressed as

$$y^s(m, n) = c_m^s(n-1)^* x_m(n). \qquad (21)$$

The double argument (m, n) denotes order m and time n. The desired signal $d^s(m, n)$ is defined as

$$d^s(m, n) = \begin{cases} a(n - D) & \text{for } s = e \\ \xi(n + 1) & \text{for } s = f \\ \xi(n - m) & \text{for } s = b \end{cases}. \tag{22}$$

The transmission delay between transmitter and receiver is denoted by D. The equalization and prediction errors $e^s(m, n)$ are defined as

$$e^s(m, n) = d^s(m, n) - y^s(m, n). \tag{23}$$

For $s = e$, $y^s(m, n)$, $d^s(m, n)$ and $e^s(m, n)$ are complex scalars, for $s \neq e$ they are p—dimensional, complex vectors.

The equalizer and prediction coefficients are determined such that they minimize the trace of the following least-squares cost function

$$\sum_{k=0}^{n} \lambda^{n-k}[d^s(m, k) - c_m^s(n)^* x_m(k)][d^s(m, k) - c_m^s(n)^* x_m(k)]^*. \tag{24}$$

Lambda is a geometric weighting factor. Differentiating the above cost function with respect to $c_m^s(n)$ and equating the resulting expression to zero yields the discrete-time, Wiener-Hopf equation for the coefficients

$$A_m(n) c_m^s(n) = v_m^s(n), \tag{25}$$

where

$$A_m(n) = \sum_{k=0}^{n} \lambda^{n-k} x_m(k) x_m(k)^* = \lambda A_m(n - 1) + x_m(n) x_m(n)^* \tag{26}$$

$$v_m^s(n) = \sum_{k=0}^{n} \lambda^{n-k} x_m(k) d^s(m, k)^* = \lambda v_m^s(n - 1) + x_m(n) d^s(m, n)^*. \tag{27}$$

In eq. 27, $A_m(n)$ is an $mp \times mp$ dimensional, Hermitian matrix, and $v_m^s(n)$ is a $mp \times p$ dimensional complex matrix. They are equivalent to the autocorrelation matrix and the cross-correlation vectors which occur in the familiar mean-square approach.

The optimal value of the cost function is obtained when the solution resulting from eq. (25) is substituted into eq. (24)

$$\epsilon^s(m, n) = E^s(m, n) - v_m^s(n)^* c_m^s(n), \tag{28}$$

where

$$E^s(m, n) = \sum_{k=0}^{n} \lambda^{n-k} d^s(m, k) d^s(m, k)^* = \lambda E^s(m, n - 1) + d^s(m, n) d^s(m, n)^*. \tag{29}$$

For $s = e$, $\epsilon^s(m, n)$ is a scalar. For $s \neq e$, it is a $p \times p$ Hermitian matrix.

A.1 Time update recursions

We observe that eqs. (26) and (27) contain a recursive definition of $A_m(n)$ and $v_m^s(n)$. This allows us to obtain a recursion in time for the optimal coefficients. Upon combining eqs. (25) and (27) we have

$$A_m(n)c_m^s(n) = \lambda A_m(n-1)c_m^s(n-1) + x_m(n)d^s(m, n)^*. \tag{30}$$

Add and subtract $x_m(n)x_m(n)^*c_m^s(n-1)$ to the right-hand side of eq. (28), then use eq. (23), the recursive form of eq. (26), and premultiply both sides with $A_m(n)^{-1}$ to obtain the desired recursion

$$c_m^s(n) = c_m^s(n-1) + A_m(n)^{-1}x_m(n)e^s(m, n)^*. \tag{31}$$

To obtain time update recursions for various auxiliary variables, we consider $c_m^s(n)^*v_m^t(n)$, where s and $t \in \{e, b, f\}$. From the time update recursion for the coefficients eq. (31), we conclude that

$$c_m^s(n)^*v_m^t(n) = [c_m^s(n-1)^* + e^s(m, n)x_m(n)^*A_m(n)^{-1}]v_m^t(n). \tag{32}$$

We multiply out and use eq. (27) for the first term. In the second term, we apply eq. (25) and obtain

$$\begin{aligned} c_m^s(n)^*v_m^t(n) &= \lambda c_m^s(n-1)^*v_m^t(n-1) \\ &\quad + c_m^s(n-1)^*x_m(n)d^t(m, n)^* + e^s(m, n)x_m(n)^*c_m^t(n). \end{aligned} \tag{33}$$

Now add and subtract $d^s(m, n)d^t(m, n)^*$ and use eqs. (21) and (24) to obtain

$$\begin{aligned} c_m^s(n)^*v_m^t(n) &= \lambda c_m^s(n-1)^*v_m^t(n-1) \\ &\quad + d^s(m, n)d^t(m, n)e^s(m, n)[x_m(n)^*c_m^t(n) - d^t(m, n)]. \end{aligned} \tag{34}$$

From eq. (31) it follows that the error after updating the coefficients

$$\epsilon^t(m, n)^* \equiv -x_m(n)^*c_m^t(n) + d^t(m, n) = (1-\gamma(m, n))e^t(m, n)^*, \tag{35}$$

where we defined the real scalar

$$\gamma(m, n) = x_m(n)^*A_m(n)^{-1}x_m(n). \tag{36}$$

Finally, we obtain from eqs. (34) and (35)

$$\begin{aligned} c_m^s(n)^*v_m^t(n) &= \lambda c_m^s(n-1)^*v_m^t(n-1) + d^s(m, n)d^t(m, n) \\ &\quad - e^s(m, n)e^t(m, n)^*[1-\gamma(m, n)]. \end{aligned} \tag{37}$$

A time recursion for $\epsilon^s(m, n)$ can be obtained from eq. (28) by using eqs. (28), (29), and (37)

$$\epsilon^s(m, n) = \lambda\epsilon^s(m, n-1) + [1-\gamma(m, n)]e^s(m, n)e^s(m, n)^*. \tag{38}$$

The mth component of the transformed correlation vector is defined as, cf. eqs. (11) and (19)

$$z(m, n) = [-c^b_{m-1}(n)^*, I]v^e_m(n)$$
$$= -c^b_{m-1}(n)^* v^e_{m-1}(n) + \sum_{k=0}^{n} \lambda^{n-k}\xi(k+1-m)d^e(m, k)^*.$$

We note that $d^e(m, k) = d^e(m-1, k)$, and within apply eq. (37) and obtain the time recursion

$$z(m, n) = \lambda z(m, n-1) + [1 - \gamma(m-1, n)]e^b(m-1, n)e^e(m-1, n)^*. \quad (39)$$

For future use, we define the $p \times p$ matrix

$$k(m-1, n) = \sum_{k=1}^{n+1} \lambda^{n+1-k}\xi(k-m)\xi(k)^* - v^b_{m-1}(n)^* c^f_{m-1}(n).$$

We apply eq. (37) and observe that $d^f(m-1, n) = \xi(n+1)$ and that $d^b(m-1, n) = \xi(n-m+1)$. Thus, we obtain

$$k(m-1, n) = \lambda k(m-1, n-1) + [1 - \gamma(m-1, n)]e^b(m-1, n)e^f(m-1, n)^*. \quad (40)$$

A.2 Order update recursions

Observe that from eq. (20) it follows

$$x_{m+1}(k) = \begin{bmatrix} \xi(k) \\ \hline \cdot \\ x_m(k-1) \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot \\ x_m(k) \\ \cdot \\ \hline \xi(k-m) \end{bmatrix}. \quad (41a, b)$$

This relation is the order-time update equation for the signal vector. It allows the derivation of order-time update equations for the various coefficient vectors, for the error values and for the error residuals. Update equations for various auxiliary variables necessary for the algorithm are also derived.

From eq. (41) and the definitions eqs. (26) and (27) and under the condition that $x_m(0) = 0$ it follows

$$A_{m+1}(n+1) = \begin{bmatrix} E^f(m, n) & v^f_m(n)^* \\ v^f_m(n) & A_m(n) \end{bmatrix} = \begin{bmatrix} A_m(n+1) & v^b_m(n+1) \\ v^b_m(n+1)^* & E^b(m, n+1) \end{bmatrix}. \quad (42a, b)$$

Upon combining eq. (26) and eq. (42a, b) we obtain the augmented Wiener-Hopf equations

$$A_{m+1}(n+1)\left[\begin{array}{c|c} I & \\ \hline & -c^b_m(n+1) \\ -c^f_m(n) & \\ \hline & I \end{array}\right] = \left[\begin{array}{c|c} \epsilon^f(m,n) & 0 \\ 0 & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ 0 & \epsilon^b(m,n+1) \end{array}\right], \tag{43a, b}$$

where we used partitioned matrices to represent the two systems of equations for the forward and the backward predictors, having the same matrix of coefficients.

Similar equations can be derived for predictors of reduced order

$$A_{m+1}(n+1)\left[\begin{array}{c|c} I & 0 \\ \hline -c^f_{m-1}(n) & -c^b_{m-1}(n) \\ \hline 0 & I \end{array}\right] = \left[\begin{array}{c|c} \epsilon^f(m-1,n) & k^b(m-1,n) \\ 0 & 0 \\ \cdot & \cdot \\ \cdot & \cdot \\ 0 & 0 \\ k^f(m-1,n) & \epsilon^b(m-1,n) \end{array}\right]. \tag{44a, b}$$

Equation (44a) can easily be verified by applying the expansion of (42b) with reduced order and time indices to $A_m(n)$ in eq. (42a). The same method with the order reversed verifies eq. (44b).

The auxiliary variables $k^f(m-1,n)$ and $k^b(m-1,n)$ are defined as

$$k^f(m-1,n) = \sum_{k=0}^{n+1} \lambda^{n+1-k}\xi(k-m)\xi(k)^* - v^b_{m-1}(n)^* c^f_{m-1}(n) \tag{45a}$$

$$k^b(m-1,n) = \sum_{k=0}^{n+1} \lambda^{n+1-k}\xi(k)\xi(k-m)^* - v^f_{m-1}(n)^* c^b_{m-1}(n). \tag{45b}$$

From eq. (45a, b) and with the definition of the predictor coefficients from eq. (25), it is easily verified that

$$k^f(m-1,n) = k^b(m-1,n)^* \equiv k(m-1,n). \tag{46}$$

The order update equations for the predictor coefficients and for the error residuals are now obtained through the combination of eqs. (43) and (44). We consider that

$$(43a) = (44a) - (44b)\epsilon^b(m-1,n)^{-1}k(m-1,n)$$

and

$$(43b) = (44b) - (44a)\epsilon^f(m-1,n)^{-1}k(m-1,n)^*.$$

This, with n reduced by one, yields an update equation for the prediction error residuals

$$\epsilon^f(m, n-1) = \epsilon^f(m-1, n-1) - k(m-1, n-1)^* \epsilon^b(m-1, n-1)^{-1} k(m-1, n-1) \tag{47a}$$

$$\epsilon^b(m, n) = \epsilon^b(m-1, n-1) - k(m-1, n-1) \epsilon^f(m-1, n-1)^{-1} k(m-1, n-1)^*. \tag{47b}$$

We assume now that $A_{m+1}(n+1)$ is nonsingular and premultiply eqs. (43) and (44) by its inverse. The same combinations of the new equations yield update equations for the coefficients.

$$c_m^f(n) = \left[\frac{c_{m-1}^f(n)}{0}\right] - \left[\frac{c_{m-1}^b(n)}{-I}\right] \cdot \epsilon^b(m-1, n)^{-1} k(m-1, n) \tag{48a}$$

$$c_m^b(n+1) = \left[\frac{0}{c_{m-1}^b(n)}\right] - \left[\frac{-I}{c_{m-1}^f(n)}\right] \cdot \epsilon^f(m-1, n)^{-1} k(m-1, n)^*. \tag{48b}$$

We premultiply eq. (48a) with $x_m(n+1)$ and eq. (48b) with $x_m(n+2)^*$. This yields eqs. (41), (21), and (23)

$$x_m(n+1)^* c_m^f(n) = x_{m-1}(n+1)^* c_{m-1}^f(n) - [x_{m-1}(n+1)^* \cdot c_{m-1}^b(n) - \xi(n-m+2)^*] \epsilon^b(m-1, n)^{-1} k(m-1, n) \tag{49a}$$

$$x_m(n+2)^* c_m^b(n+1) = x_{m-1}(n+1)^* c_{m-1}^b(n) - [x_{m-1}(n+1)^* \cdot c_{m-1}^f(n) - \xi(n+2)^*] \epsilon^f(m-1, n)^{-1} k(m-1, n)^*. \tag{49b}$$

With eq. (23) we identify the terms in the bracket of eq. (49a) as $e^b(m-1, n+1)^*$ and in the bracket of eq. (49b) as $e^f(m-1, n+1)^*$. We transpose eq. (49a), decrease n by 2, and use eq. (23) to obtain the update equations for $e^f(m, n-1)$. The update equation for $e^b(m, n)$ is obtained similarly.

$$e^f(m, n-1) = e^f(m-1, n-1) - k(m-1, n-2)^* \epsilon^b(m-1, n-2)^{-1} e^b(m-1, n-1) \tag{50a}$$

$$e^b(m, n) = e^b(m-1, n-1) - k(m, n-2) \epsilon^f(m, n-2)^{-1} e^f(m, n-1). \tag{50b}$$

The Kalman gain $g_{m+1}(n)$ is defined by

$$A_{m+1}(n) g_{m+1}(n) = x_{m+1}(n). \tag{51}$$

From eq. (40) we deduce

$$A_{m+1}(n)\left[\begin{array}{c|c} 0 & g_m(n) \\ \hline g_m(n-1) & 0 \end{array}\right] = \left[\begin{array}{c|c} v_m^f(n-1)^* g_m(n-1) & x_m(n) \\ \hline x_m(n-1) & v_m^b(n)^* g_m(n) \end{array}\right]. \qquad (52a, b)$$

We note from eq. (41) that eqs. (51), (52), and (43) are related as follows

$$(51) = (52a) + (43a)\epsilon^f(m, n-1)^{-1}[\xi(n) - v_m^f(n-1)^* g_m(n-1)]$$

$$(51) = (52b) + (43b)\epsilon^b(m, n)^{-1}[\xi(n-m) - v_m^b(n-1)^* g_m(n-1)].$$

We identify the terms in the brackets as the forward and backward prediction errors after updating the coefficients eq. (35). Performing the above-defined linear combinations of eqs. (43), (51), and (52) and premultiplying with $A_{m+1}(n)^{-1}$, yields order update equations for the Kalman gain.

$$g_{m+1}(n) = \left[\begin{array}{c} 0 \\ \hline g_m(n-1) \end{array}\right] + \left[\begin{array}{c} I \\ \hline -c_m^f(n-1) \end{array}\right] \cdot\epsilon^f(m, n-1)^{-1}\tilde{e}^f(m, n-1) \qquad (53a)$$

$$g_{m+1}(n) = \left[\begin{array}{c} g_m(n) \\ \hline 0 \end{array}\right] + \left[\begin{array}{c} -c_m^b(n) \\ \hline I \end{array}\right]\epsilon^b(m, n)^{-1}\tilde{e}^b(m, n). \qquad (53b)$$

We now proceed to obtain order update equations for $\gamma(m, n)$ as defined by eq. (36). Note from eqs. (36) and (51) that

$$\gamma(m, n) = x_m(n)^* g_m(n). \qquad (54)$$

Upon using eqs. (41a), (53a), (41b), and (53b) respectively, we obtain

$$\gamma(m, n) = \gamma(m-1, n-1) + \tilde{e}^f(m-1, n-1)^* \cdot\epsilon^f(m-1, n-1)^{-1}\tilde{e}^f(m-1, n-1) \qquad (55a)$$

$$\gamma(m, n) = \gamma(m-1, n) + \tilde{e}^b(m-1, n)^* \cdot\epsilon^b(m-1, n)^{-1}\tilde{e}^b(m-1, n). \qquad (55b)$$

APPENDIX B

Least-Squares Equalizer Update Algorithms

Here the least-squares equalizer update algorithms are listed and ordered such that they can be evaluated in the given sequence. Emphasis is put on a simplified notation compared to Appendix A. Generally, capitals denote matrices and lower case letters denote scalars and vectors. Table II gives the correspondence of variables and their dimensions, where $M = Np$.

Table II—Correspondence of variables

Variable	Appendix A	Appendix B	Dimension
Signal vector	$x_m(n)$	$x(n)$	M
Correlation matrix	$A_m(n)$	$A(n)$	$M \times M$
Equalizer coefficients	$c_m^e(n)$	$c(n)$	M
Equalizer error	$e^e(m, n)$	$e(m, n)$	1
Forward prediction			
Coefficients	$c_m^f(n-1)$	$F(n)$	$M \times p$
Error	$e^f(m, n-1)$	$f(m, n)$	p
Error residual	$\epsilon^f(m, n-1)$	$E^f(m, n)$	$p \times p$
Backware prediction			
Coefficients	$c_m^b(n)$	$B(n)$	$M \times p$
Error	$e^b(m, n)$	$\mathrm{b}(m, n)$	p
Error residual	$e^b(m, n)$	$E^b(m, n)$	$p \times p$
PARCOR coefficient	$k(m-1, n-1)$	$K(m, n)$	$p \times p$
Kalman gain	$g_m(n)$	$g(n)$	M

B.1 The Kalman algorithm

The Kalman algorithm makes use of the recursive definition of $A(n)$ in eq. (26) and the matrix inversion lemma, i.e.,

$$A(n)^{-1} = \frac{1}{\lambda}\left[A(n-1)^{-1} - \frac{A(n-1)^{-1}x(n)x(n)^*A(n-1)^{-1}}{\lambda + x(n)^*A(n-1)^{-1}x(n)}\right] \tag{56}$$

and defines

$$P(n) = A(n)^{-1}. \tag{57}$$

Upon using eqs. (2), (3), (8), (56), and (57) we obtain the Kalman algorithm for equalizer updating

$$t(n) = P(n-1)x(n), \tag{58}$$

$$g(n) = t(n)/(\lambda + x(n)^*t(n)), \tag{59}$$

$$P(n) = [P(n-1) - g(n)t(n)^*]\frac{1}{\lambda}, \tag{60}$$

$$y(n) = c(n-1)^*x(n), \tag{61}$$

$$e(n) = d(n) - y(n), \tag{62}$$

$$c(n) = c(n-1) + g(n)e(n)^*. \tag{63}$$

To initialize, set all variables to zero except $P(0)$ which is set to $P(0) = 1/\delta\, I$. Note that because of the Hermitian nature of $A(n)$ and consequently of $P(n) = A(n)^{-1}$, eq. (16) needs only be evaluated for the upper (or lower) triangle including the diagonal.

B.2 The fast Kalman algorithm

To obtain the fast Kalman algorithm, apply eqs. (21) to (23), (31), (35), and (38) for the forward predictor of fixed order m. This yields eqs. (64) to (67)

$$f(n) = \xi(n) - F(n-1)^* x(n-1) \tag{64}$$

$$F(n) = F(n-1) + g(n-1) f(n)^* \tag{65}$$

$$f(n)' = f(n)[1 - g(n-1)^* x(n-1)] \tag{66}$$

$$E(n) = \lambda E(n-1) + f(n)' f(n)^*. \tag{67}$$

Then use eq. (53a) to calculate the extended Kalman gain $\bar{g}(n)$ and partition as indicated

$$\bar{g}(n) = \begin{bmatrix} E(n)^{-1} f(n)' \\ \hline g(n-1) - F(n)E(n)^{-1} f(n)' \end{bmatrix} = \begin{bmatrix} g(n)' \\ \hline \mu(n) \end{bmatrix}. \qquad (68)\ (69)$$

Now the backward prediction error $b(n)$ is calculated from eq. (23)

$$b(n) = \xi(n-N) - B(n-1)^* x(n). \tag{70}$$

Equations (31) and (53b) can now be used to update the backward predictor and finally to determine the updated Kalman gain

$$B(n) = [B(n-1) + g(n)' b(n)^*][I_{pp} - \mu(n) b(n)^*]^{-1} \tag{71}$$

$$g(n) = g(n)' + B(n)\mu(n). \tag{72}$$

Equations (21) to (23) and eq. (31) applied to the equalizer conclude the algorithm.

$$y(n) = c(n-1)^* x(n) \tag{73}$$

$$e(n) = d(n) - y(n) \tag{74}$$

$$c(n) = c(n-1) + g(n) e(n)^*. \tag{75}$$

To initialize, set

$$F(0) = B(0) = 0_{Mp}$$

$$x(0) = g(0) = c(0) = 0_M$$

and

$$E(0) = \delta I_{pp}.$$

Notice for the numerical evaluation that E is Hermitian and that the matrix inversions in eqs. (68) and (71) can be avoided if a p–dimensional system of linear equations is solved for multiple right-hand sides.

B.3 The lattice algorithm

For each time instant, the algorithm is initialized for order zero

$$y(0, n) = \gamma(0, n) = 0 \tag{76}$$

$$e(0, n) = d(n) \tag{77}$$

$$f(0, n) = b(0, n) = \xi(n) \tag{78}$$

$$E^f(0, n) = E^b(0, n) = \lambda E^f(0, n-1) + \xi(n)\xi(n)^*. \tag{79}$$

From eq. (40), the following time update equation follows

$$K(m, n) = \lambda K(m, n-1) + t(m, n-1) f(m-1, n)^*. \tag{80}$$

Then we obtain from eq. (50) order update equations for the prediction errors

$$f(m, n) = f(m-1, n) - G(m, n-1) b(m-1, n-1) \tag{81}$$

$$b(m, n) = b(m-1, n-1) - H(m, n-1) f(m-1, n), \tag{82}$$

where auxiliary $p \times p$ matrices are determined as

$$G(m, n) = K(m, n)^* E^b(m-1, n-1)^{-1} \tag{83}$$

$$H(m, n) = K(m, n) E^f(m-1, n)^{-1}. \tag{84}$$

Equation (48), together with eqs. (83) and (84), permits the update of the prediction error residuals

$$E^f(m, n) = E^f(m-1, n) - G(m, n) K(m, n) \tag{85}$$

$$E^b(m, n) = E^b(m-1, n-1) - H(m, n) K(m, n)^*. \tag{86}$$

The equalizer output and output error follow

$$y(m, n) = y(m-1, n) + z(m, n-1)^* \cdot E^b(m-1, n-1)^{-1} b(m-1, n) \tag{87}$$

$$e(m, n) = d(n) - y(m, n). \tag{88}$$

From eqs. (35) and (55), we have

$$t(m, n) = [1 - \gamma(m-1, n)] b(m-1, n) \tag{89}$$

$$\gamma(m, n) = \gamma(m-1, n) + t(m, n)^* E^b(m-1, n)^{-1} t(m, n). \tag{90}$$

Equation (39) finally allows to update the coefficients

$$z(m, n) = \lambda z(m, n-1) + t(m, n) e(m-1, n). \tag{91}$$

Equations (87), (88), and (91) are evaluated for $m \epsilon [1, N]$. The other equations are evaluated for $m \epsilon [1, N-1]$. To initialize, set all variables to zero except $E^f(0, 0) = E^b(0, 0) = \delta I_{pp}$.

For the numerical evaluation, it should be noted that $E^f(m, n)$ and $E^b(m, n)$ are Hermitian, thus, only the real diagonal and the upper or the lower triangle need be computed. Note also, that $G(m, n)$ and $H(m, n)$ are computed best as the solution of a p-dimensional system of linear equations with p right-hand sides.

REFERENCES

1. D. Godard, "Channel Equalization Using a Kalman Filter for Fast Data Transmission," IBM J. Res. Develop. (May 1974), pp. 267–73.
2. D. D. Falconer and L. Ljung, "Application of Fast Kalman Estimation to Adaptive Equalization," IEEE Trans. on Commun., *COM-26,* No. 10 (October 1978), pp. 1439–46.
3. E. H. Satorius and F. D. Pack, unpublished work.
4. E. Shichor, private communication.
5. D. D. Falconer, V. B. Lawrence, and S. K. Tewksbury, "Processor-Hardware Considerations for Adaptive Digital Filter Algorithms," Conf. Records of the ICC 1980, June 1980, Seattle, Washington, Paper No. 57.5.
6. M. J. Shensa, "A Least Squares Lattice Decision Feedback Equalizer," Conf. Records of the ICC 1980, June 1980, Seattle, Washington, Paper No. 57.6.
7. T. L. Lim and M. S. Mueller, "Rapid Equalizer Start-Up Using Least-Squares Algorithms," Conf. Records of the ICC 1980, June 1980, Seattle, Washington, Paper No. 57.7.
8. M. S. Mueller, private communication.
9. G. Ungerboeck, "Fractional Tap-Spacing Equalizers and Consequences for Clock Recovery for Data Modems," IEEE Trans. on Commun., *COM-24,* No. 8 (August 1976), pp. 856–64.
10. L. Ljung, M. Morf, and D. D. Falconer, "Fast Calculation of Gain Matrices for Recursive Estimation Schemes," Int. J. on Control (January 1978), pp. 1-19.
11. M. Morf, A. Vieira, and D. T. Lee, "Ladder Forms for Identification and Speech Processing," Proc. 1977 IEEE Conf. on Decision and Control, New Orleans, Louisiana, December 1977, pp. 1074–8.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Numerical Integration of Stochastic Differential Equations—II

By H. S. GREENSIDE and E. HELFAND

(Manuscript received March 19, 1981)

In a previous paper, a method was presented to integrate numerically nonlinear stochastic differential equations (SDEs) with additive, Gaussian, white noise. The method, a generalization of the Runge-Kutta algorithm, extrapolates from one point to the next applying functional evaluations at stochastically determined points. This paper extends (and at one point corrects) algorithms for the simple class of equations considered in the previous paper. In addition, the method is expanded to treat vector SDEs, equations with time-dependent functions, and SDEs higher than first order. The parameters for several explicit integration schemes are displayed.

I. INTRODUCTION

There are two approaches to the study of a physical system described by a stochastic differential equation (SDE). On the one hand, one may work with an equation for the probability distribution function for the random variables such as the Fokker-Planck equation. On the other hand, one may attempt to generate representative points on a trajectory by direct solution of the SDE. With either approach it is rare that analytical solutions can be found, except for linear systems. While the deterministic equation for the probability distribution can be solved numerically with standard techniques, in practice there are great difficulties. Numerical techniques for SDEs are a less-developed subject, but quite promising since they are capable of giving direct information about the random process, such as the power spectrum, higher moments, and transition rates. Several discussions of the problem have been published.[1]

A previous paper[2] (hereafter referred to as I) describes a systematic approach to the numerical solution of SDEs. Attention was limited to the simple one-variable equation of the form

$$\frac{dx}{dt} = f(x) + A(t), \tag{1}$$

where $f(x)$ is a differentiable function through some order, and $A(t)$ is a Gaussian white noise source with

$$\langle A(t) \rangle = 0, \tag{2}$$

$$\langle A(t)A(t') \rangle = \xi\delta(t - t'). \tag{3}$$

The procedure introduced was an extension of the Runge-Kutta method for numerical solution of deterministic differential equations. In the Runge-Kutta technique, as applied for instance to $dx/dt = f(x)$, $f(x)$ is evaluated at $x(t)$ and a number of other definite points. From these evaluations an extrapolation from $x(t)$ to an estimate, $\hat{x}(t + h)$, is constructed which is accurate to a given order in the time step, h, i.e. errors are less than order h^k. To apply this procedure for SDEs, the function $f(x)$ is evaluated at stochastically selected points. The algorithm is such that all moments of $\hat{x}(t + h) - x(t)$ are correct to the kth order in the step size h.

In this paper, we continue and extend the work begun in I in two ways. First, we discuss further the algorithms given in I. The two possible second-order algorithms described earlier are generalized to two families of parameter sets. A third-order algorithm proposed in I was in error and is corrected. We go on to consider a fourth order four-stage algorithm, but report our inability to find one. Our analysis suggests that kth order k-stage algorithms do not exist for $k \geq 4$.

The second way in which we extend the discussion in I is to generalize the method to three other classes of SDEs. The first class is vector SDEs in which each component has its own independent Gaussian noise source:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}) + \mathbf{A}(t). \tag{4}$$

This generalization may then be applied to the study of the class of SDEs in which $\mathbf{f}$ is an explicit function of both $\mathbf{x}$ and t. Also we show how to handle SDEs which are higher than first order (higher order derivatives of $\mathbf{x}$ with respect to t appear).*

In Section II, we briefly review the previous work and introduce the nomenclature. Section III contains a discussion of explicit algorithms which may be used for one-variable SDEs such as eq. (1). In Section IV we discuss the integration of vector SDEs and how to solve time-dependent and higher-order systems. This is illustrated in Section V

* A discussion on how our method may be applied to SDEs with multiplicative random variables will be presented elsewhere (H. S. Greenside, to be published).

with an explicit, third-order, vector algorithm. Finally, we indicate some new directions which seem important to explore.

II. REVIEW AND NOTATION

A convenient way to solve an SDE such as eq. (1) is to rewrite it as an integral equation

$$x(h) = x(0) + \int_0^h ds\, f[x(s)] + w^{[0]}(h), \tag{5}$$

where

$$w^{[0]}(t) = \int_0^t ds\, A(s) \tag{6}$$

is the Wiener process. We later need the iterates of $w^{[0]}$ defined by

$$w^{[n]}(t) = \int_0^t ds\, w^{[n-1]}(s). \tag{7}$$

The $w^{[n]}$ are Gaussian random variables with zero mean and covariances given by eqs. (18) and (19) of I. One can expand the right-hand size of eq. (5) in a series in $h^{1/2}$, where the order of the stochastic terms is determined in probability. The result is

$$x(h) = x_0 + hf + \tfrac{1}{2}h^2 ff' + (\tfrac{1}{6})h^3(ff'^2 + f^2 f'') + \cdots + S(h), \tag{8}$$

where the stochastic part is given by

$$\begin{aligned} S(h) = {} & \{w^{[0]}(h)\}_{1/2} + \{f' w^{[1]}(h)\}_{3/2} \\ & + \left\{\frac{1}{2} f'' \int_0^h ds\, [w^{[0]}(s)]^2\right\}_2 \\ & + \Bigg\{ f'^2 w^{[2]}(h) + ff''[hw^{[1]}(h) - w^{[2]}(h)] \\ & + \left(\frac{1}{6}\right) f''' \int_0^h ds\, [w^{[0]}(s)]^3 \Bigg\}_{5/2} \\ & + \Bigg\{ \frac{1}{2} f' f'' \left(\int_0^h ds\, (h-s)[w^{[0]}(s)]^2 + [w^{[1]}(h)]^2 \right) \\ & + \frac{1}{2} ff''' \int_0^h ds\, s[w^{[0]}(s)]^2 \\ & + \left(\frac{1}{24}\right) f^{(iv)} \int_0^h ds\, [w^{[0]}(s)]^4 \Bigg\}_3 + \cdots . \end{aligned} \tag{9}$$

[Note that in the equation for $S(h)$ in I, eq. (I14), the third-order term involving $f'f''$ was incorrectly given.] We have written x_0 for $x(0)$, and $f^{(n)}$ indicates the nth derivative of f evaluated at $x = x_0$. In $S(h)$ terms of order h^j in probability have been gathered together in braces and the subscript j placed after the braces. The moments of the stochastic variable $S(h)$ are, to third order in h,

$$\langle S \rangle = \tfrac{1}{4}h^2\xi f'' + h^3(\tfrac{1}{4}\xi f'f'' + \tfrac{1}{6}\xi ff''' + \tfrac{1}{24}\xi^2 f^{(iv)}) + \cdots, \tag{10}$$

$$\langle S^2 \rangle = h\xi + h^2\xi f' + h^3(\tfrac{2}{3}\xi f'^2 + \tfrac{2}{3}\xi ff'' + \tfrac{1}{3}\xi^2 f''') + \cdots, \tag{11}$$

$$\langle S^3 \rangle = (7/4)h^3\xi^2 f'' + \cdots. \tag{12}$$

(Note that the coefficient of $f'f''$ in $\langle S \rangle$ is in error in I and is corrected here.) For the expansion through order h^k the terms of S nonlinear in the w's, hence non-Gaussian, do not contribute to moments higher than $2k - 1$. It follows that if errors in $\langle S^2 \rangle$ are reduced to $O(h^{k+1})$ then errors in $\langle S^n \rangle$, $n \geq 2k - 2$, will be that order or higher order in h.

In I it was proposed to integrate the SDE, eq. (1), by an extension of the Runge-Kutta scheme.[3] The algorithm for an l stage procedure is as follows:

$$g_1 = f(x_0 + h^{1/2}\xi^{1/2}Y_1), \tag{13}$$

$$g_2 = f(x_0 + h\beta_{21}g_1 + h^{1/2}\xi^{1/2}Y_2), \tag{14}$$

$$\vdots$$

$$g_l = f(x_0 + h\beta_{l1}g_1 + \cdots + h\beta_{l,l-1}g_{l-1} + h^{1/2}\xi^{1/2}Y_l), \tag{15}$$

$$x = x_0 + h(A_1g_1 + \cdots + A_lg_l) + h^{1/2}\xi^{1/2}Y_0. \tag{16}$$

The $l + 1$ stochastic variables $Y_0, \cdots, Y_l$ are Gaussianly distributed with mean zero and covariance

$$\langle Y_iY_j \rangle = L_{ij}. \tag{17}$$

The matrix $\mathbf{L}$, being symmetric, has $\tfrac{1}{2}(l + 1)(l + 2)$ independent parameters. Numerically, it is convenient to generate the $\mathbf{Y}$ set by writing

$$Y_j = \sum_{n=1}^{j+1} \lambda_{jp}Z_p. \tag{18}$$

where the $\mathbf{Z}$'s are a set of independent Gaussian random variables with mean zero and variance unity. Note particularly that in eq. (18) only $j + 1$ variables Z_p need be used to define Y_j. The λ_{jn} form $l + 1$ vectors of $l + 1$ components

$$\lambda_0 = \{\lambda_{01}, 0, 0, \cdots, 0\}, \tag{19}$$

$$\lambda_1 = \{\lambda_{11}, \lambda_{12}, 0, \cdots, 0\}, \tag{20}$$

$$\cdots$$

$$\lambda_l = \{\lambda_{l1}, \lambda_{l2}, \lambda_{l3}, \cdots, \lambda_{l,l+1}\}. \tag{21}$$

The $\frac{1}{2}(l+1)(l+2)$ parameters λ_{jn} are related to the same number of independent parameters in the symmetric **L** matrix by

$$L_{ij} = \lambda_i \cdot \lambda_j \tag{22}$$

The algorithm eqs. (13) to (16) can be expressed as a power series in $h^{1/2}$, there being a deterministic part and a stochastic part, $\tilde{S}$. In turn, the moments of the stochastic part can be expanded in powers of h. (A two-stage, second-order illustration is given in I.) Each term of the deterministic part and of the moments takes the form of: (a power of h) $\times$ (a power of ξ) $\times$ (a product of powers of f and its derivatives) $\times$ (a coefficient which is a function of the parameters A_i, β_{ij}, and λ_{jn}). Corresponding terms occur in the expansion, eq. (8), and in the moments of $S(h)$ given by eq. (9), except that in the latter cases the coefficients have definite numerical values. Therefore, equations for the parameters are obtained by equating the two coefficients for each different term (i.e., different product of f and derivatives) through a given power, h^k. The series match independently of the explicit form of $f(x)$.

There are $(l+1)^2$ parameters: A_i $(i = 1, \cdots, l)$; β_{ij} $(i = 2, \cdots, l$, and $j = 1, \cdots, i-1)$; λ_{ij} $(i = 0, \cdots, l$, and $j = 1, \cdots, i+1)$. There may be fewer conditions to be satisfied than this. If so, it is convenient to use only m rather than $l+1$ Gaussian random variables, Z_p. This amounts to setting $\lambda_{jp} = 0$ for $p > m$.

A procedure which is correct through order h^k, which involves l stages, and which utilizes m Gaussians will be called a $k_O l_S m_G$ algorithm. Explicit examples are given in Section III.

III. THE $2_O 2_S 1_G$, $3_O 3_S 2_G$, AND OTHER ALGORITHMS

In I the parameters were displayed for a $2_O 2_S 1_G$ algorithm. There is one degree of freedom (6 parameters, 5 equations). The most general choice of parameters is

$$\begin{aligned}
A_1 &= 1 - \tfrac{1}{2}\alpha^{-1},\\
A_2 &= \tfrac{1}{2}\alpha^{-1},\\
\beta_{21} &= \alpha,\\
\lambda_{01} &= 1,\\
\lambda_{11} &= \tfrac{1}{2}[1 \pm (2\alpha - 1)^{-1/2}],\\
\lambda_{21} &= \tfrac{1}{2}[1 \mp (2\alpha - 1)^{1/2}],
\end{aligned} \tag{23}$$

with $\alpha > \frac{1}{2}$. In I the solutions with $\alpha = 1$ were suggested as particularly convenient.

In the Appendix of I, a discussion of the $3_O 3_S 2_G$ algorithm was presented. The exposition contained an error and should be disregarded. A proper discussion follows.

The $16 - \frac{1}{2}(4 - m)(5 - m)$ parameters of a $3_O3_Sm_G$ algorithm must satisfy 14 equations obtained by matching the expansion of eqs. (13) to (16), and the expansion of eq. (8) and moments of eq. (9):

$$\sum_{i=1}^{3} A_i = 1, \tag{24}$$

$$\sum_{i=2}^{3} A_i\alpha_i = \tfrac{1}{2}, \tag{25}$$

$$\sum_{i=2}^{3} A_i\alpha_2^2 = \tfrac{1}{3}, \tag{26}$$

$$A_3\beta_{32}\beta_{21} = \tfrac{1}{6}, \tag{27}$$

$$L_{00} = 1, \tag{28}$$

$$\sum_{i=1}^{3} A_iL_{0i} = \tfrac{1}{2}, \tag{29}$$

$$\sum_{i=1}^{3} A_iL_{ii} = \tfrac{1}{2}, \tag{30}$$

$$\sum_{i=1}^{3} A_iL_{0i}^2 = \tfrac{1}{3}, \tag{31}$$

$$\sum_{i=1}^{3} A_iL_{ii}^2 = \tfrac{1}{3}, \tag{32}$$

$$\sum_{i=1}^{3} A_iL_{0i}L_{ii} = \tfrac{1}{3}, \tag{33}$$

$$\sum_{i=2}^{3} A_i\alpha_iL_{ii} = \tfrac{1}{3}, \tag{34}$$

$$\sum_{i=2}^{3} A_i\alpha_iL_{0i} = \tfrac{1}{3}, \tag{35}$$

$$\sum_{i=2}^{3} A_i \sum_{j=1}^{i-1} \beta_{ij}(L_{ij} + \tfrac{1}{2}L_{jj}) = \tfrac{1}{4}, \tag{36}$$

$$\sum_{i=1}^{3}\sum_{j=1}^{3} A_iA_jL_{ij} + 2\sum_{i=2}^{3} A_i \sum_{j=1}^{i-1} \beta_{ij}L_{0i} = \tfrac{2}{3}, \tag{37}$$

where, by definition,

$$\alpha_i = \sum_{j=1}^{i-1} \beta_{ij}, \qquad i = 2, 3. \tag{38}$$

A remarkable simplification occurs if we assume

$$A_1 = 0, \tag{39}$$

$$L_{0i} = L_{ii} = \alpha_i, \qquad i = 2, 3 \tag{40}$$

(it can be shown that no real solutions exist without these conditions). Then the 14 equations, eqs. (24) to (37), reduce to 7 independent equations in 8 unknowns for a $3_O3_S2_G$ algorithm. This leaves one degree of freedom which we can take as α_2. We are, of course, only interested in solutions for which all the parameters are real. This requires that

$$0 < \alpha_2 < \tfrac{1}{3} \qquad \text{or} \qquad \tfrac{2}{3} \le \alpha_2 \le 1. \tag{41}$$

Since some of the equations are nonlinear, there are multiple solutions in certain regions. Further details are presented in the Appendix. Table I gives an indication of the behavior of the parameters as α_2 is varied. Since λ_{12} is obtained from the solution of a quadratic equation, two choices are shown. As α_2 increases through 0.247583, a new pair of real roots of the equations appears, while at 0.2689703 the other pair becomes complex. There are four roots in the range $\tfrac{2}{3} \le \alpha_2 \le 1$ (although at $\tfrac{2}{3}$ and 1, roots are degenerate). The parameter set corresponding to $\alpha_2 = \tfrac{2}{3}$ looks particularly interesting because all

Table I—Parameters for $3_O3_S2_G$ algorithms appropriate to a one-variable SDE*

α_2†	β_{31}	A_2	λ_{11}	λ_{12}		λ_{32}
0.1	−1.82639	0.34247	0.03341	−1.14271	0.22458	0.45453
0.2	−0.82716	0.48077	0.12491	−1.15128	0.31072	0.41574
0.25	−0.72222	0.57143	0.24733	−0.90403	0.26211	0.37268
			−0.22692	0.81865	1.30789	−0.37268
0.30	−0.79630	0.67568	−0.31442	0.19962	5.71406	−0.27639
$\tfrac{2}{3}$	−1.0	$\tfrac{3}{4}$	$-\tfrac{1}{12}$	0.61844‡	−2.50406‡	0.0
0.7	−0.65079	0.67568	−0.14525	0.67143	−1.88108	0.27639
			0.06670	0.47809	−2.57869	−0.27639
0.8	−0.17901	0.48077	−0.14275	0.63842	−1.42839	0.41574
			0.14191	0.46368	−1.78360	−0.41574
0.9	0.01003	0.34247	−0.12381	0.63799	−1.24446	0.45453
			0.07126	0.64299	−1.21137	−0.45453
1.0	$\tfrac{1}{9}$	$\tfrac{1}{4}$	$-\tfrac{1}{16}$	0.76579§	−1.00149§	$2^{1/2}/3$

* The parameters not listed are given by:

$\beta_{21} = \alpha_2$
$\beta_{32} = 1/(6A_3\alpha_2)$
$A_1 = 0$
$A_3 = 1 - A_2$
$\lambda_{21} = \alpha_2$
$\lambda_{22} = +(\alpha_2 - \alpha_2^2)^{1/2}$
$\lambda_{31} = \beta_{31} + \beta_{32}$

† α_2 is varied as the one degree of freedom.
‡ $\pm 39^{1/2}/4 - 2^{3/2}/3$.
§ $-2^{1/2}/12 \pm 1799^{1/2}/48$.

parameters are ≤ 1, and $\alpha_3 = \lambda_{31} = \lambda_{32} = 0$. A second interesting parameter set is the one for $\alpha_2 = 1$.

A $3_O4_S2_G$ solution will be discussed in Section IV. In this case, there are enough degrees of freedom so that the parameters can be selected to produce an algorithm which integrates the deterministic part of the equation through fourth order.

It is straightforward, but quite lengthy, to extend all of the equations in Section II to fourth order. We have done so. For a $4_O4_Sm_G$ algorithm, there are $25 - \frac{1}{2}(5 - m)(6 - m)$ parameters which must satisfy 29 equations (39 coefficients must be matched but 10 of the resulting equations are not independent). The assumption that $A_1 = 0$ and $L_{0i} = L_{ii} = \alpha_1$, $i = 2, 3, 4$, reduces the number of unknowns to $18 - \frac{1}{2}(5 - m)(6 - m)$; and, remarkably, the number of independent equations is reduced to 18. Thus, there may be solutions with 5 Gaussians (the maximum possible). Unfortunately, after a reasonably thorough search for solutions we were not able to find any real solutions.* Although we do not have a proof that no real solutions exist, it appears that there is no $4_O4_Sm_G$ algorithm.

IV. ALGORITHM FOR VECTOR SDEs

Consider next vector SDEs of the type

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}) + \mathbf{A}(t), \tag{42}$$

with

$$\langle A_\kappa(t)\rangle = 0,$$
$$\langle A_\kappa(t)A_\mu(t')\rangle = \xi_\kappa\delta_{\kappa\mu}\delta(t - t'). \tag{43}$$

If the $\mathbf{A}$ covariance matrix is not diagonal, then linear combinations of the equations can be taken to diagonalize it, or the algorithm described below can be modified.

In a fashion analogous to that used to obtain eqs. (8) and (9) one can write (using the summation convention on repeated indices)

$$x_\kappa(t) = x_{0\kappa} + hf_\kappa + \tfrac{1}{2}h^2 f_{\kappa,\mu}f_\mu + (\tfrac{1}{6})h^3(f_{\kappa,\mu}f_{\mu,\nu}f_\nu + f_{\kappa,\mu\nu}f_\mu f_\nu) + \cdots + S_\kappa(h), \tag{44}$$

* The algebra involved in many of these calculations is extremely lengthy. Occasionally it is susceptible to simplification by a combination of equations. For these reasons, we would be willing to provide further details of our calculations to interested parties.

$$\begin{aligned}
S_\kappa(h) = {} & \{w_\kappa^{[0]}(h)\}_{1/2} + \{f_{\kappa,\mu} w_\mu^{[1]}(h)\}_{3/2} \\
& + \left\{\frac{1}{2} f_{\kappa,\mu\nu} \int_0^h ds\, w_\mu^{[0]}(s) w_\nu^{[0]}(s)\right\}_2 \\
& + \left\{ f_{\kappa,\mu} f_{\mu,\nu} w_\nu^{[2]}(h) + f_{\kappa,\mu\nu} f_\mu [h w_\nu^{[1]}(h) + w_\nu^{[2]}(h)] \right. \\
& + \left. \left(\frac{1}{6}\right) f_{\kappa,\mu\nu\rho} \int_0^h ds\, w_\mu^{[0]}(s) w_\nu^{[0]}(s) w_\rho^{[0]}(s) \right\}_{5/2} \\
& + \left\{ \frac{1}{2} f_{\kappa,\mu} f_{\mu,\nu\rho} \int_0^h ds\, (h - s) w_\nu^{[0]}(s) w_\rho^{[0]}(s) \right. \\
& + f_{\kappa,\mu\nu} f_{\nu,\rho} \int_0^h ds\, w_\mu^{[0]}(s) w_\rho^{[1]}(s) \\
& + \frac{1}{2} f_{\kappa,\mu\nu\rho} f_\rho \int_0^h ds\, s w_\mu^{[0]}(s) w_\nu^{[0]}(s) \\
& + \left. \left(\frac{1}{24}\right) f_{\kappa,\mu\nu\rho\sigma} \int_0^h ds\, w_\mu^{[0]}(s) w_\nu^{[0]}(s) w_\rho^{[0]}(s) w_\sigma^{[0]}(s) \right\}_3 \\
& + \cdots,
\end{aligned} \tag{45}$$

where

$$f_{\kappa,\mu\nu\cdots\rho} = \frac{\partial}{\partial x_\mu} \frac{\partial}{\partial x_\nu} \cdots \frac{\partial}{\partial x_\rho} f_\kappa \big|_{\mathbf{x}=\mathbf{x}(0)}, \tag{46}$$

$$w_\kappa^{[0]}(t) = \int_0^t ds\, A_\kappa(s), \tag{47}$$

with $w_\kappa^{[n]}$ being the nth iterate of $w_\kappa^{[0]}$. There is one major difference to note between eqs. (45) and (9). In the former, a distinction must be made between the term with $f_{\kappa,\mu} f_{\mu,\nu\rho}$ and that with $f_{\kappa,\mu\nu} f_{\nu,\rho}$; in the latter, both are $f'f''$. This leads to an extra equation which the parameters will have to satisfy for sets. Such differences first enter in fourth order for sets of deterministic differential equations, while for SDEs they enter at third order.

The earlier algorithm for numerical integration is easily generalized to

$$g_{1\kappa} = f_\kappa(\{x_{0\mu} + h^{1/2} \xi_\mu^{1/2} Y_{1\mu}\}), \tag{48}$$

$$g_{2\kappa} = f_\kappa(\{x_{0\mu} + h\beta_{21} g_{1\mu} + h^{1/2} \xi_\mu Y_{2\mu}\}), \tag{49}$$

$$\vdots$$

$$g_{l\kappa} = f_\kappa(\{x_{0\mu} + h\beta_{l1}g_{1\mu} + \cdots + h\beta_{l,l-1}g_{l-1\mu} + h^{1/2}\xi_\mu^{1/2}Y_{l\mu}\}), \tag{50}$$

$$x_\kappa(h) = x_{0\kappa} + h(A_1 g_{1\kappa} + \cdots + A_l g_{l\kappa}) + h^{1/2}\xi_\kappa^{1/2}Y_{0\kappa}, \tag{51}$$

where $\{x_\mu\}$ denotes the set of variables $x_1, \cdots, x_N$. It is appropriate to take the covariance of the $Y_{i\kappa}$ as

$$\langle Y_{i\kappa}Y_{j\mu}\rangle = L_{ij}\delta_{\kappa\mu}; \tag{52}$$

or, equivalently, to write

$$Y_{i\kappa} = \sum_{j=1}^{m} \lambda_{ij}Z_{j\kappa}, \tag{53}$$

where the $Z_{j\kappa}$ are Nm independent Gaussian random variables of mean zero and variance unity. In general, $m = l + 1$, but it may be possible to construct an algorithm with smaller m, i.e., a $k_O l_S m_G$ scheme for vector SDEs.

Equation (51) may be expanded to any desired order, h^k, giving a deterministic and a stochastic part, $\tilde{\mathrm{S}}$. Once again, equations for the parameters are determined by demanding equality of the deterministic part to that of the expansion, eq. (44). Further equations result from equating the moments of S and $\tilde{\mathrm{S}}$. In general, there are more equations to be satisfied for sets than for a single variable because of the mixed partial derivatives. Note, however, that the parameters for the algorithm, A_i, β_{ij}, λ_{ij}, are not functions of the component index, κ.

Once an algorithm is available for vector SDEs, it can be applied to two other classes of SDEs. Consider the generalization of eq. (4) where **f** is time dependent

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, t) + \mathbf{A}(t). \tag{54}$$

By introducing an extra variable $x_{N+1} = t$ (i.e., $dx_{N+1}/dt = 1$), one can rewrite eq. (54) in the form eq. (42) as an $N + 1$ dimensional, autonomous vector SDE:

$$\frac{d\mathbf{y}}{dt} = \mathbf{F}(\mathbf{y}) + \mathbf{A}(t), \tag{55}$$

$$\mathbf{y} \equiv (x_1, x_2, \cdots, x_N, x_{N+1}), \tag{56}$$

$$\mathbf{F}(y) \equiv (f_1, f_2, \cdots, f_N, 1), \tag{57}$$

$$\xi = (\xi_1, \xi_2, \cdots, \xi_N, 0). \tag{58}$$

In this case, the random variables $Y_{i,N+1}$ or $Z_{j,N+1}$ need not be generated since they are always multiplied by zero.

An nth order differential equation may also be integrated in a straightforward manner. Consider as an example the simple case

$$\frac{d^n x}{dt^n} + c_1(x)\frac{d^{n-1}x}{dt} + \cdots + c_n(x) = A(t). \tag{59}$$

This equation is equivalent to the n dimensional vector SDE

$$\frac{dy_1}{dt} = y_2, \tag{60}$$

$$\vdots$$

$$\frac{dy_{N-1}}{dt} = y_N, \tag{61}$$

$$\frac{dy_N}{dt} = -c_1(y_1)y_{N-1} - c_2(y_1)y_{N-2} - \cdots - c_1(y_1) + A(t). \tag{62}$$

Note that

$$\xi = (0, 0, \cdots, 0, \xi) \tag{63}$$

so that only one Gaussian variable is needed per time step. More complicated equations than eq. (59) are just as easily handled; e.g., equations nonlinear in the derivatives and equations with an explicit time dependence on the left.

V. PARAMETERS FOR VECTOR SDE ALGORITHMS

To second order, the equations for the vector algorithm parameters are identical with those of a single equation. Thus, the parameters given earlier in eq. (23) may be used for a vector $2_O2_S1_G$ scheme.

To third order, one new equation enters. All the eqs. (24) to (38) hold except that eq. (36) splits into two (because of the difference of mixed derivatives):

$$\sum_{i=2}^{3} A_i \sum_{j=1}^{i-1} \beta_{ij}L_{ij} = \frac{1}{6}, \tag{64}$$

$$\sum_{i=2}^{3} A_i \sum_{j=1}^{i-1} \beta_{ij}L_{jj} = \frac{1}{6}. \tag{65}$$

The one degree of freedom of the one-variable $3_O3_S2_G$ algorithm is now removed, but a solution might still exist. Unfortunately, no real solution can be found regardless of whether there are 2, 3, or 4 Gaussians.

In order to find a third-order algorithm, it was necessary to consider a $3_O4_S2_G$ procedure, that is, to add a stage. This leaves many degrees of freedom—in fact enough so that the deterministic part of the equation could be satisfied to fourth order with one degree of freedom left. Actually, these extra degrees of freedom only exist if one assumes [in the pattern of eqs. (39) to (40)] that

$$A_1 = 0, \tag{66}$$

$$L_{0i} = L_{ii} = \alpha_i, \qquad i = 2, 3, 4. \tag{67}$$

Since the parameter equations are nonlinear there are multiple families of solutions. We have not explored all the branches, but have looked particularly at a branch for which $\alpha_4 = 1$. This implies that $\lambda_4 = (1, 0, \cdots)$. In Table II we present two parameter sets which can be used for the $3_O4_S2_G$ algorithm with 4_O deterministic-part accuracy. We have the parameters for three other families of solutions but have not presented them because some parameters are large, i.e., $\gtrsim 5$. The degree of freedom is in the relation between λ_{11} and λ_{12} which is

$$\left(\sum_{i=2}^{4} A_i\alpha_i\beta_{i1}\right)\lambda_{11} + \left(\sum_{i=2}^{4} A_i\lambda_{i2}\beta_{i1}\right)\lambda_{12} = \left(\frac{1}{6}\right) - \sum_{i=3}^{4} A_i \sum_{j=2}^{i-1} \beta_{ij}L_{ij}. \tag{68}$$

All the parameters in this equation except λ_{11} and λ_{12} are determined by other equations. In Table II we present two solutions, for which $\lambda_{12} = 0$ and $\lambda_{11} = 0$, respectively.

From the point of view of computer time, a $3_O4_S2_G$ algorithm might be faster than a $3_O3_S4_G$ algorithm (if the latter existed); i.e., an extra functional evaluation may be faster than generating more Gaussians. For problems in which the effect of noise is small (small ξ) the $3_O4_S2_G$ algorithm, being fourth order in the deterministic part, would be more accurate.

VI. CONCLUSION

In Section IV, we considered various $k_Ok_Sm_G$ algorithms and found that for the one variable problem there were 5, 14, and 29 equations to be satisfied for $k = 2$, 3, and 4, respectively. On the other hand, the numbers of parameters available to satisfy these equations are maximally $(k + 1)^2 = 9$, 16, and 25, respectively. It appears that the number of equations is increasing more rapidly than the number of parameters. [The situation is complicated for a number of reasons: (i) assumptions like eqs. (39) and (40) seem capable of reducing the number of equa-

Table II—Parameters for a $3_O4_S2_G$ algorithm appropriate to vector SDEs

A_1	0.0	A_2	0.644468
A_3	0.194450	A_4	0.161082
β_{21}	0.516719	β_{31}	−0.397300
β_{32}	0.427690	β_{41}	−1.587731
β_{42}	1.417263	β_{43}	1.170469
λ_{01}	1.0	λ_{02}	0.0
λ_{11}	0.0	λ_{12}	0.271608
or			
λ_{11}	−0.567253	λ_{12}	0.0
λ_{21}	0.516719	λ_{22}	0.499720
λ_{31}	0.030390	λ_{32}	−0.171658
λ_{41}	1.0	λ_{42}	0.0

tions considerably; (*ii*) even when there are sufficient parameters real solutions do not always exist; and (*iii*) vector SDEs produce more equations with the same number of parameters.] To achieve a third-order algorithm for sets it was necessary to use four stages, and our failure to find a 4_O4_S algorithm in Section III probably means that at least a fifth stage is necessary. A similar situation occurs for deterministic equations, in which case more than k stages are needed when the order of accuracy is $k \geq 5$.[4]

Higher order methods can be achieved in other ways than by increasing the number of stages. One possibility, suggested in I, would be to adapt iterative multistep methods, e.g., of the Adams-Moulton type.[3] Another approach would be to use implicit Runge-Kutta methods in which later stage g_j's are used in earlier stage g_i's.[4] An l-stage implicit method would then require the self-consistent solution of l nonlinear equations for the $l\,g_i$'s at each time step. For mildly nonlinear SDEs and small fluctuations of the stochastic parts, this could be more efficient than larger stage methods. It is known that deterministic implicit methods are capable of achieving kth order accuracy with fewer than k stages, so this could well be the case for SDEs also.

It is our hope that the work presented here and in I, besides providing some practical schemes for integrating SDEs, will stimulate further research on this interesting and important topic.

VII. ACKNOWLEDGMENT

H. S. Greenside would like to thank P. C. Hohenberg for bringing the subject matter of this paper to his attention.

APPENDIX

We briefly present further details of the solution of the equations for the parameters of the $3_O3_S2_G$ algorithm to illustrate the procedures which one follows in cases of higher order or more stages.

The seven independent equations to be solved, after assuming eqs. (39) and (40), are eqs. (24) to (28), (36), and (37). The eight unknowns may be taken as α_2, α_3, A_2, A_3, β_{32}, λ_{01}, λ_{11}, and λ_{12}. Other λ parameters are given by

$$\lambda_{21} = \alpha_2, \tag{69}$$

$$\lambda_{31} = \alpha_3, \tag{70}$$

$$\lambda_{22} = +(\alpha_2 - \alpha_2^2)^{1/2}, \tag{71}$$

$$\lambda_{32} = \pm(\alpha_3 - \alpha_3^2)^{1/2}. \tag{72}$$

(The use of the negative sign for λ_{22} changes all signs for the λ_{i2}'s, and is a trivial modification equivalent to changing the sign of Z_2.)

Equations (24) to (26) may be solved for A_1, A_2, and A_3 in terms of α_2 and α_3. Setting $A_1 = 0$ leads to

$$\alpha_3 = \frac{3\alpha_2 - 2}{6\alpha_2 - 3}. \tag{73}$$

By eqs. (71) and (72) we see that both α_2 and α_3 must be between zero and unit, for a real solution, which, coupled with eq. (73), implies inequality eq. (41).

The remaining parameters are solved for as follows: eq. (27) yields β_{32} in terms of parameters now dependent only on α_2; eq. (28) dictates $\lambda_{01} = 1$, which is true for every algorithm; eq. (36) is a linear equation for λ_{11}; and eq. (37) is a binomial equation for λ_{12}.

REFERENCES

1. (a) D. J. Wright, "The Digital Simulation of Stochastic Differential Equations," IEEE Trans. Auto. Control, *19* (February 1974), p. 75.
   (b) N. J. Rao, J. D. Borwankar, and D. Ramkrishna, "Numerical Solution of Itô Integral Equations," SIAM J. Control, *12* (February 1974), p. 124.
   (c) T. Yamada, "On the Approximation of Solutions of Stochastic Differential Equations," Z. Wahrscheinlichkeitstheorie Verw Gebiete, *36* (1976), p. 153.
   (d) W. Rümelin, unpublished work.
2. E. Helfand, "Numerical Integration of Stochastic Differential Equations," B.S.T.J., *58,* No. 10 (December 1979), pp. 2289–99.
3. A. H. Stroud, *Numerical Quadrature and Solution of Ordinary Differential Equations,* New York: Springer-Verlag, 1974.
4. J. C. Butcher, "Coefficients for the Study of Runge-Kutta Integration Processes," J. Australian Math. Soc., *3* (1963), p. 185.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Using Documentation as a Software Design Medium

By S. D. HESTER, D. L. PARNAS,* and D. F. UTTER

(Manuscript received March 25, 1981)

This article describes a software design method based on the principles of separation of concerns and information hiding. The principle of separation of concerns is used to structure the design documentation, and information hiding is used to guide the internal design of the software. Separation of concerns requires that design information be divided into clearly distinct and relatively independent documents. The design documents are the main products of the initial design phase, and are carefully structured to (i) expose open issues, (ii) express design decisions, and (iii) ensure that information is recorded in a way that allows it to be readily retrieved later. Information hiding is used to design software that is easy to change. We have applied many elements of the design method to the development of the No. 2 Service Evaluation System (SES), a multiprocessor data acquisition and transaction system. Our experiences in applying the design method are described, and some examples are included.

I. INTRODUCTION

This article describes a software design method based on the principles of separation of concerns and information hiding. Software design documentation is the medium used to apply the principles.

The expected benefits of the design method are as follows:

(*i*) *Ease of change.* System functions that are likely to change are identified and information hiding is applied to minimize the amount of software affected by a change in these functions.

(*ii*) *Control of the information about the functions of the system.* A carefully structured requirements document is to be maintained throughout the life of the project.

* Currently on leave from University of North Carolina. Present addresses: IBM Federal Systems Division and Naval Research Laboratory.

(*iii*) *Ordering of the development steps to meet the project objectives.* Documentation of the useful subsets of the system and the dependencies between the software modules serve to guide the scheduling of the development effort.

(*iv*) *Making the agreements between developers explicit.* Misunderstandings are avoided and a smoother system integration is achieved by documenting the interfaces between the software modules of individual developers.

This article provides an overview of the design method as adapted to a particular class of software systems, and suggests guidelines for applying the design method. Related work on software design methodology has been reported in Refs. 1, 2, and 3. The Naval Research Laboratory has reported related work on a real-time system in Refs. 4, 5, and 6. Examples and experiences are presented from our application of these principles to the design of the No. 2 Service Evaluation System (SES), a multiprocessor system performing data acquisition and transaction functions.

We will discuss the key design principles, the proposed design steps and associated documents, the guidelines for preparing each of the documents, and finally, our experiences in applying the principles.

II. A DILEMMA

We are concerned with the dilemma posed by the following two statements:

(*i*) In most software projects coding begins too early. Important design decisions about the functions of the system, the nature of its interfaces, and its maintainability are made as by-products of the coding process and do not receive the conscious attention and review they deserve.

(*ii*) When part of a project's time is invested in a preliminary phase (sometimes called a "concepts phase," "project definition phase," or "specification phase"), one sees little in the way of tangible results. When actual software design begins, the programmers do not use the products of the earlier phase and one has the impression that the time spent was wasted.

These views are held by the same designers at different times in their careers. After an experience without a preliminary design phase, the first viewpoint is espoused with great vigor. After an experience with a preliminary design phase the second viewpoint is held by most participants.

The design method described here attempts to resolve this dilemma by specifying that the preliminary design phases produce a carefully structured set of documents as the main product. The documents are the means to express design decisions, not an afterthought to be produced after the system development is completed. Since documentation is the main product of the design phases, it is important and must be produced with the same discipline and care with which code is produced.

The principles for organizing the documents are discussed in the following sections.

III. OUR KEY DESIGN PRINCIPLES

The method we are advocating for design and documentation is based on the principles known as separation of concerns[7] and information hiding.[1,2,3] Separation of concerns involves the division of information about a system into clearly distinct and relatively independent parts. A software system design can be better controlled if the information in design documentation is divided in accordance with separation of concerns. The complexity in a software system comes from the number of details that must be considered.[8] To do their jobs, the developers must deal with large amounts of information describing what the system is to do and how their work relates to the work of the other developers. If each design document contains types of information that are clearly distinct and relatively independent from the information contained in other design documents, then the users of the documents can easily determine which document should contain the information of interest.

Ease of change and enhancement of the software system is typically a major objective in adopting a formalized design method. The principle of information hiding can be used to guide the structuring of software to make specific types of changes easy to implement. Information hiding involves encapsulating information likely to change in moderate size software modules. This encapsulation limits the amount of software that must be modified when a change is made. The possibility of future change must be explicitly considered during the design process in order to apply information hiding. One cannot foresee all possible changes; however, by evaluating the possibility of change openly, at least the decisions about what is likely to change are made explicitly and one knows beforehand which functions are likely to be easy to change.

Separation of concerns and information hiding describe the same idea from two perspectives. For example, to fully separate the concerns about the different aspects of a design is equivalent to encapsulating all elements of each aspect, and hence, hiding the information about each aspect. We find viewing some issues from the perspective of separation of concerns to be helpful while other issues are better viewed from the perspective of information hiding. The division of the software documents into clearly delineated areas of coverage is conveniently viewed as separation of concerns; whereas, the determination of the information to be contained in a software module is viewed as information hiding.

We discuss a number of applications of these principles in the remainder of the article.

IV. DEFINITION OF TERMS

Before introducing the proposed design method, we define a few terms used in the paper. These terms have been used in a variety of ways in the literature; however, we attach the specific meaning described below.

Software system	- A multiperson (and typically multiversion) software development which is delivered and used as a unit.
Input data item	- A data item received by the system from a user or an external hardware device or system. An input may be used promptly in the execution of a function, as in the case of a parameter a user enters with the request for a report, or it may be stored in order to influence later operation of the system, as in the case of scheduling information used to control later execution of a function.
Output data item	- A data item displayed to the users of the system or sent to an external hardware device or system.
Event	- A stimulus to the system causing a function to be performed. Events may be internally triggered upon a change in the state of the system or they may be triggered by a signal from a user or an external hardware device or system. An example of an internally triggered event is the match of the clock time against stored scheduling information that initiates the execution of a function.
Function	- The algorithms, rules, or relationships applied by the system in response to events in order to determine the values of one or more output data items and/or the display of the output data items to the user. We do not attempt to fix the size of a function at this point, and discuss both large and small functions. Later, when dealing with module decomposition, we recommend subdividing functions until they are small enough to be developed by one person in a limited period of time.

Module	- A piece of software and the associated documentation which together contain all the information about some function(s) or part of a function. Each module is small enough to be developed by one person in a limited period of time—generally one to three months.
Access routine	- A piece of software in a module which can be invoked by software in other modules to perform some portion of the module's functions. A subroutine in a data base module which is used by software in other modules to access the data base would be a typical example of an access routine. An access routine is not restricted to being a subroutine. A macro or the top level software controlling a process (which we call the main loop of a process) could also be an access routine.
Process	- A set of access routines whose execution sequence is prescribed. The execution of a process may overlap in time with the execution of other processes in the system. In the No. 2 SES we have chosen to restrict the relationship between modules and processes for simplicity. A module does not encompass the main loop of more than one process. This restriction results in some small modules, but it reduces the potential confusion in the relationship between modules and processes.

A module is the basic unit of development and change in this design method. Each module is defined according to the information-hiding principle (i.e., containing all information about some functions) in order to localize the software affected by a change in a function. The limitation to one person doing the development eliminates the need for multiperson communications during the internal development of the module, and the time limitation restricts the amount of work necessary to recode the module in the event of a change.

The usual approach to specifying a function is to describe input, processing, and output, in that order. The above definition of a function leads to function specifications organized around the system outputs. The values and display of systems outputs are specified in terms of algorithms, rules, relationships, inputs, and events. We have found this approach encourages more precise specification of system functions, and reduces the tendency to bias the specification toward a particular implementation.

Table I—Software design documents

Document	Scope
Requirements specification	Everything the software designers need to know about the system.
Module decomposition	The division of the system into modules.
Module dependency	Tabulation of the other modules which each module uses to perform its functions.
Process structure	Groupings of access routines that have prescribed execution sequences.
Resource allocation	System resources used by each module.
Module interface	Everything another programmer needs to know to correctly use the functions provided by the module.
Module design	Description of the internal design of a module.
Test plan	Description of the subset of the requirements which will be tested and the strategies for performing the tests.

V. APPLICATION OF SEPARATION OF CONCERNS TO THE DESIGN PROCESS

We propose dividing the information about the system into the set of documents listed in Table I. This division is based on what we believe are fundamentally separate concerns in the design of a software system. These concerns continue to be relevant throughout the system's life so the documents should be kept up to date.

The relationship of the proposed documents is shown in Fig. 1. The arrows indicate the principal flow of information required for the preparation of each document. Many inputs are, of course, necessary for the preparation of the requirements specification; however, discussion of the many sources of information is beyond the scope of this

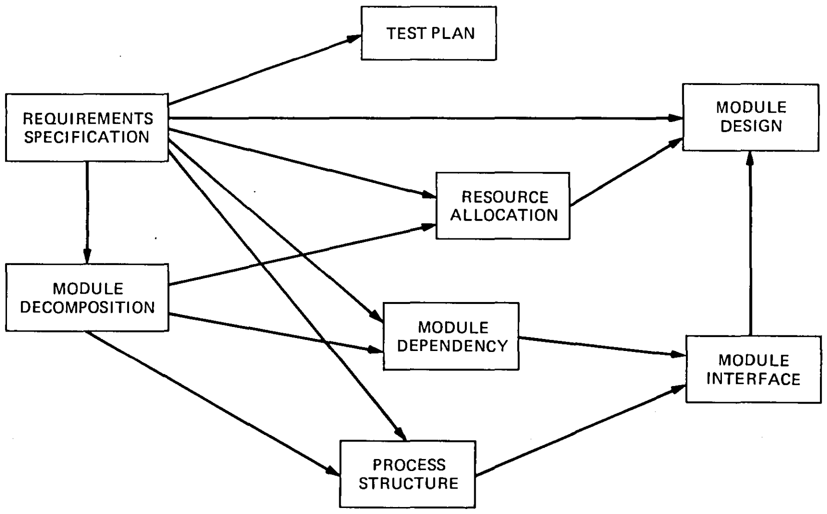


Fig. 1—Relationship of proposed documents.

article. Several feedback paths exist, but have been omitted for simplicity.

The module decomposition, module dependency, process structure, and resource allocation documents collectively constitute an overview of the software structure. Since the first draft of each of these documents can be prepared before any decisions are made about the implementation environment for the system, the overview provided by the documents can provide useful guidance in choosing the processor architecture and operating system.

One could prepare a single-design overview document with chapters dealing with module decomposition, module dependency, process structure, and resource allocation. Similarly, one could have individual module overview documents containing module interface and module design chapters; however, care must be taken to avoid mixing concerns between the chapters in an overview document. We believe the concerns are less likely to be mixed if separate documents are prepared as shown in Table I.

The discussion in the remainder of the paper may appear to represent the design process as a linear progression through the design steps. In fact, experienced developers know feedback to earlier steps occurs repeatedly during the design process. We recognize feedback must occur; however, the feedback should be recorded in the proper document. For example, modifications to the requirements should be found in the requirements specification and not in a note in a module design document. We are aware of the price of adhering to this discipline; however, we feel that the cost of neglecting it is even higher.

We will next present the scope, use, and design considerations for each document. By design considerations, we mean guidelines for the software design associated with the step covered by the document. The guidelines for preparing each of the documents are presented later.

5.1 Requirements specification

5.1.1 Scope

This document, together with the documents it refers to, should contain everything one needs to know to build an acceptable software system. All significant externally visible behavior of the software should be constrained to acceptable alternatives in this document.

5.1.2 Use

The requirements specification can be used both for communicating with the system user and to guide the software design. When the requirements specification has been reviewed by the system users, and the developers and users agree on the contents of the document, it can serve as part of a contract. Some of the many uses of the document in guiding the software design will be discussed in the following sections.

5.1.3 Design considerations

Many decisions about the functions of a system are made during the preparation of the requirements specification. Recommendations are made later in the paper for structuring the document in a way that encourages systematic resolution of the issues associated with specification of the system functions. The framework of the requirements specification is intended to stimulate addressing requirements issues early in the system design.

Preparation of the requirements specification should start during the earliest stages of a project. The document then evolves as the project proceeds—beginning as a rather sketchy skeleton and gradually filling out until it is complete. Gaps in the requirements specification serve to highlight the open issues.

5.2 Module decomposition document

5.2.1 Scope

The module decomposition document records the division of the software system into modules.

5.2.2 Use

It should tell readers the way the software has been structured and direct them to the appropriate component and its documentation. This document should eliminate any need to search through more detailed documents to find out which one of those documents contains a specific piece of information.

5.2.3 Design considerations

A number of the popular software design methods focus attention on the module decomposition step.[9,10,11] The module decomposition document could be prepared for a decomposition obtained by any of a number of the popular methods; however, since a major objective of the design method is to design for change, we believe module decomposition is best accomplished by applying the principle of information hiding.[1]

Module decomposition according to the principle of information hiding involves systematically hiding in a module all the information about each function defined in the requirements specification. The first step in selecting functions to hide should be to examine the "expected changes" chapter of the requirements specification (see Table II). Any function that is likely to change should be hidden in a module.

After doing the decomposition indicated by the Expected Changes chapter of the requirements, some functions may not yet be associated with modules, and many modules may still be too large. We have approached the further selection of functions to be hidden in modules from two opposing directions. The first involves decomposing the major functions into progressively smaller functions until we judge the implementation effort to be within the constraints we have set for a module. For example, a major function of displaying stored data to the users is broken down into a number of individual reports that can be independently requested. Each report may then be decomposed into a function that controls dialogue with the user, a function to compute output data, and an output-formatting function. As discussed earlier, all major functions can be defined in terms of the information required to determine the output data items associated with the function; e.g., "the prompts to the user," "the output data items required for the report," and "the format of the report."

Continuing subdivision of a complex function may eventually lead to subfunctions that do not directly control an output data item. We introduce the notational convention of intermediate data items to describe the interaction between such subfunctions. Subfunctions and intermediate data items are further discussed in the guidelines later in the article for preparing the data items chapter of the requirements specification. When a function is subdivided, the resulting parts should be chosen so they are likely to change independently.

Another approach to selecting functions to hide in modules is to identify common-use functions from the requirements. One looks for services required repeatedly by a major function or by several major functions. The common-use functions are hidden so they can be changed without affecting other parts of the system. For example, data storage and retrieval services are often required throughout a system. In support of a report generation system, one might identify a common function controlling dialogue with the user, common data base access functions, and a common output-formatting function. Having identified as many common-use functions as possible, one constructs the major functions from combinations of the common-use functions and whatever single-use functions are necessary.

The approach of identifying common-use functions has the advantage of ensuring uniformity in the user view of the system, and it reduces the redundant development of similar functions by several programmers. A disadvantage of this approach is that no single major function can be completed until development is completed on a number of modules hiding common-use functions. On balance, we favor the development of modules that hide common-use functions.

Most experienced software developers will quickly identify a number of common-use functions that should be hidden in modules. Database functions, device interfaces, output functions, and user interfaces are typical of the functions that will generally be readily identified. During implementation of the modules, the developers may identify the need for additional common-use modules which can be used by two or more developers. Some guidance is available for identifying potential common-use modules;[3] however, good communications among developers continues to be necessary to avoid the development of the same tools by two or more developers.

Access routines in a module may be used by several other modules. For example, portions of a device handler module may be used by a data acquisition module, while other parts may be used by a testing module. Neither of the modules using the device handler would contain any information about the device since that would all be hidden in the device handler.

Module decomposition involves breaking down a multiperson development into individual work assignments; therefore, the initial decomposition must be refined when the implementation effort can be better estimated. If a module is found to require only a small development effort, we generally do not try to merge it with another module since there is not a great deal of overhead associated with having additional modules. If, on the other hand, a module is found to require more development effort than one person can complete within the allowed time limit, then it should be subdivided into two or more smaller modules as described above.

Since the decomposition is based on the functions described in the requirements specification, the decomposition should be independent of the implementation chosen for the modules, with the exception that the amount of implementation effort limits the size of the modules.

The key guideline to keep in mind throughout the decomposition process is to always define a module in terms of the information hidden by the module.

5.3 Module dependency document

5.3.1 Scope

This document specifies for each module which access routines from other modules it must use to perform its function.

5.3.2 Use

The module dependency hierarchy determines which other modules must be available for a module to perform its functions; therefore, the document can be used to identify the modules necessary to provide the required subsets (i.e., the portion of the system to be developed first if time and staffing limitations prevent developing all functions).

The module dependency document is most valuable during the early stages of the design when the development order for modules and the users of each module must be identified in order to prepare and review the module interface documents. Once the module interface documents have been prepared, this document continues to serve as a summary document derived from the module interface documents.

5.3.3 Design considerations

The requirements specification, together with the module decomposition, defines for each module which other modules must be used to perform the required functions. For example, a data acquisition module which is required to obtain data from a device must use the device handler module. Similarly, if the data is to be stored, then the data acquisition module must use a database module. One must systematically examine all of the functions of a module as prescribed in the requirements specification to obtain a list of all of the modules used. No decisions about the implementation of the modules are necessary in order to perform this step. In the case of the data acquisition module, we only need to know that any access to the device must be through the device handler module and any access to the database must be through the database module.

5.4 Process structure document

5.4.1 Scope

This document specifies the groups of access routines having a prescribed execution sequence. The execution of two access routines in the same process are always clearly sequenced, whereas access routines in separate processes can be executed in arbitrary order.

5.4.2 Use

The process structure is a necessary input to the design of the module interfaces since the methods for interfacing between processes are generally different from those used within a process. The groupings of access routines into processes determine which module interfaces are between modules within a process and which cross process boundaries.

The process structure is a major determiner of the potential for exploiting extra processors.

5.4.3 Design considerations

The process structure for a system can largely be defined by determining which functions in the requirements specification can overlap in time and which must be executed in a specified order. A maximum number of processes is obtained if the only modules grouped into processes are those for which the execution order is prescribed in the requirements specification. All modules for which the execution order is not specified are separated into independent processes. The choice of the maximum number of processes would yield a more flexible design than one with fewer processes; however, the overhead associated with administering processes may cause one to choose a design with fewer than the maximum number of processes.

Additional guidance for defining the process structure is given in Refs. 12 and 13.

5.5 Resource allocation document

5.5.1 Scope

The resource allocation document summarizes the system resources used by each module. The tabulation can include any resource potentially causing a bottleneck in system performance. Resources of concern typically include CPU real time, memory, disk real time, disk space, and communications channels.

5.5.2 Use

This document can be used by module developers to judge the proper level of attention to give to resource usage in the design of each module. If each developer adheres to the resource budget for their module, then the overall system should perform properly.

The document is useful for ongoing tracking of resource usage after the initial design is completed. When enhancements to the system are evaluated, this document can be used to assess potential impact on resource usage.

5.5.3 Design considerations

The requirements specification and module decomposition document provide the basis for determining the frequency of invocation of a module and for estimating the likely resource usage for each invocation. Unfortunately, the initial estimate of resource usage must be based on a rough conception of a possible implementation, and therefore, the estimate may be inaccurate. If a module is likely to consume a large fraction of the system resources, then alternative implementations should be evaluated early in the system design to refine the estimate of likely resource usage.

Substantial effort should be invested in early study of resource allocation. We have seen several projects fail or be severely set back by encountering serious resource usage problems late in the design process. If resource needs are documented early in the project, provision can be made for adequate system resources and for careful design of the modules consuming most of the resources.

5.6 Module interface documents

5.6.1 Scope

Each module interface document describes the aspects of module behavior visible to other programmers using the module. Aspects of behavior visible to the system user are fully documented in the requirements specification and should not be duplicated in the module interface documents. For example, an interface document for a device handler module describes the means for invoking the software, the return values, and any modifications to stored data resulting from invoking the module. No messages exchanged with the device are described.[6]

Everything in the module interface document should be true for all acceptable internal implementations of the module, and should not be biased towards any particular implementation.

5.6.2 Use

The module interface documents settle the agreements between programmers about how cooperating modules will interact. Each module interface document should contain everything another programmer needs to know to develop software that interacts with the module. Clear documentation of agreements between programmers is very important on a multiperson development for smooth integration and ease of maintenance.

5.6.3 Design considerations

In order that the module interface documents adequately describe the means of communicating between modules, the implementation environment (e.g. operating system and programming language) must be selected before the documents can be completed.

5.7 Module design documents

5.7.1 Scope

The module design documents are intended to record the decisions made in the internal design of the module. Such topics as data structure design, resource usage, data buffering strategies, subroutine structure, and control logic are appropriate for the module design documents.

5.7.2 Use

The module design documents are used to guide a review of the software design and to inform future maintainers of the module of the reasons why the particular design was chosen.

5.7.3 Design considerations

Several design methods could be used for the internal design of modules.[7,9,11] The design method influences programmer efficiency and the maintainability of the module; however, since the design method that we are advocating encourages limiting the size of the modules, an entire module could be discarded and recoded if it proved to be unmaintainable.

The principle of information hiding can be used in the internal design of a module just as it is in the overall system design. If information hiding is applied in the internal design of a module, then the effects of change should be isolated to a portion of the module, and less effort should be required to maintain the module.

5.8 Test plans

5.8.1 Scope

All of the requirements stated in the requirements document are testable, but in practice we can only test a subset due to time limitations. Test plans describe the approach to be used to verify a specified subset of the requirements document.

5.8.1 Use

A separate test group should prepare and execute the test plan. Since the test plans represent only a subset of the total software requirements, the test plan should be maintained as private information within the test group to ensure that software is not written so that it will only pass the test. Even developers with the best intentions may fall into the trap of focusing on the functions to be tested.

5.8.3 Design considerations

The choice of how large a subset is to be tested must be influenced by the potential cost of not finding bugs versus the project limitations in development staff and time. For example, a medical control system could have a very high cost associated with a residual bug in the system. The test plan should explain which potential errors are considered particularly important to detect and what the testing strategy is to detect those potential errors.

VI. DOCUMENTATION PRINCIPLES

Before providing specific guidelines for preparing each of the documents, we will introduce some principles to guide the preparation of any software documentation.

6.1 General principles

(*i*) Write a specification for every document. Five questions should be answered in each document specification:

- Who will use the document?
- What will they use it for?
- What do they know before reading the document?
- What should they know after reading the document?
- What sources are there for prerequisite knowledge?

Note that a document specification is not an outline of the document. Instead, it identifies the audience, the perspective of the audience, and the way that the audience will use the document.[14]

(*ii*) When writing a document, a chapter in a document, a section in the chapter or a paragraph in a section, formulate the questions to be answered before starting to answer them. Writers often confuse organizational issues with issues about the substance of the article; to avoid this confusion, we express the organization in terms of questions rather than answers.

(*iii*) Design documents using the principle of information hiding. Every section of the document should deal with a clearly defined and limited aspect of the system; one should not yield to the temptation to include other "relevant" facts in the same section.

(*iv*) Use formalism to describe design decisions and natural language for introductions, motivation, justifications, etc. Formalisms, when appropriately designed and used, can greatly increase the precision and compactness of a description. Formal descriptions are more easily checked for completeness and consistency. Natural language is preferable to formalism for describing motivational material. There is never a need to describe the same thing both ways.

(*v*) If there are a large number of descriptions containing the same information, restructure the document so common aspects are described only once. It is essential to make the structure explicit or the reader will not know where to find the information that has been pulled out of the individual descriptions to avoid repetition. Repetitious documentation is both time wasting and a cause of errors due to inattentive reading.[15]

6.2 Stylistic rules

(*i*) Eliminate all statements containing little information. If the negation of a sentence would rarely be uttered, the sentence itself communicates very little.

(*ii*) Replace oblique statements with direct statements. Often sentences containing little information are there as indirect ways of saying something else. If something else needs to be said, say it directly.

(*iii*) Avoid saying the same thing twice. If you say the same thing two different ways because neither is perfectly clear, you decrease the clarity because readers will wonder about differences. It is better to spend the time necessary to say it clearly once. However, remember purpose is different from method, and a decision is different from a reason. Stating the intent behind a design and stating the design is not saying the same thing twice.

(*iv*) When describing a program do not confuse its effects with its intended use. A program may do A and be used to accomplish B, but we often mix A and B in a way that makes it unclear what the program itself actually does.

(*v*) Make the significance of a design decision more explicit by stating the alternatives excluded by the decision. We often read about designs with a "ho hum" feeling because we are not made aware of the significance of the decisions.

(*vi*) Do not justify things in terms of principles nobody could be against. State precisely what pragmatic benefits will result.

6.3 Diagrams in program documentation

Pictures have been hotly debated as a means of documenting programs. If a program is clearly understood, it can be described precisely in terms of predicates and states or in terms of mathematical functions. Pictures tend to be quite imprecise as a means of documentation. On the other hand, pictures are quite useful as a means of introducing someone to a program he does not yet understand. Pictures should be used as introductory material but never as the binding documentation.

When pictures are used, precision in drawing the picture is necessary. Many computer system diagrams are confusing and easily misinterpreted because there is no precise meaning given to the symbols used. Often the same symbol is used to represent a program, a data structure, a hardware device, and a user, all in one diagram. If each picture is accompanied by a legend, there will be less of this confusion.

6.4 Review procedures

Effective review of the documents serve to verify the correctness of the documents and to ensure that they are understandable. The following guidelines can help one achieve effective document reviews.

(*i*) The selection of the reviewers for a document can be approached at the following levels depending upon one's objectives.

(a) The user of the software is an obvious choice for a reviewer. This would be the system user for the requirements specification, and the software developer who will use the module in the case of a module interface document. The user has a clear interest in the proper operation of the software, and hence, has a reason to do a thorough review of the document.

(b) A developer other than the one who prepared the document can be given the work assignment of reviewing the document. This so-called "buddy system" can result in someone else in the development group who is responsible for the correctness of the document and who is prepared to defend it. An additional benefit of the "buddy system" is the cross-knowledge gained within the development group. This cross-knowledge can be helpful when task reassignments are necessary.

(c) A person outside the project can be brought in to review the document. This reviewer will uncover omissions presumed to be common knowledge by those closer to the development. The outside reviewer is also a good choice for reviewing the overall set of documents for consistency.

(*ii*) The reviewer should be asked to examine the document from a specific perspective. For example, an expert in the system outputs should be asked to verify that section of the requirements specification. A questionnaire can be used to ensure reviewer will consider specific issues. Such a questionnaire should be prepared by the person who is directly concerned with the correctness of the document.

(*iii*) The reviewer should be asked to provide input in a comments section of the document. The reviewer should sign off on the document and note the areas of the document with which they were chiefly concerned. A record is then available of who has examined the document. The reviewer can be consulted later if issues arise that they may have considered.

6.5 Inclusion of justification material in the documents

Arguments can be made both for and against the inclusion of justification material in the documents to record why decisions were made. On the one hand, inclusion of a justification section in each document encourages the writer to record the reasons for making each decision at the time the decision is made. The reader is also more likely to read the justification material if it is included in the primary documents.

On the other hand, justification material can be quite verbose and its inclusion can swell the size of the document to the point that it becomes unwieldy, and the potential users of the document are discouraged from reading the document by its sheer bulk. The use of separate justification documents (referred to in the primary design documents) encourages clear separation of the concerns between what was decided versus why it was decided.

Faced with this dilemma, we have chosen to use separate justification documents for all of the documents, except the module interface and module design documents. The documents dealing with the whole system are quite large—particularly the requirements specification. Inclusion of justification material in these would make them excessively bulky and forbidding. The individual module interface and module design documents are typically only a few pages in length so the inclusion of justification material does not make them excessively large. A principal part of the module design document is, in fact, an explanation of the strategies used in the design.

VII. DOCUMENT PREPARATION GUIDELINES AND EXAMPLES

In this section, we provide preparation guidelines for each of the documents.

The guidelines and examples for the requirements specification are more detailed than for some of the other documents; however, the other documents may be of equal or greater importance for a particular project, and some of the other documents may require more effort to prepare. For example, module interface and module design documents are prepared by each software developer, so the collective effort in this area is quite large.

7.1 Requirements specification

The guidelines we have evolved for preparing the requirements specification for the No. 2 SES are based on a model project to prepare requirements for a real-time system.[4,5] The transaction-oriented nature of the No. 2 SES has motivated us to shape the guidelines to be more appropriate for our type of system.

We believe the requirements specification is most effective if it is a concise reference document. Formalisms are used wherever possible and tabular organization is frequently used. These techniques aid us in making the document concise. A concise document may require some additional effort for first-time readers to familiarize themselves with the formalisms and background material; however, the concise format is more efficient for day-to-day use, it eases updating of the document, and it encourages precision in the specification of requirements.

We organized the document into ten chapters which separate the concerns about the external behavior of the system. The chapter organization we have used is shown in Table II.

7.1.1 Introduction

The introduction should provide a guide to reading the document rather than an introduction to the system. Reference can be made to a separate system description for an overview of the system. We include a discussion of the organization of the document. Formalisms are explained and examples of the formalisms are given.

7.1.2 Input and output data items

The input and output data items are specified in several tables and

Table II—Chapter organization

Chapter	Contents
1. Introduction	A guide to using the document.
2. Input and Output Data Items	Definition of the input and output data items presented to the user and/or to external devices or systems.
3. Communication Protocols	Details of communications with hardware devices, software systems, and users. The user command syntax may be included here since it is a protocol.
4. User Transactions and Reports	Specification of the user interaction with the system, plus all scheduled and spontaneous reports generated by the system.
5. Performance Requirements	Constraints on how functions must be performed. We include timing, concurrency, accuracy, and storage considerations in this chapter.
6. Response to Undesired Events	What the software must do when undesired events occur.
7. Fundamental Assumptions	Characteristics of the system that are not expected to change.
8. Expected Changes	Changes expected or planned for future releases.
9. Required Subsets	Description of one or more subsets of the functions which would still constitute a useful system.
10. Glossary of Acronyms and Terms	Explanation of the acronyms and technical terms associated with the system.

supporting sections within this chapter. This chapter corresponds to a data dictionary.

Examples in Tables III, IV, and V illustrate the techniques we have used to specify the data items. These examples are arranged around a user transaction in the No. 2 SES needed to display some of the data stored about entities (telecommunications switches). The display consists of a set of output report data items.

We introduced the concept of data types to aid in the specification of the data items. Two criteria are applied to determine whether two data items are of the same type.

(*i*) The data items have the same set of values.

(*ii*) It is meaningful to use them in an assignment statement. For example, even though a computer identifier and a data link identifier might have the same set of values, using them together in an assignment statement would not be meaningful.

We bracket an item by "+" to indicate it is a data type. Our text processing system is used to audit the data types to ensure that every data item has a valid type and every type is used in at least one data item. Sample data types are presented in Table III. An enumerated type is a set of values. A list is a one-dimensional array.

Input data items are grouped into two classes—user inputs and device inputs. Output data items are grouped into three classes—report data items, interactive messages (errors, help, prompts, and positive feedback), and outputs to devices.

Input and output data items are bracketed by "/" and "//", respectively. All references to the data items use the bracketed notation.

Table III—Data types

Type	Values	Description
+boolean+	enumerated	boolean (two values)
	YES	yes
	NO	no
+ent-no+	integer, range (1–999)	entity number
+ent-state+	enumerated	entity state
	$NOT-DB$	not in database
	$READY$	ready
	$OFF-MAN$	off manual (user action)
	$OFF-AUTO$	off automatic (by program)
+list-4+	list of integers size 4 entries	list used for many reports
+nsc+	integer, 4 digits	network service center number

Wherever a bracketed item appears in the document, the reader can readily recognize that it is an input or output data item. When we change a data item, our text processing system is used to search for all occurrences of the bracketed item.

Separate tables are prepared for input and output data items. A description of each data item is provided in these tables, and the data type is specified. The specifications of the functions controlling each output data item are identified in the table for output data items. We have categorized the No. 2 SES functions as either user transactions or data acquisition functions, and have grouped the specifications of the functions into two separate lists. The %A-B% notational convention shown in Table IV is used to point into the two lists of function specifications. The A number points to the specification of the user transaction controlling the output data item, and the B number points to the specification of the data acquisition function. If both A and B are nonzero, then the output data item values can be set by either a user transaction or a data acquisition function; e.g., the entity state data item, //ENT-STATE//, in Table IV can either be set by the user or by a data acquisition function responding to an error event.

Some functions are made up of several parts that may change separately. Such functions should be described in terms of two or more subfunctions each of which is likely to change as a unit. To describe the communications between individual subfunctions, we have intro-

Table IV—Output data items

Data Item	Description	Functions	Data Type
//ENT-CLLI//	entity's text identifier	%1-0%	+char(13)+
//ENT-COM-PT//	common evaluation done on entity?	%1-0%	+boolean+
//ENT-LOOPMAX//	maximum loop on entity	%2-0%	+loop-no+
//ENT-NO//	entity's number	%1-0%	+ent-no+
//ENT-NPA//	entity's NPA	%1-0%	+npa+
//ENT-STATE//	entity's state	%3-8%	+ent-state+
//CALL-DISP//	call disposition	%0-12%	+disp+

duced the notational convention of intermediate data items (bracketed by !). An intermediate data item is not visible to the user and serves only as a notation for the output of one subfunction that is, in turn, used as an input to another subfunction.

The No. 2 SES function of determining the disposition of a customer call attempt is subdivided into two subfunctions. The first determines the initial disposition !INIT-DISP! by analysis of voice signals. The second subfunction uses stored information about the data source and the value of !INIT-DISP! to determine the final call disposition //CALL-DISP//. The two subfunctions are likely to change independently so they are hidden in different modules.

Intermediate data items are specified in a table similar to that used for output data items.

7.1.3 Communications protocols

The communications required with external hardware devices and software systems are specified in this section. External hardware devices include devices used for data acquisition, control, and/or display. If the communications with an existing device or software system are fully documented elsewhere, then the appropriate document can be cited.

The user command syntax rules can also be specified here because the syntax rules can be viewed as a protocol; however, the detailed command semantics should be specified in the chapter on user transactions.

7.1.4 User transactions and reports

All functions of the system visible to the user are specified in this chapter. These functions include computer operations, data base interactions, maintenance, user requested reports, scheduled reports, and spontaneously generated reports, such as equipment failure alerts. These functions are defined in terms of the input and output data items defined in the data items chapter. The same data items may appear in many reports.

A sample specification of a user transaction to obtain an output report is given in Table V. Some words of explanation may be needed to interpret the notation. The User Data Entry specifies the user input required to produce the report. The user enters the command "display entity-list" and selects which entities are desired. The default value for the entity selection is "all" entities. The output report is the collection of data items listed under Output Values. These are output for each entity displayed. The Transaction Effects section describes any changes to the state of the system or modifications to stored data resulting from performing the transaction; therefore, in this example,

Table V—Output report specification

```
Transaction name: entity list
User data entry: display entity-list /ENT-SELECTION/ = all
Output values:
  //CH-NO//
  //ENT-CLLI//
  //ENT-COM-PT//
  //ENT-NO//
  //ENT-NPA//
  //ENT-NSC//
  //ENT-STATE//
  //SCA-Port//
Transaction effects: none
Error messages:
  a. type error +ent-select+
  //E-ENT-SEL//
  b. constraint error +ent-select+ (entity not in database)
  //E-NO-ENT-SEL//
```

the Transaction Effects are "none" because this function simply reads the database and leaves no trace. The Error Messages section lists all messages specific to this transaction. The type error can be detected by examining the input data item itself, whereas constraint errors must be determined by checking the input data items against data stored in the system. The purpose and use of this report are not discussed here—that information is contained in the user guide.

7.1.5 Performance requirements

The preceding chapters of the requirements used the narrow definition of a function as being what the system was to do. The considerations of timing, concurrency, data volumes, data retention, and data accuracy are reserved for this chapter. A separate chapter is provided for these considerations because the requirements on what the system is to do may change separately from the performance requirements.

The process structure selected for the system must permit satisfying the sequencing and concurrency requirements described in this chapter.

7.1.6 Response to undesired events

Undesired events (UEs) prevent the software from performing the desired functions. Undesired events may be caused by input data errors, computer hardware malfunctions, or software errors. The number and variety of things to go wrong are quite large, and one has difficulty anticipating all possible problems when writing the requirements. One can begin by documenting all known UEs together with the desired response to each of the UEs. As the system is developed, additional UEs will be identified, and can be documented in this chapter.

The key objectives are (*i*) consciously consider the desired response to each UE, and (*ii*) document all UEs and responses to the UEs in one place.

7.1.7 Fundamental assumptions

This chapter consists of the list of functions and subfunctions that are not expected to change during the life of the system.

Discussions between the users and developers about which functions are not likely to change can be a useful part of the review of the requirements. These discussions will often involve challenges to the fundamental assumptions and may result in moving several of these functions to the next chapter. If the users wish to change functions appearing in this section after the system has been developed, they can expect such a change will require a large development effort since the developers did not have a reason to make it easy to change. Note that one does not try to make a function difficult to change. It will naturally become difficult to change if all information about the function is not carefully hidden in a module.

7.1.8 Expected changes

This chapter complements the fundamental assumptions chapter by providing a list of functions that are expected to change. The lists of functions in this chapter and in the chapter on fundamental assumptions should constitute a complete list of functions since a function is either likely to change or not.

Since many functions are likely to change at some point in the lifetime of a system, ranking the expected volatility of these functions may be helpful. Changes to some functions may already be planned for a future release of the software. Changes to a second group of functions may not be planned, but from historical data one knows functions of this type have often been subject to change. One may have no reason to expect changes in a third group of functions, but, on the other hand, there may be no firm reason to expect them to be stable. The additional cost of designing, with the expectation that most functions may change at some point, is modest in relation to the cost of later changing a function for which no thought had been given to possible change.

Functions that are likely to change should be carefully considered when the decomposition into modules is performed. At that stage, one should ensure that all information about each of these functions is completely contained in a module.

7.1.9 Required subsets

The functions of the system should be analyzed to determine what would constitute a minimal useful system. This minimal subset should be the first part of the system to be developed. If delays are encountered in developing the software, then a useful subset of the system can still be delivered on a timely schedule.

This chapter should define the minimal subset, plus any larger subsets which would provide additional valuable functions.

Developers are often under pressure to start development before the requirements are fully defined. If moderate risk can be taken, development can, in fact, begin once certain critical parts of the requirements are completed. If the functions in a minimal subset are defined and the performance requirements, fundamental assumptions, and expected changes associated with the minimal subset are defined and reviewed, then one can start the development of the minimal subset without excessive risk of wasting effort. A continuing source of risk arises from the possibility that the performance requirements for the full system will be more demanding than for the minimal subset. Common-use modules should be developed to accommodate the performance demands anticipated for the full system.

The people on a small project will often have excellent informal knowledge of the requirements before the formal document is written. In such a situation, the other design steps can be started, while the requirements specification is being written; however, issues that were thought to be resolved are often revealed to be incompletely defined when the attempt is made to write them down.

7.1.10 Glossary of acronyms and terms

This chapter is, of course, useful in supporting all of the document; however, it is particularly helpful in expanding the descriptions of the data items.

7.2 Module decomposition document

In the module decomposition document, we list the modules produced in the decomposition phase, and state what information is hidden in each module. Since a concise document is desired, we do not include any discussion of the strategy used to obtain the decomposition; instead, we refer to a separate justification document.

The modules are grouped into major classes to assist in locating a module dealing with a particular type of information and to assist in reviewing the decomposition for completeness. Most systems will have at least three classes of modules that hide (*i*) hardware information, (*ii*) user visible behavior, and (*iii*) software design decisions. We have found a somewhat larger number of module classes to be useful for the No. 2 SES. Our module classes are as follows: Database, Device Interface, Data Acquisition, User Input/Output, and Maintenance.

To review the completeness of the decomposition, we check to ensure that the information about each function in the requirements specification is stated to be hidden in some module.

A portion of our module decomposition document is shown in Table VI. Since the modules represent individual work assignments, we have found the document to be quite useful in tracking the progress of the work; therefore, we identify the author and reviewers of each module (the author and reviewers are not shown in Table VI to save space), and we have indicated the current status of the module. The abbreviations in the status fields are as follows:

NW	Module interface document has not been written.
MS	Module interface document has been written.
MSR	Module interface document has been reviewed.
DD	Module design document has been written.
DDR	Module design document has been reviewed.
C	Coding of the module has begun.
CR	Code for the module has been reviewed.

7.3 Module dependency document

The module dependency document should list all of the modules in

Table VI—Module decomposition document

Class	Module	Status	Information Hidden
Database	—	—	Modules providing access to the stored information.
	Call-record	CR	Storage and retrieval of call data.
	Bureau	CR	Storage and retrieval of bureau data.
Device interface	—	—	Modules providing communication to the call acquisition hardware.
	SCA*-handler	CR	Communications with SC/A devices.
	VDAS†-handler	NW	Communications with VDAS devices.
Input-output	—	—	Modules providing the user-required inputs and outputs.
	Term-interface	CR	Syntax rules for the user terminal command and feedback.
	Bureau activity	CR	Report summarizing system activity.
	DB-builder	C	The means for the user to alter contents of the No. 2 SES databases.
Data acquisition	—	—	Modules associated with the acquisition of call records.
	CR-generator	C	Control of the acquisition of call records.
	Classify-call	C	The computation of call dispositions from input voice and call data.
	CCT-sched	C	Scheduling the acquisition of calls from entities.
	Cr-proc	C	Control of the processing of call records.

* Signal converter allotter.
† Voice data systems.

the system, and for each of the modules a secondary list should contain all of the modules used by each module. An example module dependency document is shown in Table VII.

7.4 Process structure document

The process structure document should list all of the processes in the system and indicate which module encompasses the main loop of the process. Recall, we have restricted the scope of modules so no module encompasses the main loop of more than one process. We give the process the name of the module which encompasses the main loop.

The modules containing the main loop of a process are identified in Table VII. In fact, this simple table could serve as a process structure document; however, as we discuss later in the experiences section, the process structure document for the No. 2 SES includes a summary of interprocess communications.

7.5 Resource allocation document

The resource allocation document should contain a list of all modules and the amount of resources allocated to each. The total resources consumed when the module is invoked should be recorded including the resources consumed by any subordinate modules used.

When the module design documents are available, the resource allocation document can be derived from information in the module design documents, and the resources used by each access routine can be included. Thus, just as the module dependency document becomes a summary document once the module interfaces are written, this document also becomes a summary document once the module design documents have been prepared.

Table VII—Module dependency document

Module	Other Modules Used	Process Main Loop
Call-record	—	—
Bureau	—	—
SCA-handler	—	—
VDAS-handler	—	—
Term-interface	Bureau	—
Bureau-activity	Call-record	X
	Bureau	—
	Term-interface	—
DB-builder	Bureau	X
	Term-interface	
CR-generator	SCA-handler	X
	CCT-sched	—
	Classify-call	—
Classify-call	—	—
CCT-sched	—	—
CR-proc	Bureau	X
	Call-record	—

Table VIII—Standard form for module interface documents

Section	Contents
Module name	Name of module.
Author	Name of author.
Reviewers	Names of reviewers.
Information hidden	List of functions for which all information is contained in the module.
Access routines	The process and/or subroutines invoked by other modules to perform the functions provided by this module. The input parameters and return values for each access routine are defined using the conventions of the chosen programming language.
Effects on stored data	The stored data items modified by the invocation of the access routine are tabulated for each access routine. All changes in the system resulting from the invocation of each access routine should be recorded; therefore, the effects of all other modules used by the access routine to perform its functions must be noted.
Undesired events	Undesired events UEs occur when an access routine is not able to perform the requested function. All potential UEs for each access routine are listed and the return value is specified for each UE.
Other modules used	List of access routines in other modules that must be invoked in order for this module to perform its function.
Design issues	Discussion of the design issues which were considered in choosing the access routines to be provided, in choosing the input parameters and return values, and in defining the UE responses. Typically, the discussion of the choice of the UE responses is a major part of this section.
Review comments	The reviewer's comments and sign-off. This is the only section of the document that the reviewers may edit.

7.6 Module interface documents

Each module interface document should contain all the information another programmer needs to know to use the module. The document should specify how to invoke the module, what functions are performed by the module, and what return values and error indications are provided.

Since one of these documents will be prepared for each module, we used a standard form for the document to ensure that the same information is available for each module and to make it easier to find information in the document. A description of the contents of each section of our standard document is shown in Table VIII.

7.7 Module design documents

Each programmer should prepare a module design document before the code is written. The document deals only with the implementation of the functions of the module. The strategies the programmer used in the design are discussed. Typical topics include data buffering strategies, resources usage, subroutine structure, UE handling strategies, and program control flow. Pseudocode is used to show the control flow. Pseudocode is more readable than the "prose programs" often written when a programmer attempts to document what their software does.[16]

Design documents for modules that invoke a subordinate module should identify it, but not describe the internal design of the subordinate module.

7.8 Test plan

The test plan should list the tests to be performed together with the planned order of testing, testing strategy, and test environment. The test should be prepared directly from the requirements specification rather than from any lower level design documents. The test descriptions should refer to the requirements for the functions to be tested rather than paraphrasing the requirements since such paraphrasing may introduce subtle differences between the test objectives and the requirements.

VIII. DESCRIPTION OF THE NO. 2 SES DEVELOPMENT ENVIRONMENT

To provide the reader with some perspective on our experience with the design method, we will discuss the purpose, resources, and development environment of No. 2 SES.

The No. 2 SES collects data on the quality of service offered to the users of the telephone network. More specifically, it collects data on whether a customer-dialed call attempt succeeds or fails, and if it fails, the failure type. This data on network performance is the basis for an overall assessment of the adequacy of equipment provisioning and maintenance.

The No. 2 SES architecture, illustrated in Fig. 2, consists of a central

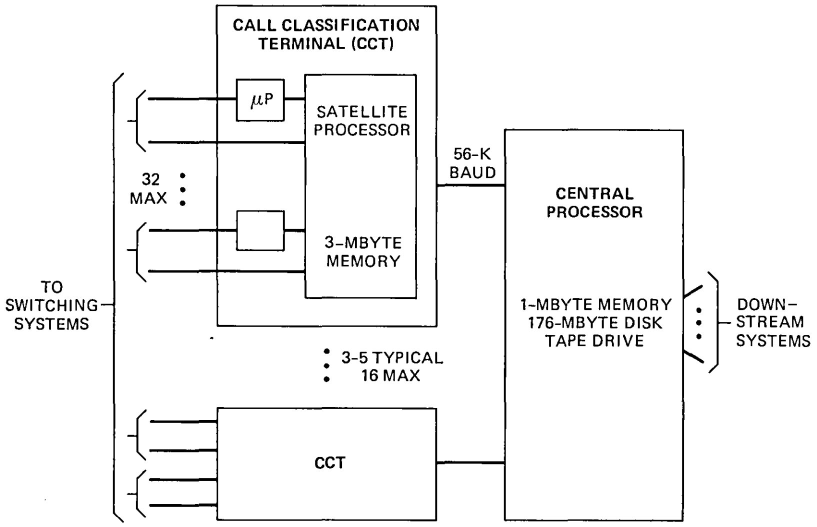


Fig. 2—No. 2 SES architecture.

processor and a number of satellite processors. The satellite processors are used for a signal recognition task requiring extensive computation. The satellite processors are, in turn, each supported by 32 microprocessor systems that extract data from analog telephone signals.

The development environment and schedules for the No. 2 SES project have much in common with a number of operations systems developed at Bell Laboratories over the past eight years. The system uses an enhanced version of the *UNIX** operating system and is programmed in the C language. The development group is modest in size—on the order of ten people. The development schedule is typical of the first development cycle of many operations systems. Feasibility was examined with a small staff beginning in 1978. The definition of functions and architecture were done in 1979, and the development group was fully staffed. Most of the coding was done in 1980, and the first field system became operational early in 1981.

No relaxation in the schedule nor increase in staffing was provided to aid the prove-in of the new software method. The effort invested in generating additional documentation was offset by effort saved during the system integration largely due to the clear expectations between developers fostered by the use of module interface documents. One of the authors was employed as a consultant between November 1979 and July 1980, and another worked full time on the requirements specification.

IX. EXPERIENCES IN APPLYING THE DESIGN METHOD TO THE NO. 2 SES PROJECT

9.1 General comments

Since we are writing this paper shortly after the No. 2 SES became operational in the first field application, we cannot present a full retrospective evaluation of the process; however, we will review some of our experiences thus far in applying the design method.

The development environment for the No. 2 SES has, of course, shaped our experience in using the design method. The environment of a small staff working against a tight development schedule offers advantages of flexibility and easy communication throughout the group, but on the other hand, there is little time or staff available to prepare detailed plans or to provide detached review and testing support. Most of the projects with a small staff with which we have been associated in the past have taken advantage of the easy communications within the group, and have correspondingly minimized the amount of documentation prepared. Such projects have often met

* Registered trademark of Bell Laboratories.

their initial objectives, but have been costly to maintain over their lifetime.

We were handicapped by adopting the principles after the development was well underway. We had to learn how to apply the principles, while the development was proceeding under the constraint of a fixed-project schedule. Much of the additional effort we have incurred in using the design method has been the result of the inefficiency of trying to learn the method, while the development was in progress. Hopefully, this article will help the reader understand beforehand what is involved in adopting this design method so the learning phase can precede the development rather than being concurrent with it.

9.2 Requirements specification

9.2.1 Relationship between the user guide and the requirements specification

We prepared a draft user guide late in 1979, and used it for a review of the proposed system features with an advisory panel of prospective Bell System operating company users. With the draft user guide as a starting point, we prepared a requirements specification in the first half of 1980. The first step in preparing the requirements specification involved recasting the general feature descriptions contained in the user guide into the more precise format described in this article. Many decisions were required to make the general descriptions more precise. Additional material was then prepared for the chapters on performance requirements, undesired events, fundamental assumptions, expected changes, and required subsets.

The overlap of the user guide and requirements specification has been a continuing source of concern. We now see how to have common-source text files form the core of both the requirements specification and the user guide. The requirements specification is divided into those parts visible to the user (e.g., reports) and those parts not visible to the user (e.g., communication protocols). The user guide is constructed by augmenting the user visible portion of the requirements specification with descriptive material to explain the intended uses of the functions. The use of common source files for both documents avoids the duplication of information that makes documents so difficult to keep up to date. We are only now starting to implement the use of common source files for the requirements specification and user guide.

9.2.2 Preparation of the document

Our late start on preparing the requirements specification dictated that it be written in parallel with the other design steps. During the preparation of the document, we depended upon the general knowledge of the requirements within the group and upon the draft user guide which described most of the output reports.

The sections on output data items and reports were prepared first. The database was defined from these sections. The input data items and user transactions were defined next. The communications protocols with external devices and systems were documented elsewhere, so preparation of these sections did not have high priority. The user command syntax was defined after the development was well underway.

Considerable time was consumed in choosing the organization for the document and the formalisms to be used. We now believe the basic chapter organization proposed here is sufficiently general to satisfy the need of a broad range of developments with minimal modifications. Much time was devoted to selecting the formalisms for describing the data items. Time can be saved by starting with simple formalisms to describe data items, such as the table descriptions illustrated here. If the simple formalisms prove to be cumbersome and verbose for a portion of the data items, then additional formalisms can be introduced to handle just the troublesome items. Our selective use of intermediate data items is an example of this approach. Similar selective use of formalisms is appropriate for user transaction and report descriptions. The more sophisticated notational conventions, such as modes and event tables in Ref. 4, yielded more concise descriptions of real-time functions than the simple formalisms we have used.

The size of the requirements specification is a major concern of many people who are considering this design method. Concern about size is appropriate when deciding how to staff the task of preparing the document, and when considering subdivision of the document; however, size should not be a consideration when deciding whether to prepare a requirements document. We do not know of a good alternative to adequately document requirements. There are many examples of projects that experienced serious trouble because they did not have well-defined requirements.

Issue 2 of the requirements specification for the No. 2 SES contains about 250 pages. The specification of 115 user transactions and reports occupies about 150 pages. About 50 pages are required to describe 800 input and output data items and 80 data types. The remaining 50 pages is mostly text. The number of user transactions and data items required for a system is a useful indicator of the potential size of the requirements specification.

9.3 Module decomposition

We established the module decomposition with surprising ease and unanimity among the people involved in the task. This decomposition has remained substantially intact through the rest of the development. Most of the later changes have involved the definition of additional modules as the requirements have been refined in areas that initially were vague.

Provided the requirements are clearly understood and the decomposition is approached by asking questions about what functions of the system should be hidden in modules, then we believe most people will generate similar module decompositions. The chief difference we have seen in the results of several people doing a decomposition is the degree to which the system should be broken down; i.e., should some function of the system be in a single module or should the function be divided into two or more modules. For example, we had no difficulty agreeing database access routines belong in a different module from data acquisition tasks; however, we could not definitely determine whether all database functions should be in one module or whether we should have several database modules. The appropriate size for a module is difficult to estimate early in the design; fortunately, it is easy to later decompose a module into two or more smaller modules if closer examination of the implementation indicates too much work is involved.

9.4 Module dependency

The module dependency for the No. 2 SES was rather simple. Only the database and user interaction modules were extensively used throughout the system. Most of the other modules had one or two users. If we were developing an operating system rather than an application based on an existing operating system, we might have found a much more complex module dependency. Since most commonly used utilities for our system are provided by *UNIX*, we only needed to develop a few common-use modules.

9.5 Process structure

We chose a process structure allowing very near the maximum concurrency permitted by the requirements. Most of the small processes resulting from the maximum partitioning reside on the central computer. The central computer has ample resources available in the initial versions of the system so the overhead of administering the additional small processes is acceptable, and the ease of maintaining the small processes is valuable.

The multiprocessor architecture of the No. 2 SES has caused us to have more complex interprocess communications than would have been necessary for a single processor system. Because of the complexity of the interprocess communications, we have found the inclusion of an overview of all interprocess communication in the process structure document to be useful as an aid in introducing people to the system design. This overview is derived from the module interface documents.

In some cases, implementation considerations have caused us to use two *UNIX* processes to perform the functions of one logical process. Interprocess communications sequence the execution of the two processes as prescribed in the requirements. For most purposes, these two *UNIX* processes can be considered to be one logical process.

9.6 Resource allocation

A small number of modules in the No. 2 SES consume most of the system resources, and we were careful to track the resource usage of these modules. We did not recognize the need for a resource allocation document until well into the development so the tracking has been informal. If resource usage had been more uniformly distributed among the modules, we would probably have been motivated to prepare a resource allocation document earlier in the development.

9.7 Module interface

We have prepared a module interface document for each of the modules in the system using the format illustrated earlier. These documents have been quite valuable in coordinating work among developers on the project. Our experience confirms the expectation that the use of module interface documents reduces the effort required for system integration. Misunderstandings about interfaces are exposed during system integration. Since we had documented and reviewed the interfaces before coding started, we discovered fewer misunderstandings during system integration.

9.8 Module design

These documents have been useful for guiding the review of the design. We have not used a standard format for these documents, partially because we did not have a clear idea of what a good format would be. The format for the documents written by the developers has tended to converge during the course of the development so we could probably specify a suitable standard format now.

9.9 Test plan

Developers test their own module, and a small integration and test team tests the overall system. A testing strategy was established early in the development, and more recently, we have devised a detailed test plan. The minimal subset of the system was developed first, and we have used the subset to provide the test environment for the remaining features in the system.

X. APPLICABILITY OF THE DESIGN APPROACH TO LARGER AND SMALLER PRODUCTS

We believe the design method described here can be used effectively on both small and large projects. Resistance can be encountered from people on small projects who are often able to learn most of the requirements and design decisions, and therefore, do not see the need to generate documentation containing the level of detail we have described here. To accept the need for careful documentation one must recognize that most software systems must be maintained for a number of years, and the original developers generally move on to other projects. If the original developers do not adequately document the design, replacement people find the maintenance of the system increasingly difficult as the reasons behind undocumented design decisions are lost.

The need for careful documentation is more readily accepted by people on a large project; however, we have observed cases where people on large projects have overreacted by specifying the generation of redundant documentation that has been a burden to the project.

People on a large project are likely to recognize that a precise specification of requirements is essential to guide development and testing. Module interface documents are particularly important for a large project since the agreements between developers become much more complex as the number of developers is increased.

A large project is often subdivided into several subsystems in order to aid project management. All of the design steps described in the article could be applied to each subsystem. The requirements specification for a subsystem would include functions that are external (visible to the system user) and others that are internal (visible only to the developers of other subsystems). To obtain a complete view of the user visible functions, the text files describing the external functions of each of the subsystems could be combined into a single document. Information hiding should guide the decomposition of a large system into subsystems.

XI. USING THE DESIGN METHOD AS THE BASIS FOR PROJECT MANAGEMENT

The framework provided by the design documents can be used as the basis for project management. The agreements with the user about the functions of the system are embodied in the requirements specification. The basic development unit is the module—a work assignment for one person for a limited period of time. The agreements between developers are recorded in the module interface documents. The order in which the modules are developed is determined from the combination of the required subsets chapter of the requirements specification and the module dependency documents.

Several additional planning and tracking tools (e.g., PERT charts) are needed to aid project management; however, the additional tools should use the work units and agreements specified in the design documents as building blocks. For example, a PERT chart displaying development activities should use modules as the basic development units and the completion of required subsets should be major milestones in the development.

We have used the design method as the basis for managing the development of the No. 2 SES. The module interface documents have been particularly valuable. With the module interface document agreed upon before internal design of the module begins, the developer is much freer to work independently on the development of the module. The developer only needs to negotiate with other members of the development group if a change is required in the module interface. If the supervisor and developer agree on a work plan for developing the module, then the developer is free to execute the work plan without continual involvement of the supervisor or other group members. This autonomy fosters a high level of professionalism and a sense of personal responsibility.

XII. MAINTENANCE OF THE DOCUMENTS

We have used a concise format for most of the design documents. Justification material has been separated into supporting descriptive documents with the exception of the module interface and module design documents. Lists of modules or data items make up much of the other documents.

The concise format of the documents should ease updating and checking for consistency. Automated text processing and static code analysis tools are readily available to reduce the amount of the manual document updating. We have used *UNIX* text processing capabilities to check for consistency and usage of data items.

Several tools are available to extract dependencies from source code. Use of these tools could ensure that the design documents were consistent with the source code. We have not yet adapted these tools for use with our documents; however, we hope to use them in the future. The documents that could be automatically checked for consistency with the code include the module decomposition, module dependency, process structure, and module interface documents. The communications protocols, data items, and user transaction chapters of the requirements specification could also be similarly checked.

The resource allocation document must be updated from system resource usage measurements. Justification documents including the module design documents must be manually updated and reissued periodically.

XIII. CONCLUSION

The principle of separation of concerns requires the division of the design information into clearly distinct and relatively independent documents. These design documents are the main products of the initial design process and, therefore, are the instruments for recording and communicating design decisions. The documents are to be kept up to date throughout the lifetime of the project so one should be able to find current information on any aspect of the software design by examining the relevant document.

The principle of information hiding is used to guide the internal design of the software. The functions of the system that are expected to change are hidden in modules in order to minimize the amount of software affected by a change in these functions. Explicitly designing for change is very desirable for systems like ours that are expected to evolve over a period of years. By giving explicit consideration to the possibility of change, we have identified many potential areas of change. Even so, changes are sure to be proposed that we did not anticipate. We will at least know immediately whether a proposed change is likely to be easy to implement or not.

No design method will prevent one from making bad design decisions; however, the framework provided by the design documents encourages systematically answering a comprehensive set of questions about the system. This process of answering questions may uncover issues often overlooked until late in the development process when they have a costly impact. The design method does not alter what issues must be resolved, but it does change when and how the issues are decided and documented. For example, often requirements issues are not decided explicitly; instead, in the course of the coding, the programmer comes to a point where a decision about external behavior must be made for their work to proceed. They either consult someone or make a private decision. With the design method described here, when a requirements issue is recognized, it is stated in the requirements specification, and the choice of the desired external behavior is made openly. When the choice is made openly, the alternatives will often be more carefully considered. More effort may be invested in making the decision; however, time spent making a careful decision is generally well spent.

The effort required to apply these principles to the development of the No. 2 SES has been accommodated within the development interval originally allocated. The immediate benefits we have gained are (*i*) control of the design process and (*ii*) smooth system integration. In the future, we hope to be able to implement expected changes at low cost.

XIV. ACKNOWLEDGMENTS

Many members of the No. 2 SES Development Group have helped shape these procedures to be more effective for our application. Contributions worthy of special note were made by David T. Johnson in defining the module interface document and by Maureen Ahern in influencing the module design documents.

REFERENCES

1. D. L. Parnas, "On the Criteria To Be Used In Decomposing Systems into Modules," Commun. ACM, *15,* No. 12 (December 1972), pp. 1053–8.
2. D. L. Parnas, "Use of Abstract Interfaces in the Development of Software for Embedded Computer Systems," Naval Research Laboratory, Washington, D. C. 20375, NRL Report 8047, 1977.
3. D. L. Parnas, "Designing Software for Ease of Extension and Contraction," Proc. of the Third Int. Conf. Software Engineering (May 1978), pp. 264–77.
4. K. Heninger et al., "Software Requirements for the A-7E Aircraft," Naval Research Laboratory, Washington, D. C. 20375, NRL Memorandum Report 3876, November 27, 1978.
5. K. Heninger, "Specifying Software Requirements for Complex Systems: New Techniques and Their Application," IEEE Trans. Software Engineering, *SE-6* (1980), pp. 2–13.
6. K. H. Britton, R. A. Parker, and D. L. Parnas, "A Procedure for Designing Abstract Interfaces for Device Interface Modules," Proc. Fifth Int. Conf. Software Engineering, ACM Order No. 592810 (1981), pp. 195–206.
7. E. W. Dijkstra, *A Discipline of Programming,* Englewood Cliffs, NJ: Prentice-Hall, 1976.
8. H. D. Mills, "How to Make Exceptional Performance Dependable and Manageable in Software Engineering," Proc. COMPSAC Conf., IEEE, Catalog No. 80CH1607-1 (October 17–31, 1980), pp. 19–23.
9. N. Wirth, *Systematic Programming,* Englewood Cliffs, NJ: Prentice Hall, 1973.
10. E. Yourdon and L. Constantine, *Structured Design,* Second Edition, New York: Yourdon Press, 1978.
11. M. A. Jackson, *Principles of Program Design,* New York: Academic Press, 1975.
12. E. W. Dijkstra, "Co-operating Sequential Processes," in *Programming Languages,* F. Genuys, Ed., New York: Academic Press, 1968, pp. 43–112.
13. D. L. Parnas and K. Heninger, "Implementing Processes in HAS," in *Software Engineering Principles,* Naval Research Laboratory, Washington, D. C. 20375, Document HAS.9, 1978.
14. J. C. Mathes and D. W. Stevenson, *Designing Technical Reports,* Indianapolis: Bobbs-Merrill, 1976.
15. D. L. Parnas, "On the Design and Development of Program Families," IEEE Trans. Software Engineering, *SE-2,* (March, 1976), pp. 1–9.
16. S. B. Sheppard, E. Kruesi, and B. Curtis, "The Effects of Symbology and Spatial Arrangement on the Comprehension of Software Specifications," Proc. Fifth Int. Conf. Software Engineering, ACM Order No. 592810 (1981), pp. 207–14.

Copyright © 1981 American Telephone and Telegraph Company
THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 60, No. 8, October 1981
Printed in U.S.A.

Minimizing the Worst-Case Distortion in Channel Splitting

By H. S. WITSENHAUSEN

(Manuscript received February 27, 1981)

A sequence of outputs from a stationary memoryless source is encoded into n code streams sent over n parallel channels. Any k or fewer of these channels may have broken down, unbeknown to the encoder. The receiver maps the streams from the surviving channels into a reconstruction sequence for minimum distortion. This distortion will take different values depending on what subset of channels is operative. Let D_{max} be the largest of these values, the worst-case distortion. This paper shows that the infimum of D_{max} over all encodings is the same as if the encoder did have knowledge of the breakdown situation.

I. INTRODUCTION

Consider a stationary, memoryless source emitting at each unit of time a random variable X_t with values in a measurable space $\mathscr{X}$. An encoder maps this source stream into n code streams for transmission over n channels going to a common decoder.

The channels have positive capacities

$$C_1 \leq C_2 \leq \cdots \leq C_n, \tag{1}$$

the inequalities following by the choice of the indexing. Up to k of the channels may in fact have broken down, so that

$$K = \sum_{r=0}^{k} \binom{n}{r} \tag{2}$$

situations are possible, but the encoder does not know which of these K situations is realized. The decoder uses the streams from the operative channels to form a sequence of reconstructions $\hat{X}_t$ in a measurable space $\hat{\mathscr{X}}$, (often, but not necessarily, the same as $\mathscr{X}$). Performance is measured by the time average of a distortion function $d(X_t, \hat{X}_t)$. For any given coding scheme, the expected distortion will depend on the breakdown situation. Here we focus on $D_{\max}$, the largest of the K distortions. $D_{\max}$ is the expected distortion one can guarantee, subject to the assumption that no more than k of the n channels will break down.

If the k channels with the k highest capacities have broken down, a total capacity

$$\rho = \sum_{i=1}^{n-k} C_i \tag{3}$$

is left. Then even if the encoder knew that this was the situation, the distortion could not be made lower than $\delta(\rho)$, where $\delta(\cdot)$ is the classical distortion-rate function for the given source and distortion measure.[1] A fortiori, one has

$$D_{\max} \geq \delta(\rho). \tag{4}$$

In this paper it is shown that this bound is always sharp, i.e., *Theorem: For* $\epsilon > 0$ *it is possible to achieve*

$$D_{\max} < \delta(\rho) + \epsilon \tag{5}$$

by using appropriate coding with large enough block length.

Thus, in the problem of minimizing $D_{\max}$ one can do as well as if the encoder *did* know which of the K breakdown situations was realized. The price paid for this is that, as will be seen, one has effectively to "throw away" most of the excess over ρ of the capacity available in nonworst situations.

II. REGROUPING OF THE CHANNELS

If the capacities C_i, $i > n - k$ are all reduced to the value C_{n-k} then, by (1) and (3), the value of ρ is unchanged, and if the result holds after such reduction it holds, a fortiori, before the reduction. This means that the extra capacity

$$C_{\text{extra}} = \sum_{i=n-k+1}^{n} (C_i - C_{n-k}) \tag{6}$$

can be used for other purposes, such as to reduce the distortion in some situations below $D_{\max}$. Thus, we assume henceforth that $C_i = C_{n-k}$ for $i > n - k$.

The channel coding theorem[1] implies that given $\epsilon > 0$ any channel of capacity C is equivalent—for large enough block length and with appropriate channel coding—to a channel accepting binary bits at rate $C - \epsilon$ and delivering them with arbitrarily small error probability. Thus, we can assume that all n channels are binary with rates $C_i' = C_i$

$-\epsilon_1$ $(i = 1, \cdots, n)$, and that they transmit blocks of sufficient size unaltered, with probability $1 - \epsilon_2$, with ϵ_1, ϵ_2 positive, arbitrarily small.

Lemma: For all $\epsilon > 0$, it is possible to transmit sufficiently long blocks of binary bits at rate $\rho - \epsilon$ with error probability less than ϵ, as long as no more than k channels are out of order.

For the proof, let $\gamma_1 = C_1'$ and for $i = 2, \cdots, n - k$,

$$\gamma_i = C_i' - C_{i-1}'. \tag{7}$$

Then one has

$$C_i' = \sum_{j=1}^{i} \gamma_j \tag{8}$$

for $i = 1, \cdots, n - k$, and

$$C_i' = \sum_{j=1}^{n-k} \gamma_j \tag{9}$$

for $i \geq n - k$.

By (1), the $n - k$ numbers γ_i are nonnegative. As a channel of rate

$$\sum_{j=1}^{i} \gamma_j$$

is equivalent to i parallel channels of respective rates $\gamma_1, \gamma_2, \cdots, \gamma_j$, we may consider the following regrouping of these channels:

Group 1 consists of n channels of equal rate γ_1. The ith of these channels is part of the original channel i.

Group 2 consists of $n - 1$ channels of equal rate γ_2. They correspond to parts of original channels 2 through n.

Continuing in this fashion:

Group i consists of $n - i + 1$ channels of equal rate γ_i. They correspond to parts of original channels i through n.

Finally, group $n - k$ consists of $k + 1$ channels of equal rate γ_{n-k}, corresponding to original channels $n - k$ through n.

Note that for $i = 2, \cdots, n - k$, group i is missing $i - 1$ channels corresponding to the first $i - 1$ original channels. These missing channels can be viewed as permanently broken down channels of an imaginary group of n. As up to k of the original channels may break down, group i, when considered as originally made up of n channels (of rate γ_i), may have up to $k + i - 1$ broken down channels. This is so for all $n - k$ groups, $i = 1, \cdots, n - k$. Now we invoke the known fact[3] that n channels of equal rate γ_i, out of which at most $k + i - 1$ are out of order, can be used to transmit binary bits error-free at rate $(n - k - i + 1)\gamma_i$ using truncated Reed-Solomon (TRS) codes.[2]

Thus the $n - k$ groups yield a total error-free rate

$$\sum_{i=1}^{n-k} (n-k-i+1)\,\gamma_i = \sum_{i=1}^{n-k} \sum_{j=1}^{i} \gamma_j$$
$$= \sum_{i=1}^{n-k} C_i'$$
$$= \rho - (n-k)\epsilon_1.$$

To split a binary block among the $n-k$ groups and assign to each group an integral number of bits—a multiple of its TRS bloc coding length—rounding may be required with asymptotically negligible losses of rate. In addition, the assumed noiseless behavior of the n channels only holds with probability $(1-\epsilon_2)^n$. As all the ϵ's involved go to zero as block length increases, the lemma is proved.

Thus, there exist coding schemes, valid in all K situations, which convey data from transmitter to receiver as if a channel of capacity ρ were between them. Then (5) follows from the classical rate-distortion theory.

III. SPECIAL CASES

For a binary symmetric source with Hamming distortion, that is,

$$d(X, \hat{X}) = 0 \text{ when } X = \hat{X}, 1 \text{ otherwise},$$

one has

$$\delta(\rho) = h^{-1}(1-\rho),$$

where $h^{-1}(x) = 0$ for $x \le 0$, while for $0 < x \le 1$ it is the inverse of the restriction to $(0, \frac{1}{2})$ of

$$h(x) = -x \log_2 x - (1-x) \log_2 (1-x).^1$$

For n channels of equal capacity C, of which k can break down, one has $\rho = (n-k)C$ so that the limit of achievability is given by

$$D_{\max} = h^{-1}(1-(n-k)C). \qquad (10)$$

If in particular $C = n^{-1}$ (channels of total capacity 1), then

$$D_{\max} = h^{-1}(k/n). \qquad (11)$$

For $k = 1$, if one insists that the distortion approach zero when all n channels are up, one can achieve[4] distortion $(2^{1/n} - 1)/2$ when any channel is down, which is of order n^{-1}. If, however, one only cares about maximum distortion, then one can approach $h^{-1}(1/n)$ which is of order $(n \log n)^{-1}$.

If, for example, $n = 3$ and $k = 1$, then $(2^{1/3} - 1)/2 \simeq 0.130$ while $h^{-1}(\frac{1}{3}) \simeq 0.062$.

REFERENCES

1. R. G. Gallager, *Information Theory and Reliable Communication*, New York: Wiley, 1968.
2. F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*, Amsterdam: North Holland, 1977.
3. H. S. Witsenhausen, "On Source Networks with Minimal Breakdown Degradation," B.S.T.J., *59,* No. 6 (July-August 1980), pp. 1083-7.
4. H. S. Witsenhausen, unpublished work.

CONTRIBUTORS TO THIS ISSUE

Gerald R. Ash, B.S. (Electrical Engineering), 1964, Rutgers University; M.S. (Electrical Engineering), 1965, Ph.D., 1969, California Institute of Technology; Staff Member, IBM, E. Fishkill, NY, 1968-1969; Captain, U.S. Army, 1969-1971; Staff Member, Institute for Defense Analyses, 1971-1972; Bell Laboratories, 1972—. At Bell Laboratories, Mr. Ash has worked on short duration call analyses and error control in Operation Support System data bases. He is presently a member of the Traffic Network Planning Department and is working on dynamic nonhierarchical routing. Member, Eta Kappa Nu, Tau Beta Pi. Associate Member, Sigma Xi.

Richard H. Cardwell, B.S. (Electrical Engineering), 1967, Michigan Technological University; Bell Laboratories, 1968—. Mr. Cardwell has been engaged in the development of new telephone call routing techniques and optimum network design. He is now a member of the Digital Data Transmission and Services Department and is developing new digital data system equipment. Member, Tau Beta Pi, Eta Kappa Nu.

Anthony Carnevale, B.S. (Physics), 1960, Fairleigh Dickinson University; Bell Laboratories, 1969—. At Bell Laboratories, Mr. Carnevale has been engaged in work on nuclear magnetic resonance, electron paramagnetic resonance, and computer software. For the last two years, his work has been devoted to fiber optics.

Fan R. K. Chung, B.S. (Mathematics), 1970, National Taiwan University; Ph.D., 1974, University of Pennsylvania; Bell Laboratories, 1974—. Mrs. Chung's current interests include combinatorics, graph theory, and the analysis of algorithms. She is presently investigating various problems in the theory of swtiching networks.

Leonard G. Cohen, B.E.E., 1962, City College of New York; Sc.M., 1964, and Ph.D. (Engineering), 1968, Brown University; Bell Laboratories, 1968—. At Brown University, Mr. Cohen was engaged in research on plasma dynamics. At Bell Laboratories, he has concentrated on optical fiber transmission studies. He received the 1980 IEEE (London, England) Electronics Letters Premium Award for a paper relating to optical fibers. Member, Sigma Xi, Tau Beta Pi, Eta Kappa Nu, Optical Society of America; senior member, IEEE.

Bernard S. Glance, Electronic Engineering, 1958, Ecole Spéciale de Méchanique et Electricité, Paris, France; Electronic Engineering, 1960, Ecole Supérieure d'Electricité, Paris, France; Ph.D. (Electronic Engineering), 1964, Paris University; Thompson, CSF, France, 1958–1966; Varian Associates, 1966-1968; Bell Laboratories, 1968—. At Thompson, CSF, Mr. Glance was engaged in research on microwave and millimeter-wave tubes, and at Varian Associates he worked on microwave tube amplifiers. After joining Bell Laboratories, his main field of interest was in millimeter-wave integrated circuits and RF sources. He subsequently worked on microwave integrated components for satellite radio systems and is presently involved with digital mobile radio systems. Member, IEEE.

R. L. Graham, B.S. (Physics), 1958, University of Alaska, M.A., Ph.D. (Mathematics), 1962, University of California (Berkeley), Bell Laboratories, 1962—. Mr. Graham has been concerned with a variety of areas of mathematics and computer science, including combinatorics, number theory, and analysis of algorithms. He is currently Head of the Discrete Mathematics Department. Mr. Graham is an officer or member of numerous professional societies and currently serves on more than 20 editorial boards of technical journals and publications.

Henry S. Greenside, B.A. (Mathematics), 1974, Harvard University; M.A., 1977, and Ph.D. (Physics), 1981, Princeton University; Bell Laboratories, 1981—. Mr. Greenside has worked on several problems in theoretical condensed matter physics including surface electronic structure and the coexistence of superconductivity and magnetism. He is currently interested in the onset of turbulence in condensed matter systems. Member, American Physical Society.

Eugene Helfand, B.S. (Chemistry), 1955, Polytechnic Institute of Brooklyn; M.S., 1957, Ph.D. (Chemistry), 1958, Yale University; Bell Laboratories, 1958—. Mr. Helfand has worked on various theoretical aspects of chemistry and materials science including transport theory, liquids, superconductivity, and critical phenomena. His current interests are in polymer science and dynamical simulation. Guggenheim Fellow, 1969. Member, American Physical Society, American Chemical Society, Sigma Xi, Phi Lambda Upsilon.

Paul S. Henry, A.B. (Physics), 1965, Harvard University; Ph.D. (Physics), 1971, Princeton University; Bell Laboratories, 1970—. Mr. Henry has been engaged in research on radio and optical communication and radio astronomy instrumentation. He has published papers or patented inventions in several fields, including millimeter-wave techniques, cosmology, mobile radio systems, and cryptography. He served as Associate Editor of IEEE Communications Magazine and received the 1979 IEEE Vehicular Technology Society Paper of the Year Award for Communications Systems. Mr. Henry is currently Head of the Optical Systems Research Department at Bell Laboratories. Member, IEEE.

S. Dean Hester, B.S. (Engineering), 1965, University of Missouri; Ph.D., 1969, Massachusetts Institute of Technology; Avco Everett Research Laboratory, 1969–1971; Bell Laboratories, 1971—. Mr. Hester initially worked on development of the T1C and T4 digital transmission systems. Between 1974 and 1978 he contributed to the development of three operations systems handling traffic data collection and network management (EADAS, EADAS/NM, and NOCS). In 1978, he supervised the group that developed the No. 1 Service Evaluation System feature that detects network faults causing loss of toll revenue. Since 1979, he has supervised the development of the No. 2 Service Evaluation System.

Frank K. Hwang, B.A. (Modern Languages), 1960, National Taiwan University; M.B.A., 1964, City University of New York; M.E.S., 1966, Ph.D., 1968, North Carolina State University; Bell Laboratories, 1967—. Mr. Hwang has been engaged in research in discrete mathematics, computing algorithms, statistical theory, and switching networks. Member, SIAM.

Alan H. Kafker, B.S. (Mathematics), 1974, S.U.N.Y. at Stony Brook; M.A., 1976, Ph.D. 1979 (Mathematics) University of Pennsylvania; Bell Laboratories 1979—. Since joining Bell Laboratories, Mr. Kafker has worked on real-time routing, and the impact of dynamic nonhierarchical routing (DNHR) on network management and the stored program controlled network. He is currently investigating the functional requirements for toll switches to implement DNHR in the intercity network. Member, American Mathematics Society, Operations Research Society of America.

Krishnan R. Krishnan, B.E., 1962, Nagpur University, India; M.S. (Electrical Engineering), 1963, University of Illinois, Urbana-Champaign; Ph.D. (Electrical Engineering), 1969, University of Minnesota, Minneapolis; Lecturer, Indian Institute of Science, Bangalore, 1963–

1966; Instructor, University of Minnesota, Minneapolis, 1969–1970; Research Associate, University of Colorado, Boulder 1970–1972; Assistant Professor, Clarkson College, Potsdam, New York, 1972–1977; Bell Laboratories 1977—. At Bell Laboratories, Mr. Krishnan has worked on real-time traffic routing and servicing methods for networks using dynamic, nonhierarchical routing. Member, IEEE, Sigma Xi.

Wanda L. Mammel, A.B. (Mathematics), 1943, Winthrop College; M.Sc. (Applied Mathematics), 1945, Brown University; Bell Laboratories, 1956—. Ms. Mammel is engaged in finding mathematical methods for the numerical solution of a variety of problems. In particular, she has applied linear programming techniques to problems of crystal plasticity. At present she is working on problems in microwave propagation and optical waveguides.

Robert J. T. Morris, B.Sc., B.E. (Computer Science/Electrical Engineering), 1974, University of New South Wales, Australia; M.S. (System Science), 1976, Ph.D. (Computer Science), 1978, University of California, Los Angeles; Bell Laboratories, 1978—. Since joining Bell Laboratories, Mr. Morris has worked on the modeling and analysis of a variety of computer-based systems under development for packet and circuit switching applications. He is currently supervisor of the Performance Analysis Studies Group in the Network Performance Planning Center.

Markus S. Mueller, E.E. Diploma, 1970, Ph.D. (Electronic Engineering), 1976, Swiss Federal Institute of Technology, Zurich, Switzerland; Swiss Federal Institute of Technology, Zurich, 1971–1976; GenRad, Zurich, 1976–1978; Bell Laboratories, 1978–1981. Mr. Mueller was teaching and research assistant at the Swiss Federal Institute of Technology, where he worked in various fields, including filter theory, data transmission, and adaptive signal processing. He was a product specialist with GenRad for computer controlled automatic test systems. At Bell Laboratories, he was a member of the Data Theory Group in the Data Systems and Technology Department, and his interests were in data communication, digital signal processing, and adaptive systems.

Robert P. Murray, B.S. (Mathematics), 1978, Manhattan College; M.S. (Operations Research), 1980, Columbia University; Bell Laboratories, 1978—. As a member of the Traffic Network Planning Department, Mr. Murray has participated in the study of engineering methods for the public telephone network. Member, Phi Beta Kappa, Pi Mu Epsilon.

Un-Chul Paek, B.S. (Engineering), 1957, Korea Merchant Marine Academy, Korea; M.S., 1965, Ph.D., 1969, University of California, Berkeley; Western Electric, 1969—. At the Western Electric Engineering Research Center, Princeton, N. J., Mr. Paek has been engaged primarily in research on laser material interaction phenomena and fiber optics. Member, Optical Society of America, American Ceramic Society, Sigma Xi.

David L. Parnas, B.S., 1961, M.S., 1964, Ph.D. (Electrical Engineering), 1965, Carnegie-Mellon University; Philips-Electrologica, Apeldoorn, Netherlands, 1969–1970; Technical University of Darmstadt, Federal Republic of Germany, 1973–1976; University of North Carolina at Chapel Hill, 1976—. Mr. Parnas has held academic positions at Carnegie-Mellon and at the University of Maryland. At Philips-Electrologica he served as an advisor, and at the Technical University of Darmstadt he was Professor and Head of a Research Group on Operating Systems. Mr. Parnas has also been a consultant to Bell Laboratories, TRW, IBM, and the U.S. Navy. Currently on leave from his position as Professor of Computer Science at the University of North Carolina, he is a visiting scientist at IBM's Federal Systems Division in Bethesda, Maryland. Member, Computer Science and Systems Branch, Naval Research Laboratory, Washington, D.C.

A. David Pearson, B.Sc. Hons. (Chemistry), 1953, King's College, University of Durham, England; Ph.D. (Inorganic Chemistry), 1957, Massachusetts Institute of Technology; Bell Laboratories, 1957—. Mr. Pearson's field of specialization is the chemistry and physics of glass. At Bell Laboratories, he has been concerned with the preparation and properties of new glass compositions, including insulating glasses, semiconducting glasses, laser glasses, and glasses for use in optical communication systems. His current work involves the design and fabrication of high-performance, single-mode optical fiber lightguides. Fellow, American Ceramic Society (past-chairman Glass Division) and recipient of the Forrest Award 1961; Member, Optical Society of America.

George E. Peterson, B.S. (Physics), 1956, Ph.D. (Solid State Physics), 1961, University of Pittsburgh; Bell Laboratories, 1961—. At Bell Laboratories, Mr. Peterson initially was engaged in studies on low-noise amplifiers. He then studied laser crystals, nonlinear optic materials, and glass structure. Presently, he is studying propagation of light in optical fibers. Member, American Physical Society, American Ceramic Society, Society for Glass Technology, and the America Crystallographic Association.

Lawrence R. Rabiner, S.B. and S.M., 1964, Ph.D. (Electrical Engineering), The Massachusetts Institute of Technology; Bell Laboratories, 1962—. From 1962 through 1964, Mr. Rabiner participated in the cooperative plan in electrical engineering at Bell Laboratories. He worked on digital circuitry, military communications problems, and problems in binaural hearing. Presently, he is engaged in research on speech communications and digital signal processing techniques. He is coauthor of *Theory and Application of Digital Signal Processing* (Prentice-Hall, 1975) and *Digital Processing of Speech Signals* (Prentice-Hall, 1978). Former President, IEEE, ASSP Society; former Associate Editor, ASSP Transactions; former member, Technical Committee on Speech Communication of the Acoustical Society, ASSP Technical Committee on Speech Communication; Member, IEEE Proceedings Editorial Board, Eta Kappa Nu, Sigma Xi, Tau Beta Pi. Fellow, Acoustical Society of America, IEEE.

Carolyn E. Schmidt, B.S. (Mathematics), 1974, Lafayette College; Bell Laboratories, 1974—. Ms. Schmidt is a member of the Acoustic Research Department and is currently involved in work on speech communications. Member, Phi Beta Kappa.

Man Mohan Sondhi, B.Sc. (Physics), Honours degree, 1950, Delhi University, Delhi, India; D.I.I.Sc. (Communications Engineering), 1953, Indian Institute of Science, Bangalore, India; M.S., 1955; Ph.D. (Electrical Engineering), 1957, University of Wisconsin, Madison, Wisconsin; Bell Laboratories, 1962—. Before joining Bell Laboratories, Mr. Sondhi worked for a year at the Central Electronics Engineering Research Institute, Pilani, India and taught for a year at the University of Toronto. At Bell Laboratories his research has included work on speech signal processing, echo cancellation, adaptive filtering, modelling of auditory and visual processes and acoustical inverse problems. From 1971 to 1972 Mr. Sondhi was a guest scientist at the Royal Institute of Technology, Stockholm, Sweden.

Julian Stone, B.S. (Physics), 1950, The City College, New York; M.S. (Physics), 1951, and Ph.D. (Physics), 1958, New York University;

Bell Laboratories, 1969—. Mr. Stone taught at The City College from 1952 to 1953 and 1956 to 1958. He was at The Hudson Laboratories of Columbia University from 1953 to 1969 where he was Associate Director for General Physics and was active in underwater acoustics and optics. At Bell Laboratories, Mr. Stone has been working on problems in optical transmission. Member, American Physical Society.

Michael Tortorella, A.B., 1968, Fordham University; M.S., 1970, and Ph.D. (Mathematics), 1973, Purdue University; Bell Laboratories, 1975—. From 1973 to 1975, Mr. Tortorella was a member of the mathematics faculty at the University of Wisconsin, Milwaukee. At Bell Laboratories, he has studied applications of stochastic processes in network planning, call setup performance objectives, and switching and transmission system reliability. He is now organizing the reliability evaluation of the proposed submarine lightwave cable system. Member, The Institute of Mathematical Statistics, SIAM.

Donald F. Utter, Jr., 1965, B.S. (Mathematics), Long Beach State; 1967, M.A. (Pure Mathematics), University of California, Berkeley; 1973, Ph.D. (Numerical Analysis), Arizona State University; Bucknell University, 1973-1978; Bell Laboratories, 1978—. Before joining Bell Laboratories, Mr. Utter taught Computer Science at Bucknell University and was department chairman during his last two years there. At Bell Laboratories, he is a member of the Operating Systems Group at Columbus and serves as a consultant on software methodology and requirements, as well as on preparation of materials to transfer methodology. His primary interest is in the design process. He worked for almost three years on the No. 2 Service Evaluation System doing system engineering and requirements work.

Hans S. Witsenhausen, Ph.D. (Electrical Engineering), 1966, Massachusetts Institute of Techology; Bell Laboratories, 1966—. Mr. Witsenhausen has been associated with the Université Libre de Bruxelles, with the European Computation Center, Brussels, and with the Princeton Research Division of Electronic Associates. At M.I.T. he was with the Electronic Systems Laboratory as a Lincoln Laboratory Associate and a Hertz Fellow. At Bell Laboratories, he is associated with the Mathematics and Statistics Research Center. He was a Senior Fellow at the Imperial College of Science and Technology, London, in 1972, a Visiting Professor at M.I.T. in 1973, and Vinton Hayes Senior Fellow at Harvard University in 1975-76. He has worked on problems of hybrid computation, control theory, optimization, geometric inequalities, and other applied mathematical fields.

CONTENTS, NOVEMBER 1981

THE BELL SYSTEM TECHNICAL JOURNAL is abstracted or indexed by *Abstract Journal in Earthquake Engineering*, *Applied Mechanics Review*, *Applied Science & Technology Index*, *Chemical Abstracts*, *Computer Abstracts*, *Current Contents/ Engineering, Technology & Applied Sciences*, *Current Index to Statistics*, *Current Papers in Electrical & Electronic Engineering*, *Current Papers on Computers & Control*, *Electronics & Communications Abstracts Journal*, *The Engineering Index*, *International Aerospace Abstracts*, *Journal of Current Laser Abstracts*, *Language and Language Behavior Abstracts*, *Mathematical Reviews*, *Science Abstracts* (*Series A*, *Physics Abstracts*; *Series B*, *Electrical and Electronic Abstracts*; *and Series C*, *Computer & Control Abstracts*), *Science Citation Index*, *Sociological Abstracts*, *Social Welfare*, *Social Planning and Social Development*, and *Solid State Abstracts Journal*. Reproductions of the Journal by years are available in microform from University Microfilms, 300 N. Zeeb Road, Ann Arbor, Michigan 48106.

CONTENTS (continued)

Bell System