DOCUMENT RESUME

ED 107 287 52 IR 002 033

AUTHOR Rouse, William B.; And Others
TITLE A Mathematical Model of the Illinois Interlibrary Loan Network: Project Report Number 2.
INSTITUTION Illinois Univ., Urbana. Coordinated Science Lab.
SPONS AGENCY Bureau of Libraries and Educational Technology (DHEW/OE), Washington, D.C.
REPORT NO UILU-ENG-75-2209
PUB DATE Mar 75
NOTE 56p.; Not available in hard copy due to marginal legibility of original document

EDRS PRICE MF-$0.76 HC Not Available from EDRS. PLUS POSTAGE
DESCRIPTORS Costs; *Interlibrary Loans; Library Automation; Library Circulation; *Library Networks; Library Research; Library Services; Library Technical Processes; *Mathematical Models; Operations Research; *Simulation; State Libraries; State Programs; Systems Analysis; Union Catalogs
IDENTIFIERS ILLINET; *Illinois Library and Information Network

ABSTRACT

The development of a mathematical model of the Illinois Library and Information Network (ILLINET) is described. Based on queueing network theory, the model predicts the probability of a request being satisfied, the average time from the initiation of a request to the receipt of the desired resources, the costs, and the processing loads. Using a hypothetical network, two sets of operating policies are analyzed: those emphasizing minimum delay and those that maximize the probability of successfully meeting user requests. Cost constraints and value judgements about tradeoffs between delays and the probability of satisfying user requests are considered in the context of network operating policies. The impact of union listings of holdings, automated circulation at the individual libraries, and computer-controlled networks is analyzed. Future plans for network modeling together with the equations used in the network simulation are also presented. (DGC)

REPORT T-16 MARCH, 1975 UILU-ENG 75-2209

COORDINATED SCIENCE LABORATORY

A MATHEMATICAL MODEL OF THE ILLINOIS INTERLIBRARY LOAN NETWORK: REPORT NO. 2

WILLIAM B. ROUSE
JAMES L. DIVILBISS
SANDRA H. ROUSE

UNIVERSITY OF ILLINOIS - URBANA, ILLINOIS

A MATHEMATICAL MODEL OF THE ILLINOIS INTERLIBRARY LOAN NETWORK

Project Report No. 2

Submitted to

Illinois State Library

William B. Rouse

James L. Divilbiss

Sandra H. Rouse

Coordinated Science Laboratory

University of Illinois at Urbana-Champaign

Urbana, Illinois 61801

March 1975

This research was made possible by a grant from the Illinois State Library under the Illinois Program for Title I of the Federal Library Services and Construction Act.

FOREWARD

This is the second in a series of reports resulting from a research grant to the Coordinated Science Laboratory, through the Library Research Center, of the University of Illinois at Urbana-Champaign. The sponsor of the grant is the Illinois State Library under the Illinois Program for Title I of the Federal Library Services and Construction Act.

TABLE OF CONTENTS

I. INTRODUCTION AND SUMMARY

This report summarizes progress on the development of a mathematical model of the Illinois Library and Information Network. The main objective of this project is to produce a mathematical model and associated computer programs for use by the State Library in evaluation and planning of the Illinois network. While this goal is rather specific, we are endeavoring to develop a general understanding of library networks and a general model of their operation. With this point in mind, much of our research has been directed at broad issues that have meaning for most library networks.

In Section II, we discuss the ILLINET model in general qualitative terms of resources, demands, and network operating policies. The model is basically a queueing network which predicts probability of a request being satisfied, average time from initiation of a request to receipt of the desired resource, costs, and processing loads.

Using a hypothetical network, we consider two fairly general classes of network operating policies; those that emphasize minimization of delay and those that emphasize maximization of probability of success. With the ILLINET model, we show how network dispersion and the distribution of resources affects the appropriateness of these two classes of policies. Combining cost constraints and value judgements concerning the tradeoff between average delay and probability of successfully satisfying a request, we briefly discuss choosing a specific network operating policy.

We consider how network performance (probability of success, average delay, and costs) is affected by union listings of holdings, automated circulation at the individual resource libraries, and a computer-

controlled network. Based on the cost reimbursement procedure in Illinois, union listings appear to have significant economic benefit not to mention the improved service possible with such listings.

The effect of processing load on average delay via queueing is considered and shown to have the potential of substantially increasing processing times throughout the network. This increase either costs the requestor in terms of delay or costs the network in terms of increased staffing and/or productivity to maintain acceptable levels of service.

Section III discusses the future plans of our project which include analysis of the Illinois network in light of the data currently being gathered and the impact of possible applications of computer and communication technology.

In Appendix A, the equations currently incorporated in the ILLINET model are derived. Appendix B includes a summary of the ILLINET User's Manual.

In this report, we have performed a rather detailed analysis of a hypothetical networking situation. This was partially motivated by the lack of recent data from the Illinois network. However, the main reason for considering a hypothetical network was to clearly illustrate the numerous policy issues and how various network parameters might affect the resolution of these issues.

II. POLICY ISSUES AND THE ANALYSIS OF A HYPOTHETICAL NETWORK

A. Introduction

Since our last project report, the ILLINET model has been considerably revised and extended. It now incorporates a distributed queueing network that allows for consideration of a rather robust set of network operating policies. To facilitate use of ILLINET as an analysis and design tool, the interactive features of the computer program have been considerably enhanced.

In this section of this report, we want to discuss how ILLINET can be used to analyze a library/information network. We will stress interpretation of model output and discussion of policy issues. The reader interested in detailed assumptions and procedures is referred to the Appendices.

B. The Model

We start with a set of geographically disperse resources and a set of geographically disperse demands. Our goal is to match demands and resources so as to maximize probability of satisfaction (fill rate), minimize average delay in receiving the desired resource, and minimize the cost of providing the service. Combining resources, demands, and a communication protocol, we have a network. We will assume that the characteristics of the resources and demands are relatively static. This leaves the communication protocol or what we term the network operating policy as the means to achieve the goals noted above.

ILLINET is a model that predicts the effects of network operating policy on fill rate, average delay, cost, and processing loads. In essence,

the model is a hierarchical queueing network. It is hierarchical in the sense that different portions of the network can be defined to have different levels of responsibility. Queueing comes into play when we allow the processing time at a point in the network to be a function of the demand on that point.

To be consistent with the terminology used in the Illinois network, the resource points in the network will be called Centers. The points from which demand emanates will be called Systems.* Demand can be categorized according to subject area. We will refer to subject areas as classes.

C. Network Operating Policies

By network operating policy, we mean the prearranged paths of requests through the network. Upon sending a request to a Center, three things may happen. First, it may be satisfied and the desired resource delivered to the requestor. If the request is not satisfied at the initial Center, it may be forwarded to another Center. It is also possible that an unsatisfied request is not forwarded, but is categorized as unfillable and returned to the requestor.

For a given class of requests from a given System, the network operating policy specifies the Center at which the request is initiated into the Center level of the network and the subsequent routing of the request should it fail to be satisfied at that initial Center. The specification of the Center at which the request is initiated we have chosen to term allocation. The choice of the path by which a request proceeds through the

*In Illinois, Systems are the points at which the requests from local libraries are aggregated for transmission to the Centers.

network we term routing. Whenever a request is satisfied, its path terminates and does not continue to follow the specified routing. Thus, a request need not always be processed at every Center specified in its route.

Class j requests that enter the network at Center i are termed type ij requests. Associated with type ij requests is a routing vector r_{ijk} where k refers to the k^{th} Center to which a type ij request is routed if it has not yet been satisfied. The length of the routing vector (the maximum k) is n_{ij} which may take on values of 1, 2, ..., N where N is the number of Centers in the network. The process by which a request proceeds along the route specified by its routing vector is termed referral.

To decide on the routing of a request one must choose r_{ijk} and n_{ij}. One has N choices for the first Center at which a request will be processed. One has N - 1 choices for the second Center, N - 2 choices for the third Center, etc. Since we might choose values of n_{ij} ranging from 1 to N, we find that the total number of possible routing vectors for a type ij request is given by

$$\sum_{i=1}^{N} \frac{N!}{(N-i)!}$$

For example, if N = 4, the total number of possible routing vectors is 64 while N = 10 yields a total number of 9,864,000.

The above results could be applied to each class of requests from each System. If there are L Systems and M classes, then there would be a total of

$$LM \sum_{i=1}^{N} \frac{N!}{(N-i)!}$$

possible policies. As an example, if there are 16 Systems, 8 classes, and 4 Centers, there are 8,192 possible network operating policies. This gives the reader an idea of the complexity of the problem of choosing a policy.

However, we need not specifically consider such a robust set of policies to pursue some of the general policy issues. Instead, we will consider two classes of policies and how each might be appropriate in different situations.

Before we define these policies, consider some of the basic trade-offs involved in choosing a policy. To maximize probability of successfully satisfying a request, the request should be sent to the Center most likely to fill a request in that class. To minimize delay in satisfying a request, the request should be sent to the Center least distant (in delivery time) from the requestor.* However, more often than not, the Center with the greatest strength in a given resource class will not be the Center least distant from the requestor. Thus, we have to trade off delay versus fill rate. In other words, how many additional requests would have to be satisfied to justify a given increase in average delay to all satisfied requestors?

One approach to resolving this issue is to initially send requests to the least distant Center and then route those not satisfied to the Center with the most strength in the desired resource class. The result is that some requestors receive relatively fast service, but the remaining requestors (those not satisfied at the least distant Center) wait longer than they would have if their requests had been initially sent to the Center with the most strength in the resource class. Whether or not this policy is acceptable

*We are, for the moment, assuming that all Centers have approximately equal processing times.

depends on three factors. First, it depends on the difference in strength between the rearmost Center and the strongest Center. Second, it depends on the dispersion of the network. For a network with low dispersion, the strongest Center may not be significantly more distant than the rearmost Center. Finally, it depends on the budget available since requests not satisfied at the rearmost Center and then routed to the strongest Center will incur processing costs at both Centers.

Referrals not only result in increased total processing costs. They also result in a higher processing load on the network. If the staffing of the network is not increased, then increased processing load will result in increased processing time (because of queueing). Thus, to keep processing time within acceptable limits, additional staff and/or new technology is necessary. Whether or not this is feasible depends on whether or not the increased funds generated by the additional processing load are sufficient to cover such investment.

There are also other investment issues. Would a union list of the network's holdings be a reasonable investment? We will show later how such a list could result in substantial savings in network operating costs. What are the effects on network performance of automated circulation systems at the Centers? Such systems would decrease processing time but, depending on the reimbursement policy, may have little effect on network operating costs. Would computerized allocation and routing of requests be of value? It certainly would make service more consistent and, if it could access a union list of network holdings and automated circulation systems at the Centers, could result in an almost instantaneous response to the requestor concerning

whether or not his request will be satisfied. However, it would not materially affect processing time* at the Center where the request was satisfied or delivery time for the desired resource. The basic savings would be the time and cost of referral.

The overall policy issues hinge on the tradeoffs between service and cost and between the two measures of service (fill rate and average delay). In the next section of this report, we will quantitatively consider these tradeoffs by analyzing a hypothetical network. We will consider policies that attempt to minimize delivery time (termed "time" policies) and policies that try to maximize fill rate (termed "strength" policies). Also, we will look at the effects of referral on all three measures of performance.

D. A Hypothetical Network

We have two motivations for considering a hypothetical network instead of an actual network. Our first reason is that there is not yet sufficient and consistent data for the Illinois network to allow a rigorous analysis. This condition will be remedied in the next few months and a detailed analysis of the Illinois network will be the subject of our next project report. The second reason for considering a hypothetical network is that we can construct a network situation that will clearly illustrate the important policy issues and avoid the obscurity brought on by statistical effects and policy irregularities.

*It would not yield improvement in processing time beyond that resulting from a union list and automated circulation.

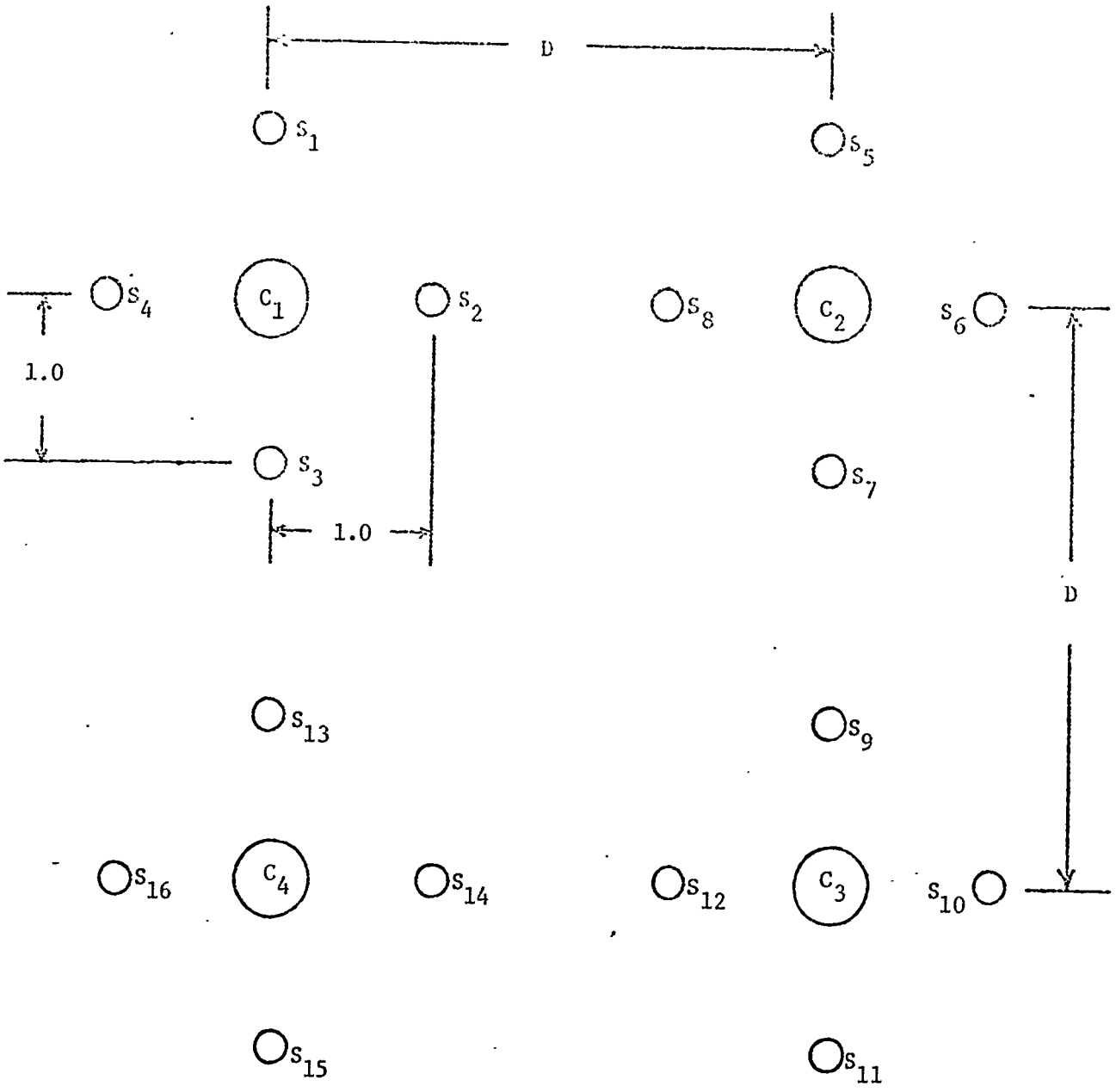
A "map" of our hypothetical network is shown in Figure 1.* There are four resource Centers and sixteen Systems where requests from local libraries are aggregated for transmission to the Centers. The scale of our map is in delivery time units. We assume that the four Centers are on the corners of a square. The square has dimensions of D delivery time units on a side and $\sqrt{2}$ D delivery time units across its diagonals. The distance between each System and the closest Center is one delivery time unit.

D is our measure of the dispersion of the network. D may be proportional to distance if the mode of delivery consumes time in proportion to distance. However, the relationship between distance and time is not always as simple as one might suppose. For example, mail traveling within a city may take longer to reach its destination than mail traveling coast-to-coast. However, the point of this discussion is to note the effects of D and not to consider how D might be measured.

We next want to consider the distribution of resources in the network. For convenience, we will assume that there are eight subject areas or classes and that the i^{th} Center will satisfy a class j request with probability p_{ij}. We will consider two resource distributions: uniform and skew. A uniform distribution is such that all four Centers have equal values of p_{ij}. A skew distribution is such that one Center has a high value of p_{ij} while the other three Centers have low and equal values of p_{ij}. We will refer to the high value of p_{ij} as the "strong" fill rate and to the low value of p_{ij} as the "weak" fill rate.

---

*The description of this hypothetical network and the data values assumed are not meant to reflect the specific operation of the Illinois network.



C = Center
S = System
Scale = Delivery Time Units

A Hypothetical Network

Figure 1

With a uniform resource distribution, we assume that all Centers have strong fill rates in all classes. With a skew resource distribution, we assume that Center 1 is strong in classes 1 and 2 while being weak in classes 3 through 8; Center 2 is strong in classes 3 and 4 while being weak in the other classes; etc.

We will assume that each System receives and transmits an average of 1,000 requests per year in each class. With sixteen Systems and eight classes, this results in a total of 128,000 requests per year.

If a request enters a Center without a call number, it is "searched" to see if the Center owns the desired item. We will assume that the cost and/or reimbursement for a search is $1.00. If a request is filled, we will assume the cost and/or reimbursement to be $2.00.

In Appendix A, we discuss how each Center is composed of six processes: in-process request, search, verify, obtain, out-process item, and forward request. Initially, we have assumed that these processes consume an average of 0.125, 0.500, 0.500, 0.500, 0.500, and 0.125 time units respectively. Later in this report, we will consider the effects of processing time being a function of demand.

We have now completely defined our hypothetical network. Its symmetry and regularity are highly idealized. The advantage of this idealization is that we now can consider the effects of network structure and policies unencumbered by the differences in demand and policies among the various Centers and Systems.

E. Analysis and Results

Before discussing our analysis, let us summarize the variables we are considering.

1. Allocation policies: time and strength
2. Routing: n_{ij} = 1, 2, 3, 4
3. Dispersion: D = 3, 6, 9
4. Resource distribution: uniform and skew
5. Strong/weak fill rates: 0.5/0.3, 0.7/0.1

We are assuming a fixed average demand of 128,000 requests per year and search/fill reimbursement costs of $1.00/2.00.

For our first analysis, we assumed that all requests enter the network without call numbers and that the only reason requests are not filled is because the desired resource is not owned. Further, we assumed the processing time is not affected by the level of demand (i.e., no queueing). We will relax these assumptions and discuss them in more detail in later analyses.

Tables I through IV are compilations of the output of ILLINET for the assorted combinations of the variables noted above. It should be stressed at this point that the specifics of the hypothetical network or, for that matter, the specifics of the Illinois network are defined by the input data for the ILLINET model and are not "hard wired" into the program. Thus, analysis of different networks is accomplished by changing the data and not the program.

The performance measures in Tables I and II are total satisfied, total cost, unit cost, and marginal unit cost. Since the total input to the network is 128,000 requests per year, the total satisfied divided by that number is the fill rate. Total cost includes all processing and includes both satisfied and unsatisfied requests. Unit cost is the total satisfied divided by the total cost. The marginal unit cost of referral is the additional number of requests satisfied by the referral divided by the additional cost incurred by the referral.

With a uniform resource distribution, the time and strength policies are identical since the nearest Center is always equal in strength to any other Center. The appropriate policy in this situation is to initialize a request at the nearest Center and, if not satisfied there, to refer it to the next nearest Center, etc. until the funding is exhausted. The unit cost and marginal unit cost of referral are identical and independent of the number of referrals. Except for the possibility of exceeding budget limitations, the only negative aspects of referral in a network with a uniform resource distribution is the possible increase in processing time due to increased demand on each Center.

It is unlikely that any actual network would have a uniform distribution of resources. We present it merely as a point of reference with which to compare other results.

If the distribution of resources is skew and we employ a time allocation policy, then a request has a probability of $(N - 1)/N$ of being initialized at a weak Center. If it is not satisfied at this initial weak

	Strength Distribution	Allocation Policy	Number of Referrals 0	1	2	3
Total Satisfied	Uniform	Time	64,000	96,000	112,000	120,000
	Skew	Time	44,800	74,240	93,504	106,048
	Skew	Strength	64,000	83,200	96,640	106,048
Total Cost	Uniform	Time	256,000	384,000	448,000	480,000
	Skew	Time	217,600	359,680	451,968	511,552
	Skew	Strength	256,000	358,400	430,080	480,256
Unit Cost	Uniform	Time	4.00	4.00	4.00	4.00
	Skew	Time	4.86	4.84	4.83	4.82
	Skew	Strength	4.00	4.31	4.45	4.53
Marginal Unit Cost	Uniform	Time	4.00	4.00	4.00	4.00
	Skew	Time	4.86	4.83	4.79	4.75
	Skew	Strength	4.00	5.33	5.33	5.33

Number of Requests Satisfied, Total Costs, and Unit Costs

(Strong Fill Rate = 0.5, Weak Fill Rate = 0.3)

Table I

	Strength Distribution	Allocation Policy	Number of Referrals 0	1	2	3
Total Satisfied	Uniform	Time	89,600	116,480	124,544	126,963
	Skew	Time	32,000	58,880	81,344	100,006
	Skew	Strength	89,600	93,440	96,896	100,006
Total Cost	Uniform	Time	307,200	399,360	427,008	435,302
	Skew	Time	192,000	341,760	455,808	539,789
	Skew	Strength	307,200	353,280	394,752	432,077
Unit Cost	Uniform	Time	3.43	3.43	3.43	3.43
	Skew	Time	6.00	5.80	5.60	5.40
	Skew	Strength	3.43	3.78	4.07	4.32
Marginal Unit Cost	Uniform	Time	3.43	3.43	3.43	3.43
	Skew	Time	6.00	5.57	5.08	4.50
	Skew	Strength	3.43	12.00	12.00	12.00

Number of Requests Satisfied, Total Costs, and Unit Costs

(Strong Fill Rate = 0.7, Weak Fill Rate = 0.1)

Table II

Center and is referred, there is a probability of (N - 2)/(N - 1) that it will be referred to another weak Center. If we continue this enumeration, we find that the probability of a request having been processed by a strong Center increases with the number of referrals. Thus, the unit cost and marginal unit cost decrease with number of referrals. The amount by which these costs decrease is directly related to the difference between the strong and weak fill rates. A large difference in these fill rates results in a large decrease while a small enough difference results in negligible decrease (I and II).*

If we employ a strength allocation policy with a skew distribution of resources, referral always results in a request going to a weak Center since the strength policy, by definition, initiates a request at the Center strongest in the class of interest. Thus, unit cost and marginal unit cost increase with the number of referrals. If all of the weak Centers in each class have equal weak fill rates, marginal unit cost will reach a constant after one referral. The increase in these costs is greatest when the difference between strong and weak fill rates is large (I and II).

Thus, these two classes of policies have opposite effects on unit cost and marginal unit cost. The effects on total satisfied and total cost immediately agree with intuition. Comparing the time and strength policies for a skew resource distribution, we see that the strength policy results in more satisfied requests and initially the total cost is higher. However, as the number of referrals increases, the difference in total satisfied between strength and time policies decreases and the total cost eventually changes in favor of the strength policy.

---

*Numerals in parentheses are references to tables.

If the four measures discussed above were the only considerations, we would choose the strength policy and use as many referrals as allowed within total cost, unit cost, and/or marginal unit cost constraints. However, we also want to consider the average total time or delay in satisfying a request. The average delays for three values of our dispersion parameter D are summarized in Tables III and IV.

We see that the average delay with the strength policy is initially much higher than with the time policy and that this difference increases markedly as the dispersion of the network increases. However, as the number of referrals increases the average delays with the two policies become more similar. In fact, if the difference between strong and weak fill rates is large enough, the strength policy can result in an average delay less than that resulting with the time policy. (See Table IV for the skew resource distribution.) This effect is seen as the number of referrals increases and is due to the fact that the time policy results in requests staying in the network longer.

Thus, we have seen the general effects of different allocation and routing policies on our measures of performance. How do we resolve the trade-offs and choose a specific policy? This is a large question which we do not intend to address in general in this report. However, we can consider a simplistic approach that illustrates how a policy might be chosen.

Let us assume that our budget is constrained to $400,000 and that we are not concerned with unit costs or marginal unit costs. Further, assume that we have made a rather arbitrary value judgement that the strength policy

Network Dispersion, D	Strength Distribution	Allocation Policy	Number of Referrals 0	1	2	3
3	Uniform	Time	2.63	3.57	3.95	4.23
	Skew	Time	2.63	3.75	4.26	4.72
	Skew	Strength	4.49	4.46	4.85	5.02
6	Uniform	Time	2.63	4.56	5.21	5.69
	Skew	Time	2.63	4.93	5.80	6.58
	Skew	Strength	7.02	7.20	7.38	7.55
9	Uniform	Time	2.63	5.55	6.49	7.17
	Skew	Time	2.63	6.12	7.36	8.45
	Skew	Strength	9.58	9.75	9.93	10.10

Average Total Time to Satisfy a Request

(Strong Fill Rate = 0.5, Weak Fill Rate = 0.3)

Table III

Network Dispersion, D	Strength Distribution	Allocation Policy	Number of Referrals 0	1	2	3
3	Uniform	Time	2.63	3.28	3.47	3.56
	Skew	Time	2.63	3.94	4.56	5.23
	Skew	Strength	4.49	4.52	4.57	4.64
6	Uniform	Time	2.63	3.96	4.30	4.45
	Skew	Time	2.63	5.31	6.37	7.47
	Skew	Strength	7.02	7.06	7.11	7.18
9	Uniform	Time	2.63	4.65	5.14	5.36
	Skew	Time	2.63	6.69	8.19	9.74
	Skew	Strength	9.58	9.61	9.66	9.73

Average Total Time to Satisfy a Request

(Strong Fill Rate = 0.7, Weak Fill Rate = 0.1)

Table IV

will be used if the resulting per cent increase in average delay (over that we would obtain with the time policy) does not exceed one half of the resulting percent increase in total satisfied.

Considering the skew resource distribution with strong and weak fill rates of 0.5 and 0.3, respectively, we find that the time policy with one referral and the strength policy with one referral satisfy our budget constraint. The strength policy results in 12% more satisfied requests (I) (than the time policy) and increased average delays of 24%, 46%, and 59% for D equals 3.0, 6.0, and 9.0, respectively (III). All of these per cent increases in average delay exceed one half the per cent increase in total satisfied and thus, the strength policy is rejected and the time policy accepted.

Considering the skew resource distribution with strong and weak fill rates of 0.7 and 0.1, respectively, we find that the time policy with one referral and the strength policy with two referrals satisfy the budget constraint. The strength policy results in 65% more satisfied requests (II) and, for D equal 3.0, this is more than twice the percent increase in average delay (IV). Thus, for D equal 3.0, the strength policy is preferred. However, for larger values of D, the per cent increase in average delay exceeds one half the per cent increase in requests satisfied and therefore, the time policy is preferred.

Thus, we see that cost constraints, network dispersion, and the parameters of the resource distribution affect the choosing of an appropriate network operating policy. Even for somewhat similar networks, different

value judgements by the managers of each network may result in very different policies being appropriate for each network. ILLINET cannot make the value judgements but it can provide the information necessary for the formulation of policy decisions consistent with the network manager's value judgements.

Our next analysis of the hypothetical network considered the effects of having a union list of network holdings such that all requests enter the Center level of the network with call number specified. In such a situation, all requests are for owned resources and the only reason for a request not being satisfied is unavailability. Thus, we adjusted the probability of a resource being available to compensate for the difference in probability of a resource being owned and the Center fill rate. As in our first analysis, we assumed that processing time was unaffected by the level of demand.

Assuming a complete union list and that all requests are for material contained in this list, we find that the resulting performance measures significantly differ from those in Tables I through IV in only one component: cost. With a complete union list, no requests would ever be "searched" since a Center would know that any request sent to it is for a resource owned by the Center. Thus, the total cost is simply the cost of filling a request ($2.00) times the number of requests satisfied. The total costs resulting for the various policies discussed above are summarized in Table V. Note that cost savings can range up to almost $340,000 per year.

Strong/Weak Fill Rates	Strength Distribution	Allocation Policy	Number of Referrals 0	1	2	3
0.5/0.3	Uniform	Time	128,000	192,000	224,000	240,000
	Skew	Time	89,600	148,480	187,008	212,096
	Skew	Strength	128,000	166,400	193,280	212,096
0.7/0.1	Uniform	Time	179,200	232,960	249,088	253,926
	Skew	Time	64,000	117,760	162,688	200,012
	Skew	Strength	179,200	186,880	193,792	200,012

Total Costs With a Complete Union List

Table V

While in actuality, one would probably not save as much as indicated by the figures in Table V, this analysis does point out that savings might justify the production of a union list of network holdings. As a by-product, each individual library in the network would benefit from having a list of its own holdings for use by local patrons, schools, etc.

A less impressive effect of the union list is the savings in time "searching" the request. As the network becomes more disperse, this difference becomes less significant. There is a potential here for savings in staff costs but we have not analyzed the situation to determine the exact effects. Actually, these savings would affect the individual library but not the network since the network would no longer be paying for searching regardless of the individual library's staffing decisions.

Our last analysis concerns the effects of queueing in the network. The specific queueing models incorporated in ILLINET are discussed in Appendix A. The basic idea of queueing is as follows. If a process has a fixed average rate at which it can serve requests (termed the service rate with dimensions requests per unit time) then, as the rate at which requests arrive increases, longer waiting lines will form and the total average time for a request to be processed (including waiting) will increase. To facilitate comparison between queueing and no queueing, we chose the service rates for the various queues in the network so that the time policy (with no referrals) in a network with a uniform resource distribution and strong fill rate of 0.7 would yield identical average processing times whether or not there was queueing. Then we increased the number of referrals which resulted in increased demand at each Center (more requests per unit time). Without commensurate increases in the rate at which requests could be serviced (by adding staff or new technology), the average delay in satisfying a request increased as shown in Table VI. The delays in this table include the time spent by requests waiting in queues, the time spent in actually being serviced, and the time to deliver the requested resource.

The increase in average delay with queueing is not a function of dispersion and thus, the <u>per cent</u> effect of queueing will decrease as dispersion increases. This will be true regardless of the queueing model we adopt.

This analysis with queueing has served to illustrate how queueing can increase average delay and/or require increased staffing or new technology.

If we were to go back and consider again the tradcoff between total requests satisfied and average delay, we would find that referral would result in increasing delay not only because of network dispersion but also because of the increased processing load on each Center.

The queueing analysis could also be used to predict the additional staffing required to maintain a given level of service. While we have not yet added this option to ILLINET, it will be included in the future. Any analysis that does not consider queueing effects is likely to ignore one of the more important determinants of network performance.

Queueing	Network Dispersion, D	Number of Referrals 0	1	2	3
No	3	2.63	3.28	3.47	3.56
Yes	3	2.63	4.36	5.28	5.70
No	6	2.63	3.96	4.30	4.45
Yes	6	2.63	5.04	6.11	6.59
No	9	2.63	4.65	5.14	5.36
Yes	9	2.63	5.73	6.95	7.49

Average Delays With and Without Queueing

Table VI

F. Conclusions

We have considered two fairly general classes of network operating policies; those that emphasize minimization of delay and those that emphasize maximization of fill rate. We have shown how network dispersion and the

distribution of resources affects the appropriateness of these two classes of policies. Combining cost constraints and value judgements concerning the tradeoff between delay and fill rate, we briefly considered choosing a specific network operating policy.

We discussed network union lists, automated circulation, and computer-controlled networks. Considering the costs and benefits of each of these alternatives, a network union list might result in substantial savings in annual operating costs and thereby justify its production. Automated circulation systems would have to be justified by the benefits to the individual library as the benefit to the network would be unlikely to justify the investment. The benefits of a computer-controlled network are numerous and this interesting alternative is discussed in more detail in a later report.

When demand increases, waiting lines or queues form. Such queueing was shown to substantially increase processing times throughout the network. This increase in processing time is one of the costs of referral. It either costs in service to the requestor or costs in increased staffing to maintain acceptable levels of service.

III. IN THE FUTURE

Our current efforts are in three areas. First, we are continually trying to enhance the interactive features of ILLINET. One very important aspect of this effort is policy formulation. As noted earlier, for any reasonably complex network, there are thousands of possible network operating policies. To enable the user of ILLINET to consider a large number of policies, extensive on-line editing features have been developed that allow the user to construct one policy from another and thus, avoid having to start from scratch for each policy. This saves considerable time, but further improvements are desirable and being explored.

A second area of research concerns using simulation to test the queueing assumptions noted in our first report. The routing vector idea has eliminated the need for assuming a lack of knowledge of a request's past routing. Currently, we are investigating the probability distributions of request interarrival times and service times. This should enable us to define the range of applicability of ILLINET.

A third effort is aimed at investigating the impact of mini-computer circulation systems at various points in a network. Also, the possibility of a central switching computer for a whole network is being studied to determine its feasibility in terms of cost and impact on service.

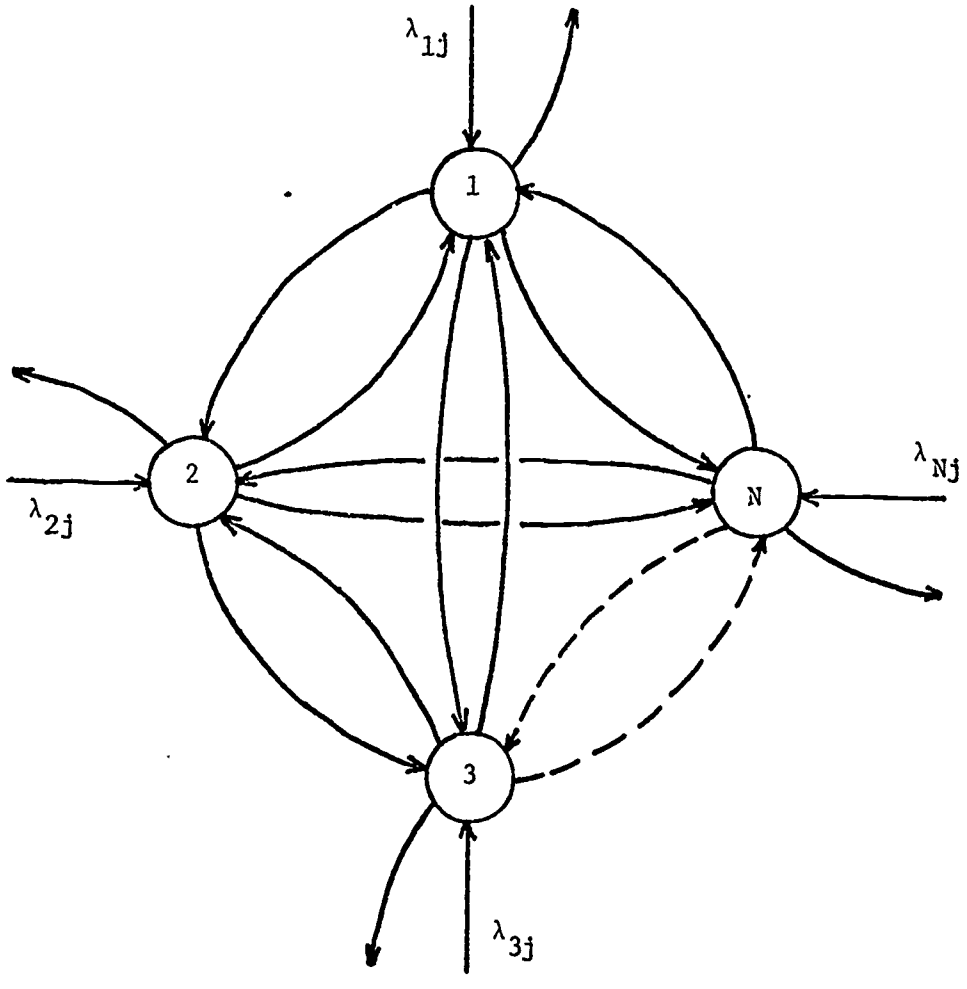
Our next project report will mainly consider analysis of the Illinois network in light of the data that is currently being collected. This report will discuss how data can be appropriately formatted and aggregated for the model as well as the analysis.

APPENDIX A

DERIVATION OF MODEL EQUATIONS

This version of ILLINET assumes a single-level distributed network. We will first consider the equations appropriate to such a network structure and then derive equations for the flow within each processing point.

The network is illustrated below. N processing points or nodes and M subject classifications are assumed. Class j requests enter the network at node i at an average rates of λ_{ij} (requests/unit time). Associated with each request is a "routing vector" r_{ijk} where $k = 1,2,\ldots,n_{ij}$ and $n_{ij} \leq N$.



The routing vector specifies the nodes that the request will successively

visit until it is either satisfied or reaches the last node specified by r_{ijk}. To illustrate, since a type ij request is by definition initiated at node i, $r_{ij1} = i$.

Defining p_{ij} as the probability of satisfying a request in class j at node i and P_{ijk} as the probability of satisfying a type ij request at the k^{th} node in its routing vector, we find*

$$P_{ijk} = \prod_{\ell=1}^{k-1} [\, 1-p(r_{ij\ell},j)]\; p\,(r_{ijk},j) \tag{1}$$

where p_{ij} is assumed to be independent of a request's processing before entering node i. The probability of a type ij request with routing vector r_{ijk} being satisfied is given by

$$P_{ij} = \sum_{k=1}^{n_{ij}} P_{ijk} \tag{2a}$$

And, the average probability of satisfying a request at the node where it entered the network is given by

$$P_i = \sum_{i=1}^{N} \lambda_{ij}\, p_{ij} \Big/ \sum_{i=1}^{N} \lambda_{ij} \tag{2b}$$

while the cumulative probability of satisfying a class j request is given by

$$P_j = \sum_{j=1}^{M} \lambda_{ij}\, P_{ij} \Big/ \sum_{j=1}^{M} \lambda_{ij} \tag{2c}$$

*Subscripts are sometimes bracketed for clarity. However, the meaning does not change. For example p_{ij} and p(ij) are equivalent. Also, for $k = 1$, $\prod_{\ell=1}^{k-1}(\quad) \triangleq 1$ and $\sum_{\ell=1}^{k-1}(\quad) \triangleq 0$.

and the overall probability of satisfying a request is given by

$$P = \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij} P_{ij} \Big/ \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij} \qquad (2d)$$

We should note here that the subscript i in equations (2) indicates the node at which the request entered the network and not necessarily the node at which it was satisfied.

To determine the average time from when the request enters the network until the requested document or information is delivered, define $\hat{\omega}_{ij}$ and $\tilde{\omega}_{ij}$ as the average processing times for class j requests satisfied and not satisfied at node i, respectively. Also, define $t_{i_1 i_2 j}$ as the average time to deliver the class j document or information when the request entered the network at node i_1 and was satisfied at node i_2. $t_{i_1 i_2 j}$ includes only delivery time and not processing time. With these definitions, we find that the average processing plus delivery time for a type ij request, satisfied at the k^{th} node in its routing vector, is given by

$$W_{ijk} = \sum_{\ell=1}^{k-1} \tilde{\omega}\,(r_{ij\ell}, j) + \hat{\omega}\,(r_{ijk}, j) + t\,(i, r_{ijk}, j) \qquad (3)$$

Defining D_{ij} as the expected value of the total time from initiation of the request into the network until delivery of the requested document or information,

$$D_{ij} = \sum_{k=1}^{n_{ij}} P_{ijk} W_{ijk} \Big/ P_{ij} \,. \qquad (4a)$$

The expected total time to satisfy a request entering the network at node i is given by

$$D_i = \sum_{j=1}^{M} \lambda_{ij} D_{ij} \Big/ \sum_{j=1}^{M} \lambda_{ij} \tag{4b}$$

while the expected total time to satisfy a class j request is given by

$$D_j = \sum_{i=1}^{N} \lambda_{ij} D_{ij} \Big/ \sum_{i=1}^{N} \lambda_{ij} \tag{4c}$$

and the expected total time to satisfy a request is given by

$$D = \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij} D_{ij} \Big/ \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij} \tag{4d}$$

To determine $t_{i_1 i_2 j}$, we need to know the source of requests entering the network at node i. Define λ_{ijk}, k = 1,2,...,L, as the rate at which class j requests from source k enter the network at node i. Further let t_{ik} be the average time to deliver a document or information from node i to source k. Then,

$$t_{i_1 i_2 j} = \sum_{k=1}^{L} \lambda_{i_1 jk} t_{i_2 k} \Big/ \lambda_{i_1 j} \tag{5}$$

Next, we want to determine the processing load on each node in the network. Each node must process not only λ_{ij} but also requests referred to it from other nodes. Define Λ_{ij} as the total class j processing load on node i. This is easily calculated recursively by intializing $\Lambda_{ij} = 0$ for all i, j and then using

$$\Lambda(r_{ijk},j) = \Lambda(r_{ijk},j) + \prod_{\ell=1}^{k-1} [1-p(r_{ij\ell},j)] \lambda_{ij} \tag{6a}$$

for $i = 1,2,\ldots,N$; $j = 1,2,\ldots,M$; $k = 1,2,\ldots,n_{ij}$. The total processing load on node i is given by

$$\Lambda_{i.} = \sum_{j=i}^{M} \Lambda(i,j) \tag{6b}$$

while the total class j processing load is given by

$$\Lambda_{.j} = \sum_{i=1}^{N} \Lambda(i,j) \tag{6c}$$

and the total network processing load is given by

$$\Lambda = \sum_{i=1}^{N} \sum_{j=1}^{M} \Lambda(i,j). \tag{6d}$$

Also of interest is the satisfied processing load. Defining $\hat{\Lambda}_{ij}$ as the average rate of type ij requests that are satisfied,

$$\hat{\Lambda}_{ij} = p_{ij} \Lambda_{ij} \tag{7a}$$

The average rate of satisfied requests entering the network at node i is given by

$$\hat{\Lambda}_{i.} = \sum_{j=1}^{M} \hat{\Lambda}_{ij} \tag{7b}$$

while the average rate of satisfied requests for class j requests is given by

$$\hat{\Lambda}_{.j} = \sum_{i=1}^{N} \hat{\Lambda}_{ij} \tag{7c}$$

And the total average rate of satisfied requests is given by

$$\hat{\Lambda} = \sum_{i=1}^{N} \sum_{j=1}^{M} \hat{\Lambda}_{ij} \tag{7d}$$

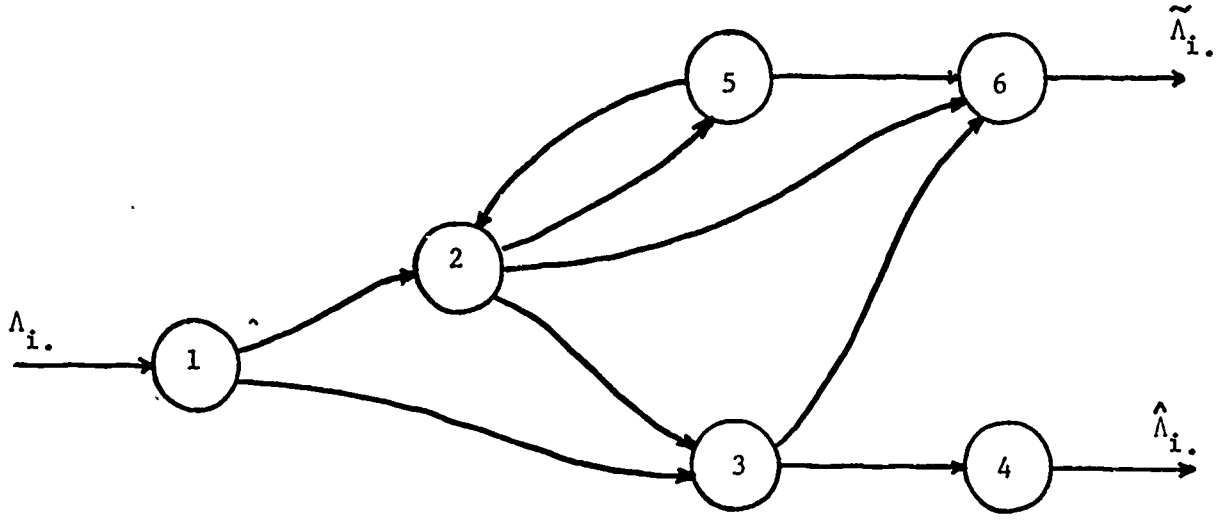
Defining λ as the total rate of requests entering the network

$$\lambda = \sum_{i=1}^{N} \sum_{j=1}^{M} \lambda_{ij} \tag{8}$$

we should note that Λ is usually much greater than λ since requests are often processed at more than one node. $\hat{\Lambda}$ can be no greater than λ since the network cannot satisfy any more requests than are input.

Thus, λ is the network input rate, $\hat{\Lambda}$ is the network output rate, and Λ is the network processing load. P is the probability that a request is satisfied while D is the average time required from initiation of the request into the network until the requester receives the desired document or information.

Each processing node is, in itself, a network. The model adopted is shown below. We will term the processing nodes in this model



as subnodes. Subnode 1 represents in-processing of requests. Subnode 2 represents the search of the catalog to determine if the material requested is owned. Subnode 3 represents the obtaining of the material. Subnode 4 represents the preparation and out-processing of the material. Subnode 5 represents verification of the request in the sense of checking the correctness of the information supplied with the request. Subnode 6 represents the out-processing of unsatisfied requests. The output of subnode 6 is $\tilde{\Lambda}_{i.}$ which equals $\Lambda_{i.}$ minus $\hat{\Lambda}_{i.}$.

After a type ij request completes processing at subnode 1, it proceeds to subnode 3 with probability c_{ij} and to subnode 2 with probability $1 - c_{ij}$, where c_{ij} represents the probability that a request enters supplied with the call number of the desired material.

After processing at subnode 3, a type ij request proceeds to subnode 4 with probability a_{ij} and to subnode 6 with probability $1 - a_{ij}$ where a_{ij} represents the probability that the desired material is owned but not available.

After processing at subnode 2, the request may proceed to subnode 3 if the desired material is owned (with probability o_{ij}). Or, it may proceed to subnode 6 if the desired material is not owned (with probability $1 - o_{ij}$). Or, if the request is unverified (information not checked for correctness, with probability $1 - v_{ij}$), the request might be forwarded to subnode 6 with probability f_{ij}. Otherwise, the unverified request (with probability v_{ij}) proceeds to subnode 5 (with probability $1-f_{ij}$).

After processing at subnode 5, a successfully verified request (with probability s_{ij}) is sent back to subnode 2. Otherwise, the request proceeds to subnode 6.

With these various probabilities defined, we now want to determine the processing load on each of the subnodes. Let Λ_{ijk} be the average rate of class j requests entering the k^{th} subnode of node i. A moderate amount of bookkeeping yields

$$\Lambda_{ij1} = \Lambda_{ij}, \tag{9a}$$

$$\Lambda_{ij2} = [1 + s_{ij}(1 - f_{ij})(1 - v_{ij})] (1 - c_{ij}) \Lambda_{ij}, \tag{9b}$$

$$\Lambda_{ij3} = (c_{ij} + o_{ij} [v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})][1 - c_{ij}]) \Lambda_{ij}, \tag{9c}$$

$$\Lambda_{ij4} = a_{ij}(c_{ij} + o_{ij}[v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})][1 - c_{ij}])\Lambda_{ij}, \tag{9d}$$

$$\Lambda_{ij5} = (1 - f_{ij})(1 - v_{ij})(1 - c_{ij}) \Lambda_{ij}, \tag{9e}$$

$$\Lambda_{ij6} = (1 - a_{ij} o_{ij}[v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})])(1 - c_{ij})\Lambda_{ij}. \tag{9f}$$

Summing over classes yields

$$\Lambda_{i.k} = \sum_{j=1}^{M} \Lambda_{ijk} \tag{10}$$

where $\Lambda_{i.1}$, $\Lambda_{i.4}$, and $\Lambda_{i.6}$ equal $\Lambda_{i.}$, $\hat{\Lambda}_{i.}$, and $\tilde{\Lambda}_{i.}$, respectively.

Now we want to consider the processing time required to pass through node i. Define ω_{ik} as the average time to be processed at the k^{th} subnode of node i. From a queueing standpoint, ω_{ik} is related to $\Lambda_{i.k}$. However, we will ignore this relationship until later in our discussion. Defining $\hat{\omega}_{ij}$ as the average delay experienced by a class j

request being satisfied at node i, we find that

$$\hat{\omega}_{ij} = \sum_{k=1}^{6} \alpha_{ijk}\, \omega_{ijk} \qquad (11)$$

$$\alpha_{ij1} = 1,$$

$$\alpha_{ij2} = (a_{ij}\, o_{ij}[v_{ij} + 2s_{ij}(1 - f_{ij})(1 - v_{ij})][1 - c_{ij}])/\alpha,$$

$$\alpha_{ij3} = 1,$$

$$\alpha_{ij4} = 1,$$

$$\alpha_{ij5} = a_{ij}\, o_{ij}\, s_{ij}(1 - f_{ij})(1 - v_{ij})(1 - c_{ij})/\alpha,$$

$$\alpha_{ij6} = 0,$$

$$\alpha = a_{ij}(c_{ij} + o_{ij}[v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})][1 - c_{ij}]).$$

Similarly, define $\tilde{\omega}_{ij}$ as the average delay experienced by a class j request as it passes through node i unsatisfied. This quantity is given by

$$\tilde{\omega}_{ij} = \sum_{k=1}^{6} \beta_{ijk}\omega_{ik} \qquad (12)$$

where

$$\beta_{ij1} = 1,$$

$$\beta_{ij2} = ([(1 - o_{ij}) + o_{ij}(1 - a_{ij})][v_{ij} + 2s_{ij}(1 - f_{ij})(1 - v_{ij})] + [(1 - s_{ij})(1 - f_{ij}) + f_{ij}][1 - v_{ij}])(1 - c_{ij})/\beta,$$

$$\beta_{ij3} = (1 - a_{ij})(c_{ij} + o_{ij}[v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})][1 - c_{ij}])/\beta,$$

$$\beta_{ij4} = 0,$$

$$\beta_{ij5} = (s_{ij}[(1 - o_{ij}) + o_{ij}(1 - a_{ij})] + [1 - s_{ij}])(1 - f_{ij})(1 - v_{ij})(1 - c_{ij})/\beta,$$

$$\beta_{ij6} = 1.$$

$$\beta = (1 - a_{ij})c_{ij} + ([v_{ij} + s_{ij}(1 - f_{ij})(1 - v_{ij})][(1 - o_{ij}) + o_{ij}(1 - a_{ij})] + [(1 - s_{ij})(1 - f_{ij}) + f_{ij}][1 - v_{ij}])(1 - c_{ij})$$

All that remains in our derivation, is the determination of ω_{ik} for use in equations 11 and 12. Each of the subnodes in a processing node can be modeled as a queueing system as follows.

Requests flow into the k^{th} subnode of the i^{th} node with an average rate of $\Lambda_{i \cdot k}$. These requests often arrive in batches. They queue up until a staff member (called the server takes all of the requests in the queue and services them as a batch. The server processes each request in the batch at an average rate of μ_{ik}. Upon completion of servicing the batch, the server puts the requests in the queues of the appropriate succeeding subnodes and returns to his queue to pick up the next batch.

Requests experience delays in two ways. First, while waiting in the various queues, delay is experienced. Second, while in servicing, a request not only experiences the time required for its own servicing, but also must endure the time required to process every other member of the batch. Only then are all the requests passed on to appropriate succeeding subnodes.

The average total delay experienced ω_{ik} is related to the probability distributions of which $\Lambda_{i\cdot k}$ and μ_{ik} are parameters. If $\Lambda_{i\cdot k}$ increases and ω_{ik} is to be maintained, μ_{ik} must be increased either by increased staffing or productivity. If μ_{ik} cannot be increased, ω_{ik} will increase as $\Lambda_{i\cdot k}$ increases. Thus, an increased allocation of demand to a given network node results in increased service costs and/or increased processing delays.

We would like an equation that relates ω_{ik} to $\Lambda_{i\cdot k}$ and μ_{ik}. Several approximations have been investigated but comparisons of these relationships with simulation results have caused us to reject these formulations and continue our search for an appropriate analytical expression. The difficulty we are encountering is basically in finding a simple expression which does not require extensive analytical gymnastics every time $\Lambda_{i\cdot k}$ is changed.

The simulation results are, in themselves, interesting and of use. A FORTRAN queueing simulation was developed and tested by simulating queueing situations where the analytical solution was known. These tests

produced statistically acceptable comparisons and thus validated the simulation. Simulating the queueing situation described above and fitting a function to the results, we found that ω_{ik} can be approximately predicted using

$$\omega_{ik} \approx \frac{B_{ik}}{\mu_{ik}} \cdot \frac{1}{(1 - \rho_{ik})^{1.15}} \qquad (13)$$

where

$$\rho_{ik} = \Lambda_{i \cdot k}/\mu_{ik} < 1,$$

B_{ik} = incoming constant batch size.

The simulation incorporated exponential interarrival and service times and constant incoming batch size. However, if we continue to use simulation, different interarrival and service time distributions may be employed as well as allowing for varying incoming batch size.

Equation 13 can also be used to predict the μ_{ik} necessary to achieve a given ω_{ik}. This would be of use for predicting the staffing and/or productivity increase necessary to meet anticipated changes in demand.

Summarizing our queueing results, we feel that more effort is needed in this direction perhaps along the line of predicting ω_{ik} for a general incoming batch size distribution and general service time distribution. We plan to pursue these items and present them in later reports.

APPENDIX B

SUMMARY OF ILLINET USER'S MANUAL

This Appendix includes the first section of the four sections of the ILLINET User's Manual. This first section of the Manual summarizes the use of the ILLINET model while the remaining three sections have detailed descriptions of model usage. Because of its length, we have not included the whole Manual. However, those readers interested in the whole Manual, will be furnished a copy upon their request.

ILLINET

Interlibrary Loan and Information Network Model

USER'S MANUAL

William B. Rouse

Sandra H. Rouse

Version No. 2

March 1975

ILLINET USER'S MANUAL

1.0 Introduction

ILLINET is an interactive computer model for the design and evaluation of hierarchical library/information networks. The acronym ILLINET has two meanings. For this computer program, ILLINET stands for Interlibrary Loan and Information Network Model. Also, ILLINET is the name of the Illinois Library and Information Network under whose support this computer model is being developed. However, we want to stress the fact that the ILLINET model has a general structure that is not specific to any particular network and can be used to design and evaluate a variety of network situations. The specific network of interest is defined by input data to the model as will be discussed in later sections of this User's Manual.

ILLINET models the following situation. We have a set of geographically disperse resources and a set of geographically disperse demands. Combining these resources, demands, and a communication protocol, we have a network. The communication protocol is termed the network operating policy. ILLINET predicts the effects of network operating policy on the following:

(1) probability of a request for a resource being satisfied (fill rate)
(2) average time from initiation of a request until the requestor receives the desired resource (delay)
(3) network operating costs
(4) network processing load.

To be consistent with the terminology used in the Illinois network, the resource points in the network are called Centers. The points from which demand emanates are called Systems. Demand is categorized into classes which can represent subject areas, requestor status, etc.

This User's Manual is composed of three main sections in addition to this introductory section. The first section deals with the input data for ILLINET that characterizes resources, demands, processing times, etc. Also, in this section, we discuss editing of data files. The second section deals with the formulation and editing of network operating policies. The final section deals with running the network model and interpreting the results.

1.1 How to Start and End ILLINET

We assume the user is familiar with the log-in and log-off procedures of the specific computer he is accessing. This section of the manual will describe how the user begins interacting with ILLINET and how to exit from ILLINET.

Once the user has properly logged-in to his specific computer, he will receive a signal that indicates he may access any files available to him. Again it is the user's responsibility to be familiar with the specific computer he is using. In the case of the DEC System 10, the user is notified by a period. This period is sometimes referred to as the monitor level. Other computers signal the user in a similar way although they may not use the period or specifically refer to the monitor level.

At the monitor level, the user is able to execute the ILLINET program. He enters RU ILLINET which in fact means run ILLINET. ILLINET will immediately respond with a '*'. The following illustrates how the user begins execution of ILLINET and ILLINET'S first response:

```
.RU ILLINET
*
```

Throughout this manual we will present examples of interaction with ILLINET. The underlined segments indicate user input. All non-underlined response is generated by ILLINET. (In the above example, the period is not generated by ILLINET but by the specific computer facility being accessed by the user).

The '*' is a signal that ILLINET is waiting for information to be entered by the user. After receiving the '*', the user may enter the following options:

(1) a "simple" carriage return
(2) a "procedure" name followed by a carriage return.

For purposes of this manual, we will define a "simple" carriage return as the user inputting only a carriage return. (For a given line of input, a simple carriage return is never preceded by other user input). Entering a simple carriage return at the '*' level of ILLINET will end execution of the program. The user will subsequently be returned to the monitor level (signaled by a period in the case of the DEC System 10). At this point the user may log-off or access any other files available to him. The user should take note that all input must be terminated with a carriage return (sometimes referred to as the send key).

The word "procedure" has a special meaning in ILLINET and will be defined in the following section. The various procedures available are summarized in section 1.2.1 and described in detail later in this manual. The user will have a better understanding of how to successfully utilize these procedures if he reads the entire manual before sitting at the terminal.

1.2 Interaction with ILLINET

To interact with ILLINET, the user must provide "procedures" and "specifications". A "procedure" invokes a particular operation such as editing, input/output,etc., while a "specification" indicates the data file or elements on which the chosen operation is to be performed.

Generally we can describe procedures as user initiated. Initiating a particular procedure will cause ILLINET to ask the user to specify some information or data. These specifications are procedure dependent. In other words, the user chooses the procedure while ILLINET asks for the necessary specifications to perform the desired procedure.

The procedures described in this manual have the following general functions:

(1) access, editing, and display of the input data files characterizing the network's demand level, fill rate, delivery time, and reimbursement schedule.
(2) creation, access, and editing of network operating policies which define the allocation and routing of requests to various resource nodes in the network.
(3) operating upon specified data with the model to predict network performance.

These functions are described in the following four subsections.

1.2.1 Procedures

The purpose of procedures in ILLINET can briefly be described in terms of their operations. At the end of this subsection there is a list of procedure names and operations.

Procedures are input by the user after receiving the '*' from ILLINET. ILLINET will only understand a procedure if it is entered at the '*' level. During a session with ILLINET the user will probably utilize most of the procedures. Since he can enter only one procedure at a time, he must direct his interaction with ILLINET in such a way that he returns to '*' level. The general strategy for escaping from a procedure and reaching the '*' level is to enter a simple carriage return in response to every specification asked for by ILLINET. (For the definition of simple carriage return see section 1.1.) Upon returning to the '*' level, the user can then select another procedure. Full explanation of all procedures are given in this manual. The specification requirements and the way in which one can exit from a procedure are explained in the sections corresponding with the procedures.

Any procedure name followed by an 'H' indicates to ILLINET that the user needs help. ILLINET responds by referring the user to the section or subsection of this manual which explains the procedure entered by the user. For example, suppose the user wants to know the names of the data files which he can view at the terminal. Since this is related to the DSDAT procedure the user would enter:

```
*DSDATH
SEE SECTION 2.1 OF THE USER'S MANUAL
*
```

The following list briefly describes the procedures. For more detailed explanation see the sections in this manual that correspond with the procedures.

Name	Operation
DATA ORIENTED PROCEDURES:	
PSDAT	Displays data file at the terminal or output all data files as hardcopy on the lineprinter.
EDDAT	Makes changes in elements of the data files.
SVDAT	Stores data files (on disk) which have usually been edited. Edited versions of data files are thus available to the user during a later session.
RDDAT	Makes data files available (from disk) to the model. Used to restore the version of the data file as it existed before an edited version.
POLICY-ORIENTED PROCEDURES:	
MAKE	Creation of network policies.
LIST	Listing at the terminal of existing policies by identifying number and label.
DELETE	Deletes policies as specified by the user.
COPY	Duplicates policies as specified by the user.
RELPL	Changes policy label as specified by the user.
EDPOL	Changes in allocation or routing of the network policies as specified by the user.
SVPOL	Stores network policies (on disk) created by the user. Access to previously created policies is possible during a later session.
RDPOL	Makes available (from disk) to the user previously stored network policies.

MODEL-ORIENTED PROCEDURES:

RUN	Executes the model for specified policies, without the queueing option.
RUNQ	Executes the model for specified policies, with the queueing option.

1.2.2 Specifications

A specification is information that ILLINET requires to perform the operation indicated by the user's choice of a procedure. A single question mark indicates that ILLINET is requesting a specification and waiting for user input. Specifications may be generally grouped as:

(1) data file name
(2) data file element (numerical value)
(3) center abbreviation
(4) system abbreviation
(5) class or subject abbreviation
(6) table name
(7) policy number
(8) policy label
(9) allocation vector for a policy
(10) routing vector for a policy.

In the following example ILLINET asks the user to specify a data file name :

FILE ? DEMAND

This specification may have been requested after the user entered the DSDAT procedure. Detailed explanation of specification options is found in each section of this manual corresponding with a procedure.

Typographical errors and misspellings are checked by ILLINET. The user is given a list of abbreviations or names acceptable for the particular specification query, and asked for input again. For example:

```
CENTER ? CT5
? ? CENTERS: CT1,CT2,CT3,CT4
CENTER ?
```

The double question mark indicates ILLINET does not understand the last input and a list of acceptable input options will follow. The user is repeatedly asked for input until an acceptable specification is entered.

Note that a simple carriage return is an acceptable response and will cause ILLINET to exit from the specification question. In some cases, this will result in return to the '*' level or, in other cases, to a more general specification level.

1.2.3 Data and Policies

Data and network policies are the quantitative input to the model. Procedures and specifications are the communication tools which cause operations to be performed upon the data and policies.

Before the model is executed, data files and network policies must be available. Data files are made available to the model automatically. The user is not required to define which data files are available for the model. However, if the user has utilized the EDDAT procedure at some time, it is his responsibility to know which version of the data files the model is using. See section 2.2 for an explanation of the EDDAT procedure.

Network policies provided to the model must be input by the user. Policies can be input using the RDPOL procedure to access previously stored policies (from disk). Or, the MAKE procedure can be used to construct new policies. Often, a new policy can be formed by editing

an old policy using the EDPOL procedure. For an explanation of network policies see section 3.0.

1.2.4 Running the Model

While most of our previous discussion has focused on preparation and editing of data files and policies, the heart of ILLINET is the network model. Basically, the model is a distributed queueing network. It is distributed in the sense that any node can communicate a request to any other node. Queueing comes into play when we choose the procedure that allows processing time at a node to be a function of the demand on that node (because higher demand results in longer waiting lines).

Running the model involves only the choice between two procedures. The RUN procedure is used to predict network performance in the absence of queueing. The user may select this procedure because he wants to compare performance with and without queueing to see the magnitude of the queueing effects. Or, the user may select the RUN procedure because he has insufficient data to consider queueing. The RUN procedure is discussed in section 4.1.

The RUNQ procedure is used when one wants to consider queueing effects. It requires a little more data than the RUN procedure (see discussion of SRVTIM file in section 2.1.3) but is a more realistic representation of a library/information network. The RUNQ procedure will eventually be programmed to predict the increased staffing necessary to keep processing time within acceptable limits in the face of increased demand. However, this version of RUNQ is not yet

available. The RUNQ procedure is discussed in section 4.2.