$$\omega/\omega_o \leq 1 + (k/n\omega_p)(n).$$

However, dividing both sides of (5) by ω_o gives us the slightly better bound

$$\omega/\omega_o \leq (\omega_p/\omega_o)[1 + (k/n\omega_p)(n-1)].$$

VII. CONCLUSIONS

In this paper we have considered the "longest path" scheduling algorithm as an "almost" optimal algorithm for the scheduling of trees. An upper bound on the execution time for this algorithm was presented and shown to be better than previous upper bounds for this and related problems.

ACKNOWLEDGMENT

The author would like to thank the referees for their helpful criticisms and suggestions during the preparation of this paper.

REFERENCES

[1] J. Bruno, E. G. Coffman, Jr., and R. Sethi, "Scheduling independent tasks to reduce mean finishing-time," *Oper. Syst. Rev.* (Extended Abstract), vol. 7, pp. 102–103, Oct. 1973.

[2] P. J. Denning and G. S. Graham, "A note on sub-expression ordering in the execution of arithmetic expressions," *Commun. Ass. Comput. Mach.*, vol. 16, pp. 700–702, Nov. 1973.

[3] R. L. Graham, "Bounds for certain multiprocessing anomalies," *Bell Syst. Tech. J.*, vol. 45, pp. 1563–1581, Sept. 1966.

[4] ——, "Bounds on multiprocessing timing anomalies," *SIAM J. Appl. Math.*, vol. 17, pp. 416–429, Mar. 1969.

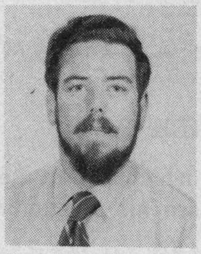
[5] ——, "Bounds on multiprocessing anomalies and packing algorithms," in *1972 Spring Joint Comput. Conf., AFIPS Conf. Proc.*, vol. 40. Montvale, N. J.: AFIPS Press, pp. 205–217.

[6] T. C. Hu, "Parallel sequencing and assembly line problems," *Oper. Res.*, vol. 9, pp. 841–848, Nov. 1961.

[7] G. K. Manacher, "Production and stabilization of real-time task schedules," *J. Ass. Comput. Mach.*, vol. 14, pp. 439–465, July 1967.

[8] R. R. Muntz and E. G. Coffman, Jr., "Preemptive scheduling of real-time tasks on multiprocessor systems," *J. Ass. Comput. Mach.*, vol. 17, pp. 324–328, Apr. 1970.

[9] C. V. Ramamoorthy and M. J. Gonzales, "Subexpression ordering in the execution of arithmetic expressions," *Commun. Ass. Comput. Mach.*, vol. 14, pp. 479–485, July 1971.



Marc T. Kaufman (S'62–M'64) was born in Oakland, Calif. on February 3, 1943. He received the B.S. degree from the California State Polytechnic College, Pomona, in 1964, the M.S. degree from the University of Virginia, Charlottesville, in 1967, and the Ph.D. degree from Stanford University, Stanford, Calif., in 1973.

He has been professionally involved in the computer field since 1962, starting as an Applications Programmer for a university computing center. Between 1968 and 1972 he was a Staff Specialist working on compiler development for Control Data Corp. in Sunnyvale, Calif. During the 1972–73 academic year he was an Acting Assistant Professor of Electrical Engineering at Stanford University, affiliated with the Digital Systems Laboratory. Since 1973, he has been an independent consultant, specializing in networks and interactive systems. He is also a Lecturer in Computer Science at Stanford University.

Dr. Kaufman is a member of Sigma Xi.

A Means for Achieving a High Degree of Compaction on Scan-Digitized Printed Text

R. N. ASCHER AND GEORGE NAGY, SENIOR MEMBER, IEEE

Abstract—**A method of video compaction based on transmitting only the first instance of each class of digitized patterns is shown to yield a compaction ratio of 16:1 on a short passage of text from the IEEE SPECTRUM. Refinements to extend the bandwidth reduction to 40:1 by relatively simple means are proposed but not demonstrated.**

Index Terms—**Bandwidth reduction, character recognition, compaction, data compression, OCR, optical character recognition, printed text.**

Manuscript received July 30, 1973; revised April 1, 1974.

R. N. Ascher is with the IBM Thomas J. Watson Research Center, Yorktown Heights, N. Y. 10598.

G. Nagy was with the IBM Thomas J. Watson Research Center, Yorktown Heights, N. Y. He is now with the Department of Computer Science, University of Nebraska, Lincoln, Nebr. 68508.

INTRODUCTION

A VERY HIGH degree of bandwidth reduction on printed text digitized by an optical scanner may be achieved by transmitting or storing only the first instance of each pattern class and thereafter substituting this exemplar for every subsequent occurrence of the symbol. This process, which approaches in efficiency actual *recognition* of the text but does not require the assignment of correct alphanumeric labels to the characters, is illustrated in Fig. 1.

The determination of which characters are to be saved in video form is based on binary correlation. A character is saved (in our terminology, becomes a *prototype*) only

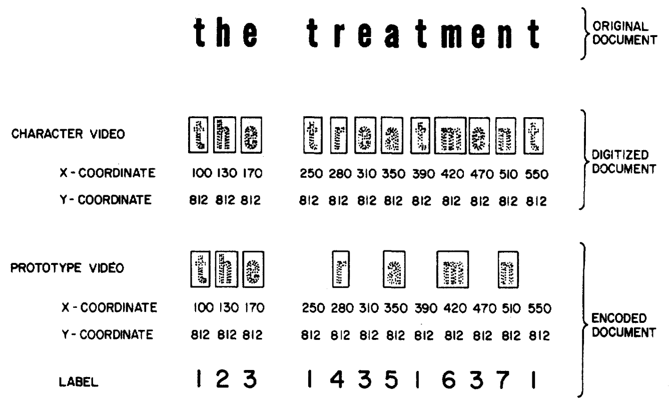

the treatment
ORIGINAL DOCUMENT
CHARACTER VIDEO
X-COORDINATE 100 130 170 250 280 310 350 390 420 470 510 550
Y-COORDINATE 812 812 812 812 812 812 812 812 812 812 812 812
DIGITIZED DOCUMENT
PROTOTYPE VIDEO
X-COORDINATE 100 130 170 250 280 310 350 390 420 470 510 550
Y-COORDINATE 812 812 812 812 812 812 812 812 812 812 812 812
LABEL 1 2 3 1 4 3 5 1 6 3 7 1
ENCODED DOCUMENT

Fig. 1. Schematic diagram of video compaction method.

if it is not highly correlated with any of the previously saved characters. If a character is not saved, then it is assigned to the prototype most highly correlated with it, and only the identification *number* of this prototype is transmitted or stored.

The method is derived from a nonsupervised method of character recognition in which the prototypes are saved for subsequent identification by a human operator [1]. In the procedure described here, however, the prototypes are either saved or transmitted without ever being assigned alphanumeric identities.[1]

The routines necessary to scan, isolate, and correlate the characters are outlined in [1], and are used without change in the present experiment. The correlative coding process does require a considerable amount of computation, but the decoding process—looking up the video pattern corresponding to a prototype number—is straightforward and rapid. Hence, this compaction method is particularly suitable for archival applications, in which the high cost of encoding is justified by the high degree of compaction achieved and the ease and faithfulness of reconstitution.

CHARACTER ACQUISITION

The character acquisition routine which controls the flying-spot scanner is designed to locate characters (of any size) within any specified area of a page. It scans downward to find lines of print, computes the raster size based on the height of each line, adjusts the black-white threshold regularly throughout the area, and initiates the character scan which isolates the characters, converting each into a binary matrix 32 bits high by a variable width, and registering it within the matrix in a standard position for subsequent recognition.

Even if the characters scanned are of almost uniform darkness, the light output of the scanner varies with time as well as with position on the document. Therefore, the threshold, which the machine uses to separate black from white points when producing binary video, must periodically be recomputed throughout the field being scanned.

Starting in the upper left corner of a field, before searching for a line, a single horizontal scan of 32 points is read in the "grey-scale" mode (which yields a set of values falling between 0 and, say, 3200, corresponding to the amount of light collected from each point). It is expected that at least one point of the scan will fall on a blank area, allowing determination of the brightness value corresponding to the background. The threshold to be used by the computer in discriminating black from white in the "binary" mode is computed by multiplying this value by a specified constant. This procedure is followed before searching for each new line and also at regular intervals during the scanning of each line.

Each line to be scanned is isolated in the following manner. Horizontal scans are generated, in the "binary" mode, extending inward from the left margin about an inch, and moving downward from the top of the field in specified increments. For each scan, the number of bits that have changed since the preceding scan, from 1 to 0 or from 0 to 1, is determined. When this number becomes larger than a minimum cutoff threshold, it indicates that the scan has picked up the tops of some of the taller characters on a line. When it falls below the cutoff, the scan has passed below the descenders of the lowest characters. The transition counts in question are easily obtained from the EXCLUSIVE OR function of successive pairs of scans.

The resolution for the character scans is chosen so that the distance between the two peaks in the transition count will account for a specified fraction (usually a little less than half) of the raster height. Upper and lower limits are also imposed to avoid either scanning two lines simultaneously or mistaking a mark separating paragraphs for a line of print. The raster is centered about the two peaks.

The actual acquisition of characters is performed by a binary vertical scan of 32 bits moving across the line of print from the left margin of the field to the right. The horizontal spacing between scans is the same as, or an integral multiple of, the vertical spacing which was computed when the line was found.

The finer scan is used for sacnning material containing closely spaced characters, which are difficult to separate. If a finer scan is used, i.e., if the horizontal resolution is K times the vertical resolution, then only every Kth scan is retained in the pattern for classification, so that the character will be in proportion.

Segmentation has been long acknowledged as among the most difficult problems of character recognition, particularly on variable pitch material. Elaborate algorithms exist for both gothic and roman type styles, but in our case the limiting factor was the spot size obtainable with our scanners. The simple segmentation procedure described below may have to be extensively modified if efforts to reduce the spot size do not prove successful.

[1] The idea of applying the nonsupervised character-classification programs to compaction was suggested to us by Dr. P. Welch. The authors are also grateful to Dr. L. Bahl, Dr. D. Barnea, and Dr. H. Kobayashi for informative discussions of the various aspects of video encoding.

As each scan is read and tested for the presence of black points, a set of rules is applied to determine whether a complete character has been scanned. These rules are specified by five program parameters: $N1$, $N2$, $N3$, $N4$, and $N5$. The first three specify different regions of the character.

In the first region (after at least one nonwhite scan, but before $N1$ scans have been stored), each scan is compared with the one before to see if they have any points in common (in the same bit positions). If they have fewer than $N4$ points in common, the character is ended and the accumulated scans are thrown away.

In the second region (after $N1$ but before $N2$ scans), the character is not terminated unless there are $N5$ consecutive white scans.

In the third region (after $N2$ but before $N3$ scans), segmentation occurs whenever a scan has fewer than $N4$ black points in common with the one before it (the same as in region 1). After $N3$ scans have been acquired, forced separation occurs. Of course, a region may be omitted entirely if desired. For example, the rule for region 2 may be bypassed if the same value is specified for $N2$ and for $N1$.

The net effect of these rules is to eliminate small fragments and to facilitate segmentation toward the end of a long connected region.

The correlation procedure used for classification requires that all characters be approximately "registered" so that their lower edges are the same distance from the bottom of the matrix. In material which has very closely spaced lines, the tops of characters in the line below may occasionally be picked up, making the character impossible to recognize due to its irregular position.

This condition is corrected as follows. If, in the union of all scans, there is a smaller group of black points below the major part of the character, it is considered to be noise and is removed. Then the character is shifted to place the lowest black point in a standard position of the matrix.

The final output of this set of programs is a tape or disk on which each logical record corresponds to a single character. A ten-word header specifies the field number, the horizontal and vertical coordinates of the character on the page, the sequence number, the number of words and black bits in the video, whether the character was preceded by a blank space, and also provides two blank words for subsequent label tags. The header is followed by the video representation of the character in binary form.

CORRELATION AGAINST PROTOTYPES

In the coding operation the bit pattern of 1's and 0's representing an incoming symbol is compared with the bit patterns of prototype symbols.

Each prototype is correlated with the incoming symbol in every shift position of a 5×7 matrix. For each prototype–symbol pair, the maximum of these sums is normalized (to take into account mismatching of corresponding bits in symbol and prototype) in the following manner:

$$\text{score} = 100 \times \frac{(\text{max number of matching 1's in prototype and character})^2}{(\text{number of 1's in character})\,(\text{number of 1's in prototype})}.$$

The normalized maximum (or "score") resulting from the correlation of a symbol with a prototype thus constitutes a measure of the resemblance between symbol and prototype.

EXAMPLE

To demonstrate the method, about half a page from the IEEE SPECTRUM [Fig. 2(a)] was scanned by means of the flying-spot scanner at a nominal resolution of 200 lines/in and processed in the manner described. The quantized video image without compaction and the corresponding segment of the reconstituted text are shown in Fig. 2(b) and (c).

The minimal loss due to the compaction process may be observed by comparing Fig. 2(b) and (c), bearing in mind that the degradation in image quality evident in both Fig. 2(b) and (c), as compared with Fig. 2(a), is due not only to the relatively large scanning aperture of the flying-spot scanner but also to the low resolution of the equipment used to produce the hardcopy output.

On the average, each prototype was represented by a 10×15 binary array [Fig. 3]. As found in [1], beyond the initial rash of new prototypes, prototypes are generated at the rate of about 60 per 1000 characters, corresponding mainly to mis-segmented pairs of characters. In the present experiment, each nonprototype character was coded into 9 bits representing the prototype number and 20 bits of x- and y-coordinate information representing the position of the character on the page. Since 1000 characters occupy about $\frac{1}{6}$ of an $8\text{-}\frac{1}{2}\text{-} \times 11$-page, the compaction ratio achieved may be estimated as

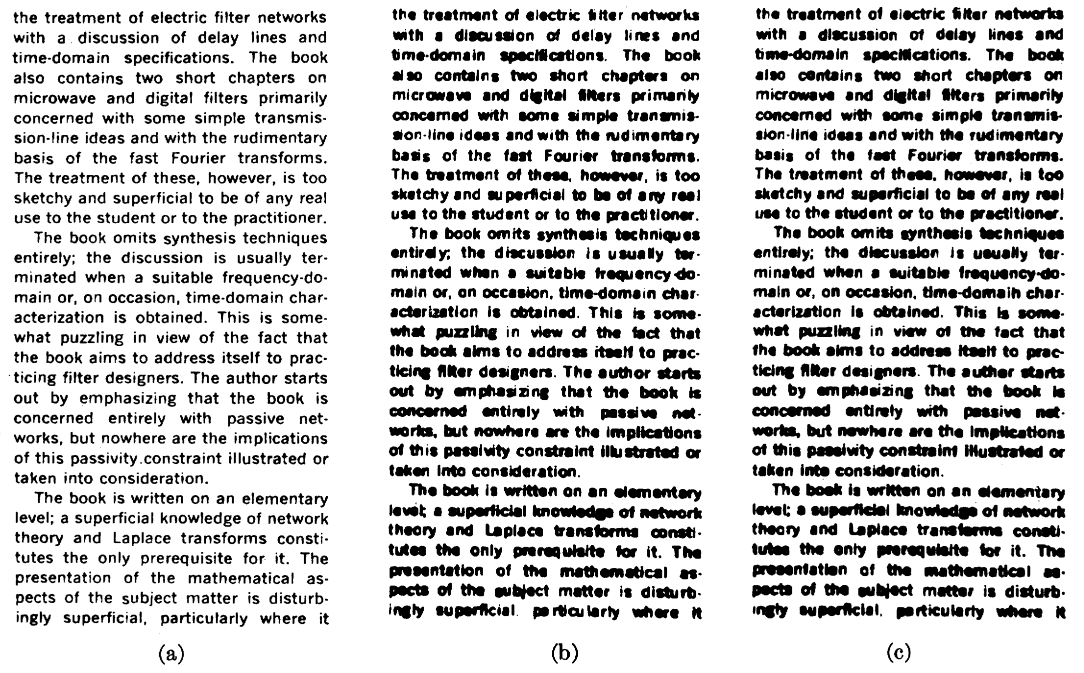
$$\frac{200 \times 200 \times 8.5 \times 11 \times 1/6}{60 \times 15 \times 10 + 1000 \times (9 + 20)} \simeq 16{:}1.$$

PROJECTED COMPACTION RATIO

It should be noted that this compression ratio may be improved significantly by combining our technique with existing methods. The following steps are described roughly in the order of ease of implementation.

1) The coordinate information could be reduced to about 8 bits per character by considering only the displacement from the previous character.

2) The prototype numbers could be Huffman coded according to their frequency distribution [2]. Using Fig. 4 (embodying data from [1]), we estimate that 4 bits per character would be sufficient; larger gains could be realized by considering higher order n-gram distributions.



the treatment of electric filter networks with a discussion of delay lines and time-domain specifications. The book also contains two short chapters on microwave and digital filters primarily concerned with some simple transmission-line ideas and with the rudimentary basis of the fast Fourier transforms. The treatment of these, however, is too sketchy and superficial to be of any real use to the student or to the practitioner.

The book omits synthesis techniques entirely; the discussion is usually terminated when a suitable frequency-domain or, on occasion, time-domain characterization is obtained. This is somewhat puzzling in view of the fact that the book aims to address itself to practicing filter designers. The author starts out by emphasizing that the book is concerned entirely with passive networks, but nowhere are the implications of this passivity.constraint illustrated or taken into consideration.

The book is written on an elementary level; a superficial knowledge of network theory and Laplace transforms constitutes the only prerequisite for it. The presentation of the mathematical aspects of the subject matter is disturbingly superficial, particularly where it

the treatment of electric filter networks with a discussion of delay lines and time-domain specifications. The book also contains two short chapters on microwave and digital filters primarily concerned with some simple transmission-line ideas and with the rudimentary basis of the fast Fourier transforms. The treatment of these, however, is too sketchy and superficial to be of any real use to the student or to the practitioner.

The book omits synthesis techniques entirely; the discussion is usually terminated when a suitable frequency-domain or, on occasion, time-domain characterization is obtained. This is somewhat puzzling in view of the fact that the book aims to address itself to practicing filter designers. The author starts out by emphasizing that the book is concerned entirely with passive networks, but nowhere are the implications of this passivity constraint illustrated or taken into consideration.

The book is written on an elementary level; a superficial knowledge of network theory and Laplace transforms constitutes the only prerequisite for it. The presentation of the mathematical aspects of the subject matter is disturbingly superficial, particularly where it

the treatment of electric filter networks with a discussion of delay lines and time-domain specifications. The book also contains two short chapters on microwave and digital filters primarily concerned with some simple transmission-line ideas and with the rudimentary basis of the fast Fourier transforms. The treatment of these, however, is too sketchy and superficial to be of any real use to the student or to the practitioner.

The book omits synthesis techniques entirely; the discussion is usually terminated when a suitable frequency-domain or, on occasion, time-domaih characterization is obtained. This is somewhat puzzling in view of the fact that the book aims to address itself to practicing filter designers. The author starts out by emphasizing that the book is concerned entirely with passive networks, but nowhere are the implications of this passivity constraint illustrated or taken into consideration.

The book is written on an elementary level; a superficial knowledge of network theory and Laplace transforms constitutes the only prerequisite for it. The presentation of the mathematical aspects of the subject matter is disturbingly superficial, particularly where it

(a) (b) (c)

Fig. 2. (a) Original document. (b) After digitization using a flying-spot scanner and plotting using a cathode-ray-tube plotter. (c) As reconstituted from compressed information—same plotter. There is no simple way to differentiate the degradation introduced in scanning from that due to the plotter. The minimal degradation due to compaction, however, can be seen by comparing Fig. 2(b) and (c).

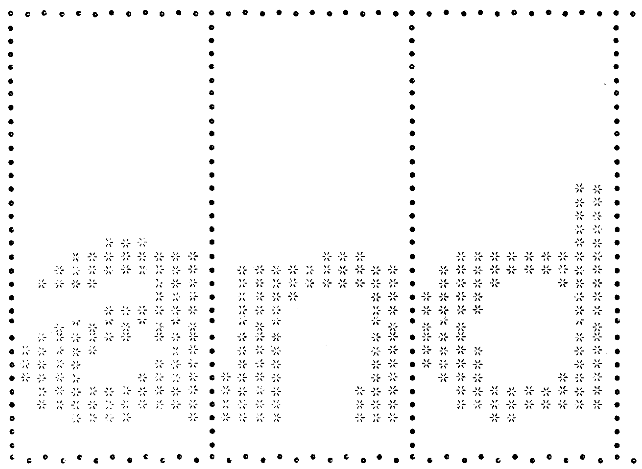


Fig. 3. Digitized video characters after segmentation.

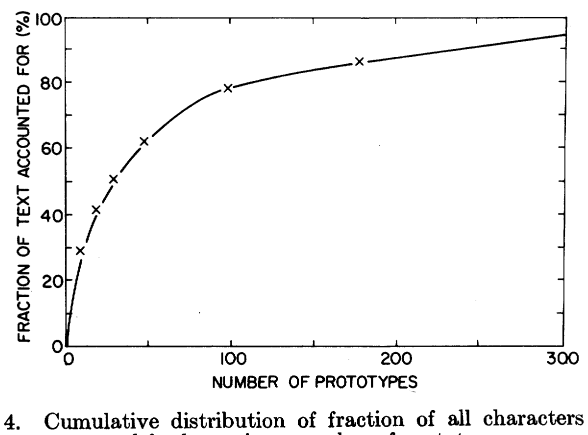


Fig. 4. Cumulative distribution of fraction of all characters accounted for by a given number of prototypes.

TABLE I
TYPICAL CORRELATION RESULTS ON SHORT SEGMENTS OF TEXT

Material	Number of Characters	Average Length of Search	Fraction saved as prototypes
Spectrum '70 Feb. p. 114	2819	54.3	0.14
ALR Vol. 91 p. 43	2374	58.6	0.17
Patent 2802067	2000	48.7	0.14

3) The video information for the prototypes could be coded by run length [3], predictive local operators [4], or sequential feature analysis [5]. Even the simplest of these processes, run-length coding (to eliminate the white spaces above and below the characters), could be expected to yield a threefold reduction in the number of bits per prototype.

These three subsidiary compression methods are applied to three completely independent components of the "message" (the prototype labels, the character position coordinates, and the prototype video). Hence, there will be no interference to reduce the saving that would result from each method considered individually. The techniques mentioned are well understood. For the same body of text for which our previous calculation was made, the expected overall compaction ratio can be estimated as follows:

$$\frac{200 \times 200 \times 8.5 \times 11 \times 1/6}{60 \times 50 + 1000 \times (4 + 8)} \simeq 40:1.$$

These measures have not been implemented because potentially even greater compression can be achieved by resorting to a high-resolution stable scanner. As such scanners usually operate in a raster mode, this would require reprogramming all of the line-finding and character-scanning routines, an undertaking beyond the scope of this project. We do know, however, from experimentation with a drum scanner [6], that in good-quality print very few adjacent characters actually touch: with a small enough scanning spot ($\simeq$2 mil), it is possible to find a "white path" in almost every case. Since many of our prototypes resulted from improperly segmented characters, such a device should substantially reduce the number of prototypes generated.

Table I shows the fraction of prototypes generated (*not* the asymptotic rate) for various short segments of text.

RECONSTRUCTION

The method outlined does not, of course, provide for perfect reconstruction of the digitized video. Nevertheless, the threshold on the degree of correlation required to assign a character to a prototype constitutes an upper bound on the Hamming distance between the original video and the reconstituted character. In the experiment described above, 90-percent correlation was required; since this is the admissible minimum, the average deviation was probably no more than about 5 percent. Although in other compaction schemes even a 5-percent error tends to result in ragged noisy characters, in our case the visual results are more pleasing since every character is the replica of some actual character.

IMPLEMENTATION

This compaction method shares a great many requirements (such as page transport, scanner, format control, segmentation, and video preprocessing) with optical character recognition (OCR) systems, and could be implemented in any of the major OCR machines.

An advantage of using a bona-fide OCR system for the compaction is that this would facilitate conversion to full alphanumeric recognition if this appears desirable. The conversion might take place along the lines described in [1], since it appears unlikely that a noninteractive system can be developed in the foreseeable future for material of the complexity of most textual applications. The interactive display terminal necessary for this purpose is already present in the reject entry feature of all OCR systems.

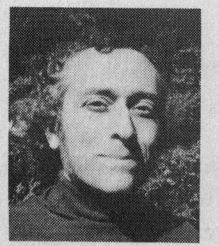
Other possible means of increasing the processing speed in compaction would be to use a fast-Fourier-transform processor [7] or an associative memory [8]. The latter, if provided with an inexact match feature, would present a particularly elegant method of implementation.

For encoding material comprising line drawings, mathematical formulas, and other nontextual segments, the method described in this paper would have to be combined with general purpose compaction algorithms such as those described in [9]. Although such algorithms yield a compaction ratio of only four or five to one on printed text, they have the advantage of being applicable to any scanned black-and-white image.

REFERENCES

[1] R. N. Ascher *et al.*, "An interactive system for reading unformatted printed text," *IEEE Trans. Comput.*, vol. C-20, pp. 1527–1543, Dec. 1971.
[2] D. A. Huffman, "A method for the construction of minimum redundancy codes," *Proc. IRE*, vol. 50, pp. 1098–1101, May 1962.
[3] S. W. Golomb, "Run-length encoding," *IEEE Inform. Theory* (Corresp.), vol. IT-12, pp. 399–401, July 1966.
[4] P. Elias, "Predictive coding," *IRE Trans. Inform. Theory*, vol. IT-1, pp. 16–33, Mar. 1955.
[5] D. Barnea, "Application of sequential decision techniques to two-dimensional image compaction," in preparation.
[6] D. R. Thompson, "An IBM special cartographic scanner," in *Proc. ASP-ACSM Conv.* (Washington, D. C.), Mar. 1967.
[7] G. D. Bergland, "Fast Fourier transform hardware implementation—An Overview," *IEEE Trans. Audio Electroacoust.* (*Special Issue on Fast Fourier Transform*), vol. AU-17, pp. 104–108, June 1969.
——, "Fast Fourier transform hardware implementation—A

survey," *IEEE Trans. Audio Electroacoust.* (*Special Issue on Fast Fourier Transform*), vol. AU-17, pp. 108–119, June 1969.
[8] R. H. Fuller, "Associative parallel processing," in *1967 Spring Joint Computer Conf., AFIPS Conf. Proc.*, vol. 30. Washington, D. C.: Spartan, 1967, pp. 471–475.
[9] L. R. Bahl and H. Kobayashi, "Image data compression by predictive coding," *IBM J. Res. Develop.*, vol. 18, pp. 164–179, Mar. 1974.



R. N. Ascher was born in New York, N. Y., in 1923. He received the B.A. and M.A. degrees from Columbia University, New York, N. Y.

Following some years of graduate study in mathematics at Columbia, he spent eight years in Louisiana performing executive functions in an oil and gas producing company. In 1959 he joined the Scientific Computations Laboratory at the IBM Development Laboratory, Endicott, N. Y. Since 1967 he has been on the staff at the IBM Thomas J. Watson Research Center, Yorktown Heights, N. Y.

George Nagy (SM'73) was born in Budapest, Hungary, in 1937. After early schooling in Hungary, Italy, France, and Canada, he obtained the B.S. degree in engineering physics and the M.S. degree in electrical engineering, both from McGill University, in 1959 and 1960, respectively, and the Ph.D. in electrical engineering from Cornell University, Ithaca, N. Y., in 1962.

Between 1963 and 1972 he was a Staff Member of the IBM Thomas J. Watson Research Center, Yorktown Heights, N. Y., conducting experiments in character recognition and, more recently, in remote sensing. In addition to several survey papers in the general area of pattern recognition, he has published articles on adaptive devices, analog matrix multipliers, image processing, feature extraction, the classification of Chinese ideographs, the use of context in reading machines, nonsupervised adaptive schemes in character recognition, and multispectral data processing. He is currently Professor and Chairman of the Department of Computer Science, University of Nebraska, Lincoln.

Finding Prototypes for Nearest Neighbor Classifiers

CHIN-LIANG CHANG, MEMBER, IEEE

***Abstract*—A nearest neighbor classifier is one which assigns a pattern to the class of the nearest prototype. An algorithm is given to find prototypes for a nearest neighbor classifier. The idea is to start with every sample in a training set as a prototype, and then successively merge any two nearest prototypes of the same class so long as the recognition rate is not downgraded. The algorithm is very effective. For example, when it was applied to a training set of 514 cases of liver disease, only 34 prototypes were found necessary to achieve the same recognition rate as the one using the 514 samples of the training set as prototypes. Furthermore, the number of prototypes in the algorithm need not be specified beforehand.**

***Index Terms*—Discriminant functions, generation of prototypes, minimal spanning tree algorithm, nearest neighbor classifiers, pattern recognition, piecewise linear classifiers, recognition rates, test sets, training sets.**

I. INTRODUCTION

ASSUME that an n-dimensional vector in a Euclidean space is a pattern. Let us also assume that there are r possible classes. The problem of designing a classifier for pattern recognition can be stated as follows: find r functions, $g_1,\cdots,g_r$, such that a pattern x is in class i if $g_i(x)$ is the optimal value among $g_1(x),\cdots,g_r(x)$. Each of these $g_1,\cdots,g_r$ is called a *discriminant function* [6]. There are many types of discriminant functions. In this paper, we shall consider classifiers based on nearest neighbor discriminant functions described below.

For $i = 1,\cdots,r$, let $p_i^1,\cdots,p_i^{k_i}$ be vectors in an n-dimensional Euclidean space E^n. If a discriminant function g_i is of the form

$$g_i(x) = \min \{d(x,p_i^1),\cdots,d(x,p_i^{k_i})\} \tag{1}$$

where $d(x,p_i^j)$ is a distance between x and p_i^j, g_i is called a *nearest neighbor discriminant function*. Note that $p_i^1,\cdots,p_i^{k_i}$ are often called *prototypes* (*reference points*) for class i. A classifier based on a set of nearest neighbor discriminant functions is called a *nearest neighbor classifier* [2], [5]. A nearest neighbor classifier assigns an unknown pattern to the class of the closest prototype. That is, a pattern x is assigned to class i if $g_i(x)$ is the smallest value among $g_1(x),\cdots,g_r(x)$. Although in a nearest neighbor classifier any distance measurement can be used, we shall restrict ourselves to the Euclidean distance.

In the sequel, for a pattern x, we shall use class (x)

Manuscript received November 29, 1973; revised March 4, 1974.
C.-L. Chang was with the Heuristics Laboratory, Division of Computer Research and Technology, National Institutes of Health, Bethesda, Md. 20014. He is now with IBM Research Laboratory, San Jose, Calif. 95193.