Beginning with a description of various degrees of computer dependency among workers, a model of the worker-computer process is constructed. The model demonstrates the characteristic forms of functional dependencies and suggests ways in which these dependencies can be evaluated.

Key among the many considerations discussed are such process characteristics as system congestion, needs and habits of users, and relative costs.

Productivity of computer-dependent workers

by D. N. Streeter

This paper examines relationships that arise within a process when that process becomes dependent on computer-based services. Questions to be discussed include how resources should be allocated among the following elements:

- User personnel costs—the salary and burden of computer-dependent workers.
- Computer system costs—the costs of obtaining, operating, and maintaining the computer service system.
- Interface costs—the cost of usability aids, such as interactive checkout facilities, debugging aids, and consulting specialists.

How should priority be assigned to the various jobs submitted to the computer?

How should a process be monitored? What measurements of system and usage are required to determine the proper policies and to determine the efficacy of the results?

The objective of the policies derived from this analysis is simply to facilitate the integration of computer-based services into organizational processes. Our means of achieving this integration harmoniously are the following:

- Avoidance of waste to the organization.
- Avoidance of frustration to the user.

User frustration arises largely from delays in receiving service or failure to achieve desired results. Both of these causes of frustration have economic as well as psychological manifestations. Our intention is to simultaneously reduce waste and frustration via the treatment of their economic indicators.

This work is an extension of the concepts of cost/benefit analysis applied to computer-based services, previously published by the author.[1] That earlier work focused on scientific computing services. Other studies of the effect of computing services on the productivity of engineers,[2] and programmers[3] were also conducted several years ago. (The substance of References 1 and 2 is available in Reference 4.) This paper attempts to extend the concepts of the earlier studies to a broader class of computer-dependent workers, to include other cost elements—such as the probability of successful use—and to further systematize the treatment of the various elements.

Description of a process

types of computer dependency

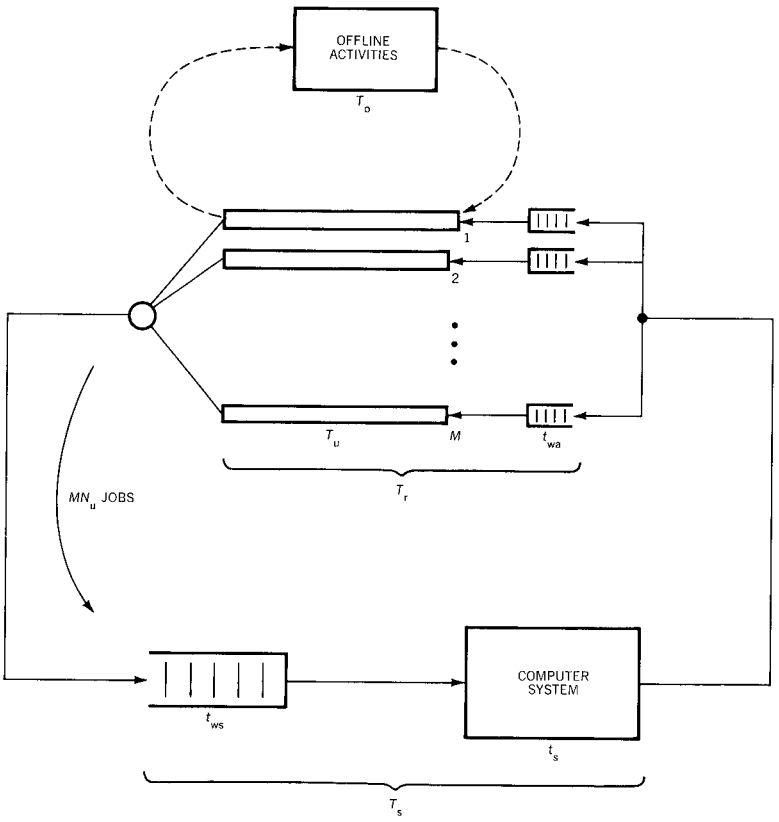
We assume that a process, such as we are studying, is an industrial or commercial function that requires a predictable number of successful computer runs per task. The mode considered here is primarily batch computer usage. We assume, however, that terminals are available for remote job submittal and checkout, when such use is justified. The user population is classified into four groups, for the purpose of our study.

Type A users are those members of the population who have made no use of the computer during the time period during which statistics are gathered. For type A people, the ratio of costs that should be allocated to computer systems relative to personnel costs is obviously zero.

Type B users are those who make some use of the computer services, but who are not dependent on these services in the sense to be defined later in this section. The important point about type B's work is that most of it is not computer-related, and, therefore, B can flexibly compensate for changes in computer service. If he must wait for computer results, B can rearrange his schedule so that his work output rate is not diminished.

For B, the calculation of a *system/personnel cost-allocation ratio* is a trivial matter. If, for example, a worker, whose personnel cost is $25 per hour, requires two overnight computer runs per week at $20 per run, then his system/personnel cost-allocation ratio is computed as follows:

Figure 1 Worker-computer model



$$\frac{\text{System costs}}{\text{Personnel costs}} = \frac{2 \times \$20}{\$25 \text{ per hour} \times 40 \text{ hours per week}} = 0.04$$

Type C users, on the other hand, are an extreme case of computer dependence because they work on one assignment at a time and use a single program repeatedly to carry out that assignment. Examples of such usage include debugging a program or using computer assistance in an iterative trial-and-error design procedure. People who work in such a single-stream mode place heavy demands on a computer facility for fast turnaround, because they do not have another assignment to turn to while waiting for computed results. Nevertheless, surveys often show a surprisingly large fraction of type C computer users. Reasons given for working only on a single assignment include the following:

- The existence of precedence constraints, i.e., one cannot work on tasks Y and Z without information derived from task X.

- The complexity of the tasks may prevent the handling of concurrent assignments without the degredation of product quality.
- The output of a given task on the critical path of a project may be needed so urgently that management directs the worker to do only that task, so as to minimize the *flow time* (the calendar time to task completion).

Type D users usually are the largest group of computer dependent workers, and they are characterized by the ability to handle two or more computer-dependent assignments concurrently. This concurrency relaxes somewhat the requirement for fast turnaround, assuming that the person can effectively interleave his assignments, which can be done if the tasks are mutually free of precedence constraints, and if the user can redirect his attention alternately so as to remain productive while waiting for computed results.

User types C and D, then, are referred to as computer-dependent workers since their productivity is directly related to the quantity and quality of computer support services. In this paper, we analyze the dynamics and economics of this worker-computer relationship by making use of the model shown—in its simplest form—in Figure 1.

process dynamics and economics

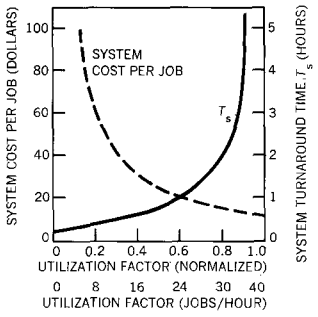
A population of M computer-dependent workers submits jobs to the the service facility. Before receiving service, the entering job must wait t_{ws} minutes for the completion of jobs that either have arrived earlier or for other reasons have higher priority. Then the job receives the t_s minutes of service it requires. The *job run time* t_s can be thought of as the time during which a job occupies the limiting bottleneck of the system or, equivalently, the time that subsequent jobs in the queue are delayed as a result of a given job's being serviced. The *system turnaround time* T_s is the sum of the time spent waiting for service t_{ws} and the job run time t_s.

After each job is serviced, it is returned to the user. Results are analyzed, modified, and/or augmented with new data, and then resubmitted for the next computer processing stage. We refer to the elapsed time the job is in the user's hands as the user resubmittal time T_r. Later in this paper, we consider the components of the resubmittal time in greater detail.

In short, the computing system and the user form a closed-loop process that contains a fast sequential service facility (computer) coupled to a set of slower parallel resubmittal paths (users). If—on the average—each of the M users have N_u concurrent active jobs, MN_u jobs are circulating in this user-system loop at any given time. N_u is called the *job concurrency level.*

Procedure for analyzing the process

Figure 2 System characteristics

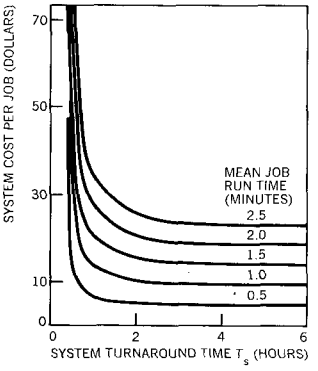


We now illustrate how one uses this process model, subject to the user population described, to investigate the productivity of computer dependent workers. One first determines his system characteristics such as are shown in Figure 2, which are the congestion and economic characteristics of a computer service system. The solid line is referred to as the congestion characteristic, which relates the mean system turnaround time to the utilization factor, i.e., the ratio of actual throughput to system throughput capacity. This characteristic is best determined empirically, by measuring the turnaround times delivered by the system that one is studying under various intensities of loading. If the system is operated at less than full capacity, in order to reduce the turnaround time, then higher charges per job must be imposed to recover system costs. The dashed line shows the economic cost-recovery characteristic.

system characteristic

In this paper, to make the analytical method clear, specific values of costs, times, etc. are assumed. The reader should remember that the particular results derived are valid only for those particular values assumed. Therefore, recalculation is required in each case, based on the relevant local data. In that spirit, we assume the following data:

Figure 3 System cost per job as a function of turnaround time



User personnel costs = $25 per hour

Computer system costs = $500 per hour

Mean job run time = 1.5 minutes

System congestion = as shown in Figure 3 characteristic

The form in which the system characteristics are most useful to us is one that shows directly the tradeoff between system costs per job and turnaround time. These data can be scaled from Figure 2 and plotted as shown in Figure 3, for a range of job sizes. The curves of Figure 3 show a fairly sharp knee. To the left of the knee, costs rise sharply because the system must be kept very lightly loaded to provide the short turnaround time that is characteristic of that region of the curves. To the right of the knee, however, costs do not decrease at a comparable rate because the loading is sufficiently heavy to keep the system busy most of the time.

usage characteristics

The next step is to determine the usage characteristics. Usage data are more difficult to obtain than system characteristics. Frequency distributions of job run time t_s and resubmittal time T_r, such as are shown in Figure 4, have been measured over extended periods of time for various user populations.[3] Figure 5 shows the result of a mean-value analysis of computer dependent workers, assuming a resubmittal time of two hours and a workday of eight consecutive hours. The aggregate output in

jobs per day is plotted as a function of system turnaround time. The solid lines indicate output per one hundred users who are working at a task concurrency levels 1 and 2, respectively. The dashed line indicates a mean concurrency level of $1\frac{2}{3}$; i.e., two out of three workers have a second task to turn to while waiting, whereas, one out of three does not. Mean levels of this magnitude have been observed in several installations.

Figure 4 Usage characteristics (A) Job run time (B) Resubmittal time

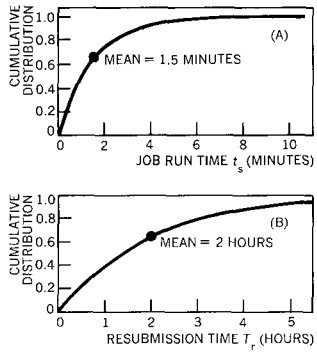


Figure 6 is a normalized version of Figure 5, relative productivity, in which the output (job executed) per unit time is divided by output achievable with no computer-caused delays. Curves for concurrency levels higher than those of Figure 5 are included in Figure 6. The normalized family of curves of Figure 6 indicates *relative productivity,* when the work output is directly proportional to the number of turnarounds per unit time.

System/personnel cost allocation

personnel costs per job

Proceeding now to the determination of personnel costs, we can construct the following formula:

$$\text{Personnel costs/job} = \frac{\text{personnel costs/day}}{\text{jobs executed/day}}$$

$$= \frac{\$200}{\text{job/day}}$$

assuming an eight-hour work day at twenty-five dollars per hour. These costs are plotted in Figure 7, as functions of turnaround time and concurrency level, for resubmittal time of 2 hours.

cost of turnaround time

Figure 7 shows the high cost of the time lost while waiting for the system to return a job. When the system overloading causes lengthy turnaround times, management often attempts to keep personnel busy by assigning more concurrent tasks per worker. This policy has two risks. The quality of the work may suffer because of confusion and memory lapses. Another effect, more clearly observed, is an increasing flow time (task completion time) as task concurrency increases. The magnitude of the flow time impact can be plotted as indicated in Figure 8, for assignments that require one hundred computer turnarounds.

Figure 5 Aggregate job output for a range of job concurrency

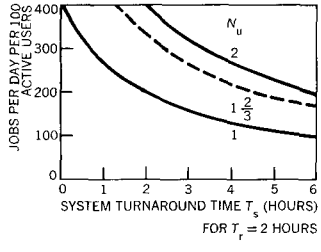
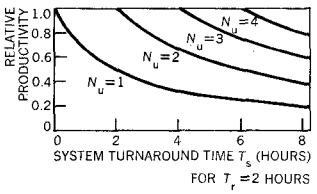


Figure 9 is a synthesis of the information plotted in Figures 6–8, showing the combined system costs plus personnel costs and the flow-time effects of system turnaround time and task concurrency level. By plotting his data in this form, the installation manager has a helpful display of facts necessary for formulating policy and making management decisions of the type mentioned earlier. Before proceeding, however, let us attempt to refine our assumptions so that our model more accurately represents actual working conditions.

Figure 6 Relative productivity of computer-dependent workers



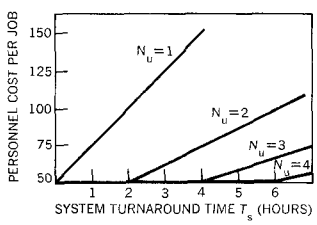
The first refinement takes into account the fact that computer-dependent workers have other things to do, things that require their attention and time outside the analysis-resubmittal-computation loop. For example, certain documentation and housekeeping duties associated with a job can be performed after a job has been submitted, thereby overlapping the human activities and the turnaround time, with the result of shortening the job cycle time. Discretionary activities such as work breaks and lunch hours somewhat relax the requirement for fast turnaround. For the purposes of our analysis, such offline activities are grouped and prorated per job. They are symbolized by T_o, and are represented schematically as shown in Figures 1, 10, and 11.

Figures 10 and 11 illustrate activity analyses of computer-dependent user types C and D, respectively. Examination of such activity charts as Figures 10 and 11 make evident the validity of the general expression for average runs per day as follows:

$$\text{Runs/day} = \frac{\text{Hours on-site/day} \times \text{task concurrency level}}{\text{hours/job cycle}}$$

where the expression for "hours per job cycle" depends on whether the process is computer-limited or human-limited.

Figure 7 Personnel costs



Stated in terms of our model, this general expression takes the following form:

Computer-limited case:

$$\text{Runs/day} = \frac{9 \times N_u}{T_u + T_s} \text{ if } T_s \geq N_u T_o + (N_u - 1)T_u \qquad (1)$$

User-limited case:

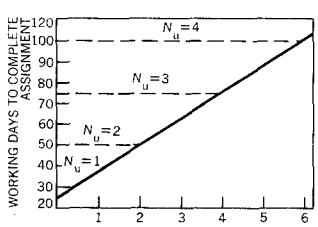
$$\text{Runs/day} = \frac{9 \times N_u}{(T_u + T_o)N_u} = \frac{9}{T_u + T_o}$$

$$\text{if } T_s < N_u T_o + (N_u - 1)T_u \qquad (2)$$

T_u = User analysis-modification time per job
T_o = User off-line activities prorated on a per-job basis

It is assumed that the user spends nine consecutive hours on the site per day.

Figure 8 Effect of system turnaround time on flow time



Application of the Equations 1–2 enables us to develop the presentation of Figure 12, which shows the costs and flow times for a range of turnaround times and concurrency levels. In Figure 12, T_u and T_o are taken as 1.5 and 0.75 hours, respectively. Although this is a useful presentation, it does not clearly reveal resource allocation effects. To obtain this information more conveniently, use the fact that in Figure 12, the system and personnel costs per job and the flow time can be determined for any value of T_s and N_u. These data can be reformatted as shown in

Figure 9 System plus personnel costs

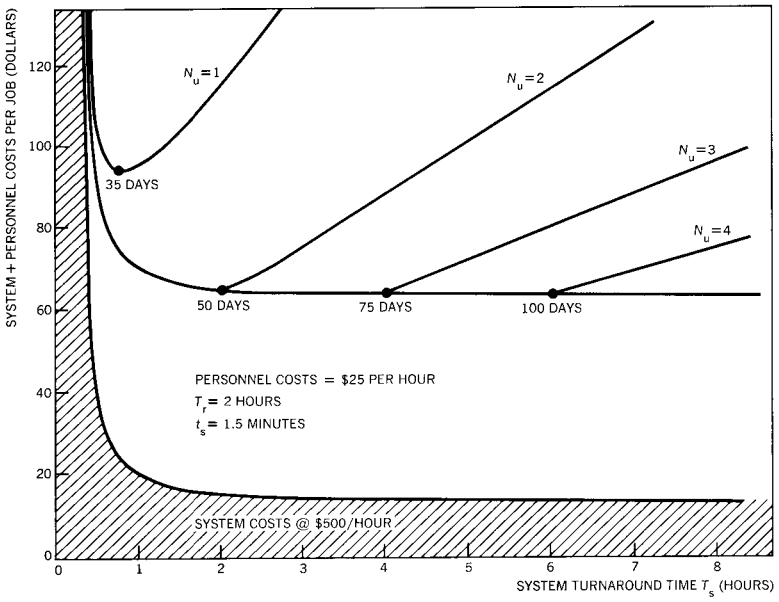


Figure 13, where the flow time and total costs are plotted directly as a function of the allocation ratio of system costs to personnel costs.

Examining the $N_u = 1$ case in Figure 13, we see that costs are minimized for a cost allocation ratio around 0.4. If insufficient resources have been allocated for system (i.e., ratio < 0.4), we see that excessive human waiting time can increase both project flow time and total costs. If excessive resources have been allocated for system (ratio > 0.4), total costs increase because of excessive system idleness. Flow time does not change because the process is absolutely human-limited.

The interpretation for the case of user type D (with $N_u = 2$) is similar, except that the optimal cost allocation ratio is less as a consequence of the relaxation of the need for very fast turnaround. Since we now have computed the cost allocation ratios for user types A, B, C, and D, the ratio for the entire department is simply the weighted average of the type ratios.

The facts are now sufficiently well organized that a summary table of costs and delays, such as Table 1, can be prepared for management.

Figure 10 Workday scenario for user type C

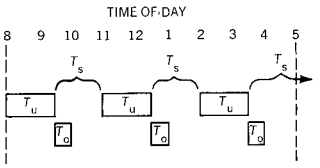
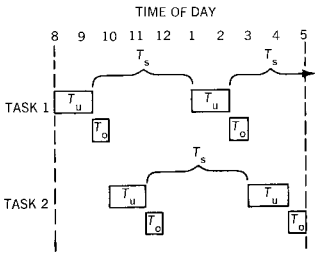


Figure 11 Workday scenario for user type D



summary of costs and delays

The following assumptions have been made in Table 1:

- Jobs per assignment = 100
- System costs = $500 per hour
- Personnel costs = $25 per hour
- Mean job run time = 1.5 minutes

Figure 12 Costs and flow times for a range of turnaround times and concurrency levels

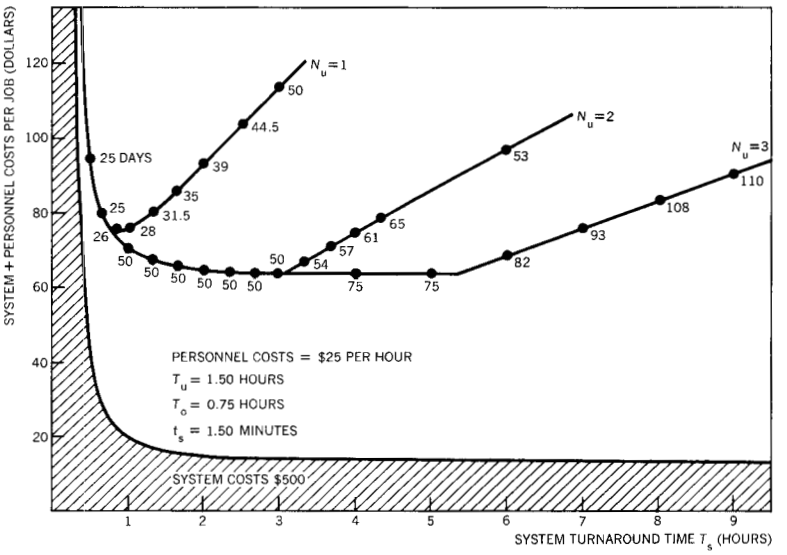


Figure 13 Flow time, cost, and cost allocation ratio
(A) Concurrency level 1
(b) Concurrency level 2

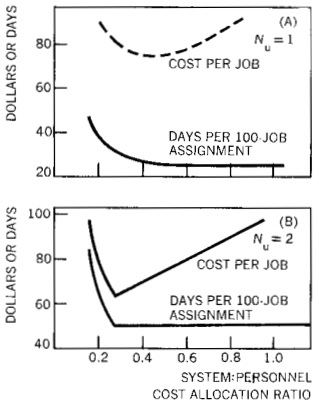
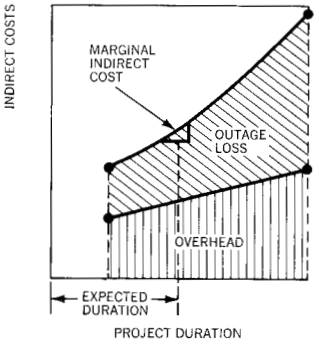


Figure 14 Marginal indirect costs



Note the presence of delay costs, which have been listed here arbitrarily as $50 per day for task duration greater than 20 days. In general a means of determining such delay costs is as follows. Assume that a critical path graph[5,6] (PERT chart) can be constructed for a project, and that we are analyzing a task that is one of the links in the project graph. Then the delay cost of the task in question can be estimated by the following expression:

Flow time cost rate = (probability that the task is on the critical path) × (marginal indirect costs)

The concept of marginal indirect costs is illustrated by Figure 14 and is expressed as the incremental indirect cost rate in dollars per unit time evaluated at the expected completion date of the project.

Priority assignment

We now make use of the model in the assigning of priorities to the jobs, to influence the sequence in which they obtain service. Our objective is to minimize the total cost to the organization, including the cost of time waiting for the return of computed results. Such an objective can be best realized by a scheduling policy that selects jobs for servicing on the basis of relative delay cost rate c (in units of dollars per hour). If the job run times are unknown, the optimal policy is to service jobs in descending order of c.[7,8] If the run times t_S are known, the servicing should be in descending order of the c/t_S ratio.

Table 1 Summary of costs and delays

N_u	*Optimum system turnaround time (hours)*	*System + personnel costs per assignment (dollars)*	*Working days to complete 100-job assignment*	*System + personnel + flowtime costs if delay penalty of $50 assessed for days > 20*
1	0.9	7500	26	7800
2	3.0	6400	50	7900
3	5.25	6400	75	9150

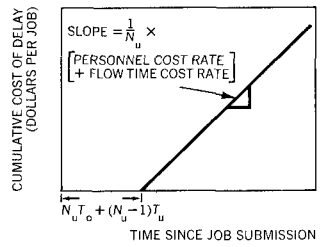
Figure 15 shows the delay cost function in terms of the parameters of the model. The function is piecewise linear, with the delay cost rate being zero for $N_u T_o + (N_u - 1)T_u$ hours. Then the cost function assumes the value:

$$C = \frac{1}{N_u} \times [\text{Personnel cost rate} + \text{flow time cost rate}]$$

$$= \frac{1}{N_u} \times [\text{Personnel cost rate} + (\text{probability that the task is on the critical path}) \times (\text{marginal indirect costs})]$$

A scheme for pricing computer servces based on delay cost functions is described in Reference 9.

Figure 15 Delay cost function



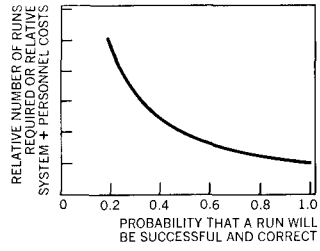
System-user interface costs

Throughout the analysis so far, consideration of the success or failure of a computer run has been neglected. We have assumed that a fixed number of computer runs is required to accomplish a task, an oversimplification that we shall now attempt to correct. A more accurate estimate of the number of runs required to accomplish a computer-dependent task is as follows:

$$\text{Total number of runs required} = \frac{\text{Number of successful and correct runs required}}{\text{Probability that a run will be successful and correct}}$$

Figure 16 is a curve of the relative effect of the probability that a run will be successful and correct on the number of runs required per task or, equivalently, the system plus personnel costs. The question provoked by this curve is: How many resources should be allocated to the effort to improve the probability that a run will be successful and correct? These resources are referred to as "interface costs", and include the following:

Figure 16 Relative effect of the probability that a run will be successful and correct on the number of runs or costs



- Personnel assigned to programming assistance and problem diagnosis.
- Software usability aids such as debugging aids and syntax checkers.
- Interactive facilities for quick fault detection and correction.

To determine the magnitude of justifiable interface costs, one forms a payoff characteristic such as shown in Figure 17A. This characteristic describes the payoff in ease of use, as measured by the probability that a run will be successful and correct for various levels of investment in the system-user interface. As usual, empirical data to construct such a curve are difficult to obtain. In the absence of collected data, it is reasonable to assume that the characteristic has the general shape shown in Figure 18A for the following reasons. If no effort is expended to develop or support a user interface (cost allocation ratio is zero), the probability of using the system successfully is also close to zero. At the other extreme, great investments in developing and maintaining the user interface are expected to result in a very high probability of successful use. Between these extremes, some kinds of diminishing returns are expected, such that the improvement in success probability per unit investment diminishes with larger and larger investments. The exponential function shown in Figure 17A describes a process in which diminishing returns set in uniformly, in the sense that equal increments of investment cause equal percentage reductions in the remaining sources of failure.

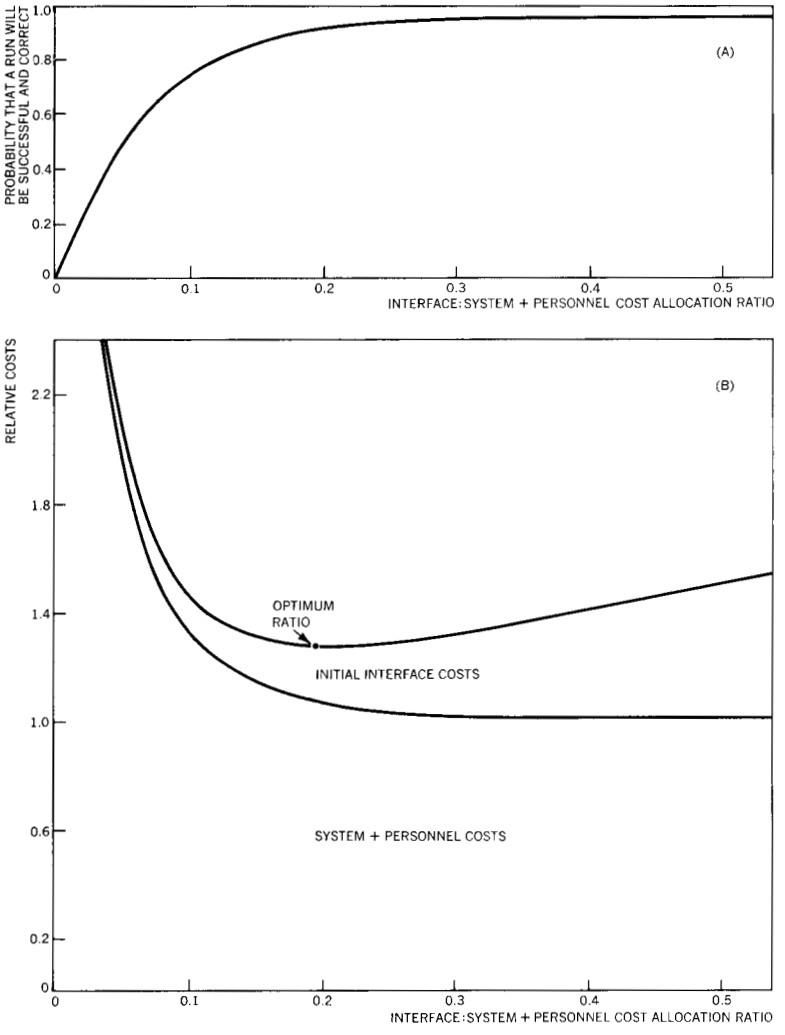
Assuming the payoff curve to be as shown in Figure 17A, we can readily calculate and display the relative costs (including system, personnel, and interface costs) as a function of the cost allocation ratio (interface costs per system and personnel costs), as shown in Figure 17B. Here, relative costs are expressed as follows:

$$\text{Relative costs} = \frac{1 + \text{Cost allocation ratio}}{\text{Probability of a successful and correct run}}$$

It is useful to make a distinction between "initial interface costs" and "continuing interface costs." Initial interface costs are the non-recurring costs that are associated with the introduction of a new system, including such items as the development and documentation of the user interface software and the education of the users. The continuing interface costs include salaries of consultants, maintenance of interface software, and resources used by interactive checkout.

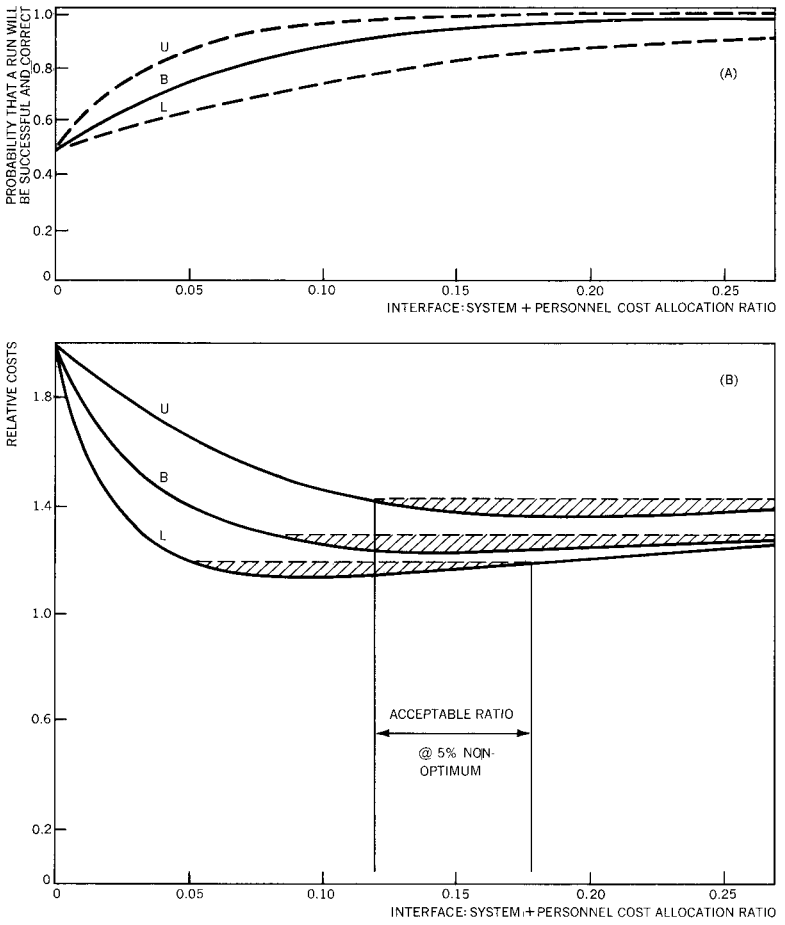
If the accuracy of the data is questionable, a sensitivity analysis may show how sensitive the conclusions are to the probable

Figure 17 Diminishing returns on investment (A) Equal increments of investment yield equal percentage reductions of cost (B) Relative cost as a function of cost allocation ratio



data error. Consider Figure 18A, and suppose that curve B is our best guess for the payoff curve for continuing interface costs. Also suppose that we are confident that the true relationship lies between the upper curve U and the lower curve L. Figure 18B shows the range of relative cost curves that result from such suppositions. The region from each curve in which costs are within five percent of the optimum has been shaded. We can see that the results are relatively insensitive to our data uncertainty in this case, since a resource allocation ratio anywhere between 0.12 and 0.18 results in costs within five percent of optimum for any likely payoff curve.

Figure 18 Sensitivity analysis (A) Payoff (B) Range of relative cost



Concluding remarks

Given the methodology and tools, the major barrier to carrying out productivity studies of the kind discussed in this paper is our ignorance of certain system and usage characteristics. Clearly, the first order of business is to develop measurement techniques so as to provide the required data. A review of this paper makes it evident that the kinds of data needed are the following:

- System congestion characteristic, as illustrated in Figure 2.
- Usage characteristics, t_s, T_r, T_u, T_o, N_u, as illustrated in Figures 1 and 4.
- Project cost-delay-precedence characteristics, as shown in Figure 14.
- Interface investment-payoff characteristics, as shown by Figure 17.

Given the incompleteness of the data that are available, one may legitimately question the value of conducting such an analysis.

On this point, it may not be inappropriate to conclude with some personal observations and those of colleagues. The alternatives to using some kind of quantative analysis are to depend on one's intuitive judgments or simply to stand pat on the policies of the past. In the rapidly changing field of computer usage, seat-of-the-pants and/or stand-pat management have resulted in wasteful inefficiencies. Models, such as the one presented here, can be used in a limited, qualitative fashion to demonstrate the form of the functional relationships and interdependencies. This may be useful as background to help clarify the issues and tradeoffs in the mind of the decision maker. If the model is to be used quantitatively, it must be modified and augmented until it represents the environment under study. This requires a substantial effort for a problem of this complexity.

In all cases, one must remember and respect the assumptions and limitations under which the analysis is conducted. If the accuracy of the data used in the study is questionable, a sensitivity analysis may show how sensitive the conclusions are to the probable data error.

REFERENCES

1. D. N. Streeter, "Cost-benefit evaluation of scientific computing services," *IBM Systems Journal* **11**, No. 3, 219–233 (1972).
2. R. E. Miller, Jr. and J. McCaslin, "Cost-effective matching of task and computing service," *Proceedings of the American Society of Civil Engineers; Journal of the Structural Division*, 329–337 (January 1971).
3. The author is indebted to Mr. Russell Mapes of the McDonnell-Douglas Corporation, Huntington Beach, California, for much of this information as well as the analysis based on such data illustrated in Figures 5 and 6.
4. D. N. Streeter, *The Scientific Process and the Computer*, John Wiley and Sons, Inc., New York, New York (1974).
5. H. Sackman and R. L. Citrenbaum, Editor, *Online Planning*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1972.
6. G. E. Whitehouse, *Systems Analysis and Design Using Network Methods*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1973.
7. M. Greenberger, "The priority problem and computer time sharing," *Management Science* **12**, No. 11, 888–906 (1966).
8. L. Schrage, *Optimal Scheduling Descipline for a Single Machine Under Various Degrees of Information*, Quarterly Report No. 40, Institute for Computer Research, University of Chicago (February 1974).
9. S. B. Ghanem, "Computing center optimization by a pricing priority policy," in this issue.

Reprint Form No. G321-5016