A Mathematical Theory of Man–Machine Document Assembly

SHMUEL S. OREN, MEMBER, IEEE

Abstract—**A mathematical model of document assembly using a computerized word processing system is introduced. In this process the operator prepares a typed document by assembling it from segments of text which are either retrieved from a file and copied on to the document or retyped by the operator. This model is used to obtain the optimal operation strategy, and the expected processing time as a function of the document's length and the man–machine parameters. Based on these results, some criteria are suggested for evaluating alternative word processors with respect to this application and for determining tradeoffs involved in designing such systems.**

I. Introduction

COMPUTER-ASSISTED word processing plays a key role in todays office environment. This computer application is based on the evolution of the basic editing features originally introduced in time-sharing systems to enable on-line program editing. Systems representing some of the early efforts in this area are described by Callahan and Grace [2] and by Magnuson [5]. Englebart and English [3] describe a more general system for "augmenting human intellect," in which text manipulation capabilities form a substantial part. Presently, computer-assisted text editing systems are available on a commercial basis as part of time-sharing computer service, as special purpose shared-logic systems driven by in house minicomputers, and in the form of numerous stand alone units (see, for example, [8], [9]). Although the specific configuration of word processors varies widely, they always consist of a keyboard, some sort of display such as a sheet of paper, a CRT, etc., storage for file and buffer, logic with a capability to search, retrieve, and edit stored information, and a hard copy printer.

The most important applications of such systems are repetitive typing, text editing, and document assembly (from canned text segments stored in the system's memory). Their effectiveness with respect to these applications is, therefore, a major factor in evaluating the merits of such systems.

The Word Processing Institute publishes regularly in its *Word Processing Report* qualitative evaluations of the various commercial word processing systems. A somewhat more quantitative cost benefit analysis has been published by Solnik and Jenkins [7], which compares IBM's Administrating Terminal System (ATS) with their Magnetic Tape Selectric Typewriter (MTST). However, there is no general theory available (to the best knowledge of the author) that will provide the analytic framework for such evaluations.

Manuscript received November 7, 1973; revised April 24, 1975. An earlier version of this paper was presented at the Fourth International Symposium on Computer and Information Systems (COINS-72), Miami, Fla., December 1972.

The author is with the Xerox Corporation Palo Alto Research Center, Palo Alto, Calif. 94304 and the Department of Engineering–Economic Systems, Stanford University, Stanford, Calif.

The work reported in this paper is a part of an effort to devise such a theory by modeling the undergoing processes in specific word processing applications. While the text editing application is treated in [6], this paper focuses on document assembly. This process is based on the capability to store frequently occuring text segments so that they can later be retrieved and copied onto the document rather than being typed repeatedly. Preparing the document on such a system may thus be viewed as assembling it from stored text segments. The paper introduces a mathematical model of man–machine document assembly where the machine is assumed to contain the basic elements outlined earlier and is characterized in terms of general parameters such as retrieval rate, display rate, etc.

The model is first described as a simulation model. Then, with some simplifying assumptions, a general mathematical representation is obtained. Analysis and further development of this representation for an idealized case leads to an expression for the minimum time of assembling a given length document in terms of the parameters of the system. This result is then used to devise criteria for evaluating word processing systems.

II. Description of the Model and Its Mathematical Representation

As indicated in the introduction, the document assembly process is based on having a collection of "canned" text segments such as phrases, paragraphs, words, etc., stored in a file, that can be retrieved and copied on to the processed document rather than being retyped. The advantage of implementing this capability clearly depends on the length of the text segment, the typing skill of the operator, the ease of retrieving and copying, and the familiarity of the operator with the system. Conceivably there is a segment threshold length, depending on the systems parameters, below which an operator will prefer to retype a text segment rather than bothering to retrieve it. This threshold length may change with time as the operator becomes more familiar with the system and the file content, cutting down the retrieval time. In this model the threshold length is used to model the behavior of the operator.

It is convenient to assume that any given text can be broken into segments stored in the file. For this assumption to be valid it is sufficient to assume that the set of all individual characters is included in the file. The generality of the model is not restrained by the later assumption since

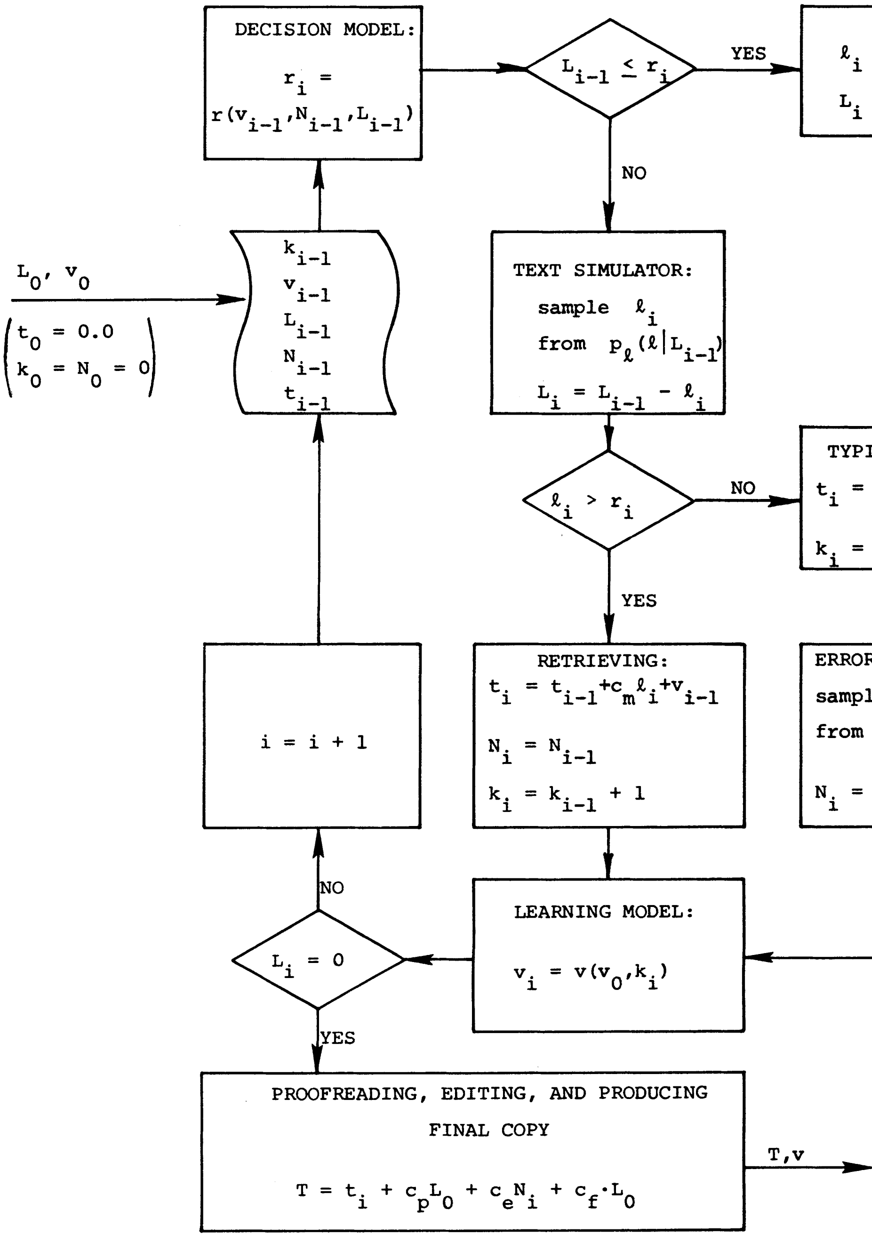

Fig. 1. Simulation flow diagram.

the threshold length is expected to be always greater than one character length, and hence it does not matter whether a real file actually contains this set. This hypothetical extension of the file allows us to view the document assembly process as a multistage control process, in which the processed document is perceived by the operator as a sequence of segments stored in the file. Each of these segments can be processed by being retrieved or typed. The operator controls the process by setting at each stage the threshold length which determines how the next perceived segment is processed.

The simulation block diagram in Fig. 1 illustrates a model of the process described above. For the sake of generality, the mathematical description of the various "blocks" is given in a functional form, which may be replaced by experimental data or by hypothesized specific mathematical relations.

Following is the list of variables and parameters used in the model and a detailed description of the various blocks in the diagram:

- L_i remaining text to be processed (number of characters),
- l_i perceived length of current text segment (number of characters),
- N_i cumulative number of errors,
- n_i number of errors in current segment,
- r_i current threshold length (number of characters),
- v_i current mean search time,
- k_i cumulative number of past retrievals,
- t_i elapsed time,
- T total processing time,
- c_m machine display rate (time per character),
- c_t typing rate (time per character),
- c_p proofreading rate (time per character),

c_e mean editing rate (time per error),
c_f final copy production rate (time per unit length),
i segment count.

Text Simulator: As mentioned above it is assumed that the processed document is perceived sequentially by the operator as a series of segments from the file. This process is simulated by the "text simulator" that generates the successive segments l_i as samples from a conditional probability distribution over l, given L_{i-1}, namely, $p_l(l \mid L_{i-1})$. This distribution has to satisfy the condition $p_l(l \mid L) = 0$, for $l > L$, since no segment can be longer than the remaining text.

Decision Model: The control variable in the model is the threshold length r_i which is set by the operator at each cycle before the new segment l_i is perceived. The process of setting r_i is simulated by the "decision model," which is described mathematically as $r_i = r(v_{i-1},N_{i-1},L_{i-1})$. This model assumes that in setting r_i the operator may consider his current mean search time, the total number of errors he has already made, and the remaining length of document to be processed. Obviously, he also considers the system's parameters, which are included as constant parameters in the function $r(v,N,L)$.

The threshold length r_i determines whether the next segment is to be typed or retrieved. If the remaining text L_{i-1} is shorter or equals r_i, then it is treated as one segment and typed; otherwise l_i is perceived and L_i updated. If l_i is shorter than r_i, then it is typed; otherwise l_i is retrieved. A typed segment l_i increases the processing time t_i by $c_t l_i$ (the typing time). On the other hand, a retrieved segment l_i increases the processing time by v_{i-1} (the mean time required to locate it in the file) and by $c_m l_i$ (the time that it takes the machine to display the retrieved segment).

Error Simulator: Typed segments are subjected to typing errors. Those are simulated by the "error simulator," which generates the number of errors n_i in l_i as samples from a probability distribution over n, given l_i, namely, $p_n(n \mid l_i)$. This n_i is then used to update the cumulative number of errors N_i. Clearly, the distribution $p_n(n \mid l)$ has to satisfy the condition $p_n(n \mid l) = 0$, for $n > l$, since there cannot be more errors than characters.

Learning Model: The mean search time v_i is updated in the "learning model" to account for the experience gained by the operator in processing the segment l_i. The learning model is represented mathematically by $v_i = v(v_0,k_i)$. This model assumes that v_i depends on the operator's familiarity with the system, which is characterized by his initial v_0 and the cumulative number of retrievals. Obviously, v_i will also depend on the system's parameters and the operator learning ability. Those are included as constant parameters in the function $v(v_0,k)$. In practice, v_i will be bounded below by some V that is a parameter of the system. To include this phenomenon in the model, it is assumed that

$$\lim_{k\to\infty} v(v_0,k) = V.$$

The cycle described above is repeated until the entire document is processed. If no more text is left ($L_i = 0$), the whole document is proofread and edited and a final hard copy is produced. This adds to the processing time $c_p L_0$ for proofreading, $c_e N_i$ for editing, and $c_f L_0$ for producing the final copy.

The expected behavior of the model presented can be described analytically by an equation for deriving the expected processing time of a document in terms of its length and the parameters of the man–machine system. To simplify the mathematical representation, text length will be treated as a continuous variable.

Let us now define $T(v_0,k,N,L,r)$ to be the expected time required to process the L remaining length of a document (including proofreading, editing, and final copy production of that length) using threshold policy r and given that N errors have been made so far while k segments were retrieved, starting with a mean search time v_0. The total expected processing time of the document will then be

$$T(v_0,0,0,L_0,r) = T(v_0,L_0,r).$$

This definition implies that $T(v_0,k,N,L,r)$ satisfied the following recursive equations:

$$\begin{aligned} T&(v_0,k,N,L,r) \\ &= \int_0^r p_l(l \mid L) \left\{ \sum_{n=0}^{\infty} p_n(n \mid l)[(c_t + c_p + c_f)l \right. \\ &\quad \left. + nc_e + T(v_0, k, N+n, L-l, r)] \right\} dl \\ &\quad + \int_r^L p_l(l \mid L)[(c_m + c_p + c_f)l \\ &\quad + v(v_0,k) + T(v_0, k+1, N, L-l)]\, dl, \\ &\qquad \text{for } L > r \end{aligned} \tag{1a}$$

$$T(v_0,k,N,L,r) = (c_t + c_p + c_f)L + \sum_{n=0}^{\infty} p_n(n \mid L)c_e n, \quad \text{for } L \le r. \tag{1b}$$

As indicated earlier the threshold may depend on the rest of the variables according to some decision rule, i.e., $r = r(v_0,k,N,L)$. Equations (1a) and (1b) express the remaining processing time for L in terms of the processing time for the next segment and the expected remaining processing time for $L - l$.

In the case $L > r$, the first term accounts for the possibility $l \le r$, in which case the segment is typed, and thus k does not increase, but N increases by n with probability $p_n(n \mid l)$. The second term in (1a) accounts for the possibility $l \ge r$, in which case the segment is retrieved, k increases by one, and N is unchanged. In the case $L \ge r$, all the remaining text L is treated as one segment, and it is typed. The proofreading, editing, and final copy production time are added to each individual segment.

For further simplification, we assume that the expected number of errors in a typed segment is proportional to the length of that segment, i.e.,

$$\sum_{n=0}^{\infty} p_n(n \mid l)n = \mu l.$$

This is equivalent to saying that the probability for having an error in any character is the same. μ can be interpreted as the expected number of errors in a unit length segment or as the probability of an error in any typed character. With this assumption, (1a) and (1b) can be rewritten in the form

$$T(v_0,k,N,L,r) = \int_0^r p_l(l \mid L) \cdot \left[\hat{c}_t l + \sum_{n=0}^{\infty} p_n(n \mid l) T(v_0, k, N+n, L-l, r)\right] dl + \int_r^L p_l(l \mid L) \cdot [\hat{c}_m l + v(v_0,k) + T(v_0, k+1, N, L-l, r)]\, dl, \quad \text{for } L > r \tag{2a}$$

$$T(v_0,k,N,L,r) = \hat{c}_t L, \qquad \text{for } L \le r \tag{2b}$$

where $\hat{c}_t = c_t + c_p + c_f + \mu c_e$ and $\hat{c}_m = c_m + c_p + c_f$.

For future reference we denote the right side of (2a) by $\tau(v_0,k,N,L,r)$. Using the facts that $p_l(l \mid L) = 0$, for $L < l$, and $T(v_0,k,N,0,r) = 0$, one can easily show that $\tau(v_0,k,N,L,L) = \hat{c}_t L$. This enables us to rewrite (2a) and (2b) in the compact form

$$T(v_0,k,N,L,r) = \begin{cases} \tau(v_0,k,N,L,r), & \text{for } L > r \\ \tau(v_0,k,N,L,L), & \text{for } L \le r. \end{cases} \tag{3}$$

III. Optimal Threshold

A possible application of the model described in Fig. 1 and represented by (1) is to examine alternative decision models. Of particular interest is the effect of choosing r by alternative strategies on the expected processing time of a given length document for a particular set of parameters describing the man–machine system. This leads to the problem of determining the optimal threshold stretegy $\bar{r}(v_0,k,N,L)$ that will minimize the total expected processing time $T(v_0,L_0)$. Determining the minimal expected processing time $\bar{T}(v_0,L_0)$ is also important since it may be used as a criterion for comparison between alternative systems. The problem of determining the optimal threshold can be formulated as a dynamic programming problem. By Bellman's [1] "Principle of Optimality" the optimal strategy $\bar{r}(v_0,k,N,L)$ is such that, starting from any values of k, N, and L, it will minimize the expected processing time for the remaining part of the document, independently of what strategy was used before that. This implies that the optimal threshold, and consequently the minimum processing time, are independent of the cumulative number of past errors N, since they do not affect the processing in the future. (This cannot be said about k, which does affect the future processing through the learning model.)

In view of this argument, we modify our notation and denote the minimum processing time of L, given v_0 and k, by $T(v_0,k,L,\bar{r})$. Let $\bar{\tau}(v_0,k,l,r)$ denote the right side of (2a) after replacing

$$\sum_{n=0}^{\infty} p_n(n \mid l) T(v_0, k, N+n, L-l, r)$$

and $T(v_0, k+1, N, L-l, r)$ with $T(v_0, k, L-l, \bar{r})$ and $T(v_0, k+1, L-l, \bar{r})$, respectively. Then again by the principle of optimality and (3),

$$T(v_0,k,L,\bar{r}) = \min\left[\min_{0 \le r < L} \bar{\tau}(v_0,k,L,r), \bar{\tau}(v_0,k,L,L)\right] = \min_{0 \le r \le L} \bar{\tau}(v_0,k,L,r) \triangleq \tau(v_0,k,L,\bar{r}) \tag{4}$$

where $T(v_0,k,0,\bar{r}) = 0$. For $\bar{r}$ to minimize $\bar{\tau}(v_0,k,L,r)$ subject to $0 \le r \le L$, it has to satisfy the necessary conditions:

$$\left.\frac{d\bar{\tau}(v_0,k,L,r)}{dr}\right|_{r=\bar{r}} \begin{cases} \le 0, & \text{if } \bar{r} = L \\ = 0, & \text{if } 0 \le \bar{r} \le L \\ \ge 0, & \text{if } \bar{r} = 0 \end{cases} \tag{5}$$

where

$$\frac{d\bar{\tau}(v_0,k,L,r)}{dr} = p_l(r \mid L)\{[\hat{c}_t r + T(v_0, k, L-r, \bar{r})] - [\hat{c}_m r + v(v_0,k) + T(v_0, k+1, L-r, \bar{r})]\}. \tag{6}$$

Let $\hat{r} = \hat{r}(v_0,k,L)$ be such that

$$(\hat{c}_t - \hat{c}_m)\hat{r} - v(v_0,k) + T(v_0, k, L-\hat{r}, \bar{r}) - T(v_0, k+1, L-\hat{r}, \bar{r}) = 0. \tag{7}$$

Then, by (5) and (6),

$$\bar{r} = \begin{cases} L, & \text{if } L < \hat{r} \\ \hat{r}, & \text{if } 0 \le \hat{r} \le L \\ 0, & \text{if } \hat{r} \le 0. \end{cases} \tag{8}$$

By virtue of the learning effect incorporated in the model, an additional retrieval reduces the expected processing time, and hence $T(v_0, k, L-\hat{r}, \bar{r}) \ge T(v_0, k+1, L-\hat{r}, \bar{r})$. Thus from (7) it follows that

$$\hat{r}(v,k,L) \le \frac{v(v_0,k)}{\hat{c}_t - \hat{c}_m} = \frac{v(v_0,k)}{c_t + \mu c_e - c_m}. \tag{9}$$

Furthermore, as $k \to \infty$, $v(v_0,k) \to V$, and

$$[\bar{T}(v_0, k, L-\hat{r}, \bar{r}) - \bar{T}(v_0, k+1, L-r, \bar{r})] \to 0$$

thus

$$\lim_{k \to \infty} \hat{r}(v_0,k,L) = \frac{V}{\hat{c}_t - \hat{c}_m} = \frac{V}{c_t + \mu c_e - c_m} \triangleq \tilde{r}. \tag{10}$$

The justification for computerized word processing is based on the assumption that $c_m < c_t$, i.e., machine display rate is higher than typing rate. This implies $\tilde{r} > 0$, which allows us to rewrite (8) for $k \to \infty$ as

$$\bar{r} = \min[\tilde{r}, L]. \tag{11}$$

Since however, $T(v_0,k,N,L,r) = T(v_0,k,N,L,L)$, for $r \ge L$, the threshold policy described by (11) is equivalent to $\bar{r} = \tilde{r}$.

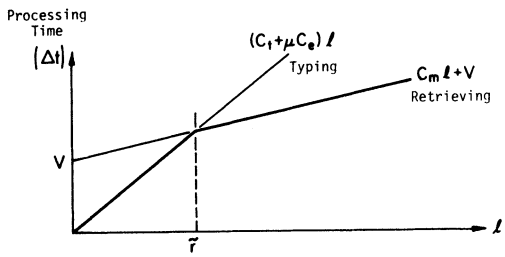

Fig. 2. Processing time via typing versus retrieving as function of segment length.

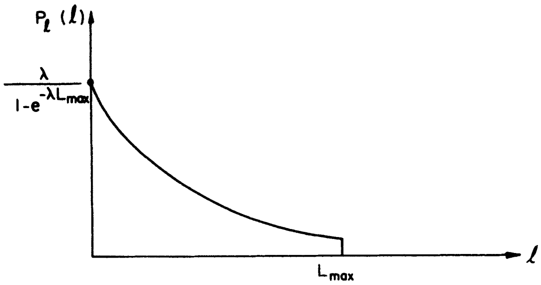

Fig. 3. Illustration of truncated exponential distribution for segment length.

The asymptotic optimal threshold policy described corresponds to the case where the operator reached the mean search speed limit V. In such a case, there is no future benefit from a present retrieval, and the threshold $\bar{r}$ is set so that each segment is processed by the method taking the least amount of time. This is illustrated in Fig. 2, which clearly shows that $\bar{r}$ is the breakpoint where retrieving becomes more economical than typing.

IV. A Special Case: Trained Operator Exponential File

In this section we introduce some further simplifying assumptions that will enable us to obtain an analytic solution to (3) for the expected processing time.

First, it is assumed that the file contains text segments ranging in length from zero to $L_{\max}$ ($L_{\max}$ being an upper bound on the length of documents processed) and that the occurrence frequency of these segments in text is described by a truncated exponential probability density function over their length

$$p_l(l) = \begin{cases} \lambda e^{-\lambda l}/(1 - e^{-\lambda L_{\max}}), & \text{for } l \le L_{\max} \\ 0, & \text{for } l > L_{\max}. \end{cases} \quad (12)$$

This distribution, illustrated in Fig. 3, captures the intuitive notion that short "canned" phrases are more universal and therefore more likely to occur in a document than long stored paragraphs. The parameter λ in the above distribution characterizes the richness of the file. A richer file contains a greater variety of text segments which enables one to assemble a document from fewer segments. This increases the mean segment length $\bar{l}$, which for the above distribution is

$$\bar{l} = \frac{1}{\lambda} - \frac{L_{\max}}{e^{\lambda L_{\max}} - 1}. \quad (13)$$

A larger $\bar{l}$ corresponds to a smaller λ, so the smaller λ is, the richer is the file. For the extreme case $\lambda = 0$, $p_l(l)$ is a uniform distribution. The other extreme case $\lambda = \infty$ corresponds to an empty or an irrelevant file since $\bar{l} = 0$. The conditional probability distribution $p_l(l \mid L)$ is obtained from (12) as

$$p_l(l \mid L) = \begin{cases} p_l(l \mid l \le L) = \lambda e^{-\lambda l}/(1 - e^{-\lambda L}), & l \le L \\ 0, & l > L. \end{cases} \quad (14)$$

The trained operator is assumed to have the following characteristics.

1) His mean search time reaches the machine potential V, thus there is no learning effect.

2) If there is more than one segment in the file matching the forthcoming text, the operator perceives the longest one. (This assumption is necessary to make $p_l(l \mid L)$ uniquely dependent on the file and L.)

3) He uses a fix threshold r, although it is not necessarily the optimal one.

Under these assumptions the expected processing time is only dependent on L and r, so it is denoted by $T(L,r)$. Equations (2a) and (2b) then reduce to

$$T(L,r) = \frac{\lambda}{1 - e^{-\lambda L}} \left\{ \int_0^r e^{-\lambda l}[\hat{c}_t l + T(L - l, r)]\, dl \right. \\ \left. + \int_r^L e^{-\lambda l}[\hat{c}_m l + V + T(L - l, r)]\, dl, \right. \qquad \text{for } r < L \quad (15a)$$

$$T(L,r) = \hat{c}_t L, \qquad \text{for } r \ge L. \quad (15b)$$

Using the identity

$$\int_0^L e^{-\lambda l} T(L - l, r)\, dl = \int_0^L e^{-\lambda(L-l)} T(l,r)\, dl \quad (16)$$

along with some straightforward manipulations, (15a) can be reduced to the form

$$e^{\lambda L}\{T(L,r) + [(\hat{c}_t - \hat{c}_m)(r - 1/\lambda) - V]e^{-\lambda r} - \hat{c}_t/\lambda\} \\ = T(L,r) - \hat{c}_m(L - 1/\lambda) + V + \lambda \int_0^L e^{\lambda l} T(l,r)\, dl, \qquad \text{for } r < L. \quad (17)$$

Differentiating (17) with respect to L, then replacing V by $\bar{r}(\hat{c}_t - \hat{c}_m)$ and collecting terms, results in

$$R(L,r) \triangleq \frac{dT(L,r)}{dL} \\ = \hat{c}_m + (\hat{c}_t - \hat{c}_m)\frac{(1 - e^{-\lambda r}[1 + \lambda(r - \bar{r})])}{(1 - e^{-\lambda L})}, \qquad \text{for } r < L. \quad (18)$$

For $L > r$, (18) together with (10) can be used to evaluate $R(L,r)$, the marginal expected processing time per additional unit length of document, for a given system characterized by the parameters $\hat{c}_t$, $\hat{c}_m$, V, and λ. For $L \le r$,

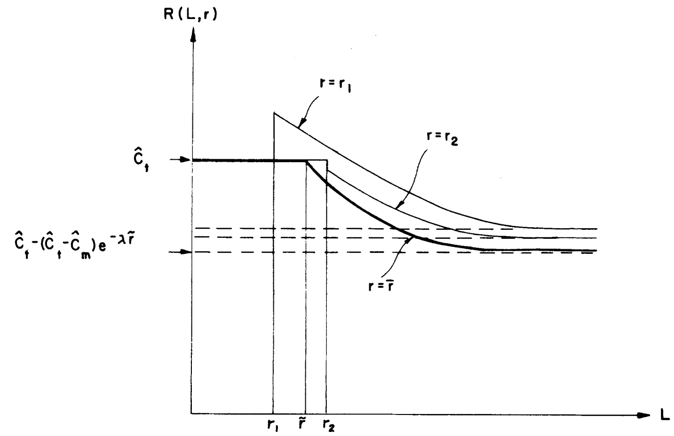

Fig. 4. Marginal expected processing time per unit length of text as function of document's total length.

it is obvious from (15b) that $R(L,r) = \hat{c}_t$. Fig. 4 illustrates qualitatively the dependency of $R(L,r)$ on L. We note that as L increases, $R(L,r)$ approaches an asymptotic value $R(\infty,r)$, which may be derived by letting $L \to \infty$ in (18):

$$R(\infty,r) = \lim_{L\to\infty} R(L,r) = \hat{c}_t - (\hat{c}_t - \hat{c}_m)[1 + \lambda(r - \tilde{r})]e^{-\lambda r}. \quad (19)$$

For the optimal threshold policy

$$R(\infty,\tilde{r}) = \hat{c}_t - (\hat{c}_t - \hat{c}_m)e^{-\lambda\tilde{r}}. \quad (20)$$

We also note that although in general $R(L,r)$ has a discontinuity at $L = r$, $R(L,\tilde{r})$ is continuous.

The total expected processing time $T(L,r)$, for $L > r$, can be obtained by integrating (18) with the boundary condition $T(r,r) = \hat{c}_t r$. This yields

$$T(L,r) = \hat{c}_m L + (\hat{c}_t - \hat{c}_m)\left\{r + \ln\left(\frac{e^{\lambda L} - 1}{e^{\lambda r} - 1}\right) \cdot [(1 - e^{-\lambda r})/\lambda - e^{-\lambda r}(r - \tilde{r})]\right\}, \quad \text{for } L > r. \quad (21)$$

V. Criteria for System Evaluation

In this section we follow the same line of reasoning used in [6] for developing evaluation criteria. Since the processing time has a major effect on the total cost of an assembled document, it seems reasonable to measure the relative efficiency of alternative systems in terms of the average processing time per unit document length. For a given length document the expected processing time per unit length is given by $T(L,r)/L$ and may be evaluated analytically under the assumptions of the previous section. Suppose now that $p(L)$ is a probability density function representing document length statistics in the particular environment under consideration. Then the average editing time per unit length will be

$$E(r) = \int_0^\infty \frac{T(L,r)}{L}\, p(L)\, dL. \quad (22)$$

This criterion is a function of the system's parameters c_m, c_t, c_e, c_p, λ, and the threshold r. Since r may be regarded as an input to the system, we would like to eliminate it from our criterion. This may be done by substituting $r = \tilde{r}$ in (22). The resulting criterion $E(\tilde{r})$, which depends only on the system parameters, may be interpreted as the "ideal" average processing rate of the system under optimal operation. This criterion can be used for comparative evaluation of a text editing system in a given environment and for determining tradeoffs among the parameter for design purposes. Unfortunately, even if we assume a simple distribution for $p(L)$ and use the results derived in Section IV, the expression for $E(\tilde{r})$ becomes far too complicated. In view of the simplistic assumptions, this would be a crude measure anyhow, and the effort required to evaluate it is not worthwhile. A simpler criterion can be obtained by considering the asymptotic behavior of $T(L,r)/L$ under the assumptions of Section IV. It can be easily shown from (18) and (21) that

$$\frac{T(L,r)}{L} \geq R(L,r), \qquad \text{for any } L \in [0,\infty]. \quad (23)$$

Furthermore, $T(L,r)/L$ is monotonically decreasing and

$$\lim_{L\to\infty} \frac{T(L,r)}{L} = R(\infty,r). \quad (24)$$

It is also clear that

$$R(\infty,\tilde{r}) = \min_{r\in[0,\infty]} R(\infty,r). \quad (25)$$

Thus

$$\frac{T(L,r)}{L} \geq R(\infty,\tilde{r}), \qquad \text{for any } r,\, L \in [0,\infty] \quad (26)$$

with equality for $r = \tilde{r}$ and $L \to \infty$. Consequently,

$$E(\tilde{r}) \geq R(\infty,\tilde{r}), \qquad \text{for any pdf } p(L). \quad (27)$$

In view of the above considerations and the simple form of (20), it seems attractive to use $R(\infty,\tilde{r})$ as a crude criterion for evaluating the relative effectiveness of text editing systems and determining crude design tradeoffs for such systems. The explicit dependency of $R(\infty,\tilde{r})$ on the man–machine parameters is obtained by substituting $\hat{c}_m$, $\hat{c}_t$, and $\tilde{r}$ in (20) in terms of these parameters. These yield

$$R(\infty,\tilde{r}) = c_t + c_e + c_p + c_f - (c_t + \mu c_e - c_m) \cdot \exp\left[\frac{-\lambda v}{(c_t + \mu c_e - c_m)}\right]. \quad (28)$$

An immediate implication that would follow from using $R(\infty,\tilde{r})$ as a design criterion is that the file should be such as to minimize λV. Let us assume, for instance, that $1/\lambda$, which is approximately the mean segment length for large $L_{\max}$, increases linearly with the total file length $\mathscr{L}$, i.e., $\lambda = \alpha/\mathscr{L}$. Then, if linear searching is used, V, the mean search time, would also increase linearly with the length of the file, i.e., $V = \beta\mathscr{L}$. In such a case, extending the file would not improve the performance of the system with respect to the above criterion since λV remains constant. It follows that under the above circumstances there is no point in building a large file unless one can improve the search techniques.

It is important to note that all the assumptions leading to the derivation of $E(r)$ are idealizations of the system so that $R(\infty,\tilde{r})$ is a true bound on the processing rate of a system even if it does not justify these assumptions.

Another criterion that one may want to consider particularly for design purposes is the average expected cost per unit length of document. It is conceivable that the average processing time per unit length of a document may be reduced at a cost. This can be done, for instance, by reducing λ, i.e., enriching the file, which requires a larger storage. Another alternative is to reduce the search time V by using a faster memory and more sophisticated software. The processing time may also be reduced by increasing the display speed, the final printing speed, and the editing speed, (i.e., reducing c_m,c_f,c_e) by using a faster printer, a CRT display, and a more sophisticated curser. One may also consider reducing the processing time by hiring highly skilled operators who type faster and make fewer errors (this will reduce c_t and μ). Clearly, each of the alternatives suggested for reducing the processing time will increase the manufacturing cost of the system and consequently its operating cost per unit time. In general, this operating cost is a function of the man–machine parameter and will hence be denoted $\mathscr{C}(c_m,c_t,c_e,c_f,c_p,V,\mu,\lambda)$. Thus, for a stand-alone system on which processing is performed sequentially, the average expected cost per unit document length is given by $E(r)\mathscr{C}(\cdot)$. Following the previous argument, $E(r)$ is replaced by its lower bound $R(\infty,\tilde{r})$ yielding the cost criterion

$$C = \mathscr{C}(c_m,c_t,c_e,c_f,c_p,V,\mu,\lambda) \cdot \left[c_t + c_e + c_p + c_f - (c_t + \mu c_e - c_m) \cdot \exp\left(\frac{-\lambda V}{c_t + \mu c_e - c_m}\right)\right]. \tag{29}$$

Varying the configuration assumptions by assuming, for instance, a shared file, off-line printing, etc., will lead to different cost criteria.

Criteria of the type given by (29) might be used as objective functions for optimal design. Assuming that $\mathscr{C}(\cdot)$ is available, one may obtain the optimal parameter for a system by minimizing C with respect to these parameters and subject to the constraints imposed by feasibility considerations. Again, in view of the idealizing assumptions, C can be regarded as a lower bound on the true cost per unit length of processed document.

VI. Conclusion

This paper introduces a mathematical model of the document assembly process using computerized word processing equipment. The model intends to provide an analytic framework for evaluating such equipment with respect to these specific applications either through simulation or by using criteria of the type introduced in Section V. Criteria of this type may be particularly useful in the design phase of such system for determining crude design tradeoffs. Although the assumptions leading to these criteria are somewhat simplistic, they are all idealizations of the reality, and therefore, the suggested performance measures are true bounds on the performance of real systems. It should be emphasized, however, that the specific criteria given in Section V are based on certain assumptions regarding the system's configuration, and they have to be modified properly for other configurations.

References

[1] E. R. Bellman and E. S. Dreyfus, *Applied Dynamic Programming.* Princeton, N.J.: Princeton Univ., 1962.
[2] N. D. Callahan and G. L. Grace, "AUTODOC: Computer-based assistance for document production," in *Proc. A.C.M. Nat. Meeting*, pp. 177–185, 1967.
[3] D. C. Englebart and W. K. English, "A research center for augmenting human intellect," *AFIPS Conf. Proc.*, vol. 33, pp. 395–410, 1968.
[4] R. A. Howard, *Dynamic Probabilistic Systems, I and II.* New York: Wiley, 1971.
[5] R. A. Magnuson, "Computer assisted writing," *Datamation*, vol. 1249, pp. 5257–5259, 1966.
[6] S. S. Oren, "A mathematical theory of man–machine text editing," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-4, pp. 258–267, May 1974.
[7] H. Solnik and L. W. Jenkins, "Evaluation of the IBM administrative terminal system and magnetic tape selectric typewriter for text processing," *J. of Chem. Doc.*, vol. 11, no. 3, 1971.
[8] *Word Processing Report*, 2, 12, and 3, 1, and 3, 7, Geyer-McAllister Publications, Inc., 1972.
[9] *Word Processing, Text Editing, and Repetitive Typing Confidential Equipment Evaluation*, Geyer-McAllister Publications, Inc., 1971.