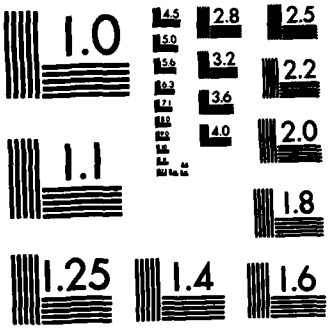
AD-A123 302 OPTIMAL FILE ALLOCATION PROBLEMS FOR DISTRIBUTED DATA BASES IN UNRELIABLE..(U) MASSACHUSETTS INST OF TECH CAMBRIDGE LAB FOR INFORMATION AND D.. M MA ET AL. 1/1

UNCLASSIFIED DEC 82 LIDS-P-1265 N00014-77-C-0532 F/G 9/2 NL



December, 1982

LIDS-P-1265

OPTIMAL FILE ALLOCATION PROBLEMS FOR DISTRIBUTED DATA BASES IN UNRELIABLE COMPUTER NETWORKS*

by

Moses Ma
Michael Athans
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

ABSTRACT

This paper deals with the problem of optimally locating files, and their optimum number of redundant copies in a vulnerable communication network. It is assumed that each node and link of the communication network can fail independently. The optimization problem maximizes the probability that a commander can access the subset of files that he needs while minimizing the network-wide costs related to storage, query and update communication costs. The problem reduces to a linear zero-one integer programming one; several theorems that reduce its complexity of solution are presented.

1.1 INTRODUCTION

This paper focuses on the problem of optimal redundant file allocation for a very vulnerable distributed data base system. This file allocation is different than previous file allocation problems because it considers the following new items:

1. Vulnerability of the nodes and links due to enemy actions, e.g. jamming
2. The importance of the users
3. The importance of particular files to particular users.

First we shall discuss the motivation for the problem of optimal file allocation in a vulnerable environment. Second we shall explain the problem and its constraints. Third we shall discuss the possible trade-off in costs. Fourth a brief literature survey will be presented. Fifth the actual formulation will be explained. Sixth the various theorems that have been developed for this formulation will be stated and explained in words; however, we do not include the theorem proofs. Seventh the conclusions and suggestions for further research will be presented.

1.2 MOTIVATION

The motivation behind our problem is in the C^3 (Command, Control, and Communications) context. In this context we are considering a Naval Battle Group (BG) composed of aircraft carriers, cruisers, destroyers, aircraft, etc. The BG gathers information through its organic sensors. This information must somehow be stored and maintained to the utmost correctness because the BG must coordinate its actions. The system may be considered as a distributed data base system. The ships and planes can be considered as nodes, and the communication channels between the ships and planes can be considered as links. The data can be considered as files stored in the computers of the ships and planes.

Considering the BG as a set of nodes, links and data files, we have defined for ourselves a distributed data base network. Since in warfare, ships can be destroyed and communication links jammed, our network is vulnerable. Therefore we must consider how to maintain a consistent and complete data base.

If we also consider the individual warfare commanders and the data files they need, the problem becomes more complex. We can also rank the importance of each commander and the importance of each data file to each commander and include this in our optimization problem.

1.3 PROBLEM

The problem is therefore as follows-we are given the following:

1. A set of M data files
2. A set of N nodes to store the data files
3. The probabilities of any node of link being destroyed from which the probabilities of any particular commander at one node can access any particular file at another node.
4. A set of L commanders.
5. The costs of assigning a particular file to any particular node.
6. The query rates for any particular file emanating from any particular node. The query rate is the rate at which files are requested.
7. The update rates for any particular file emanating from any particular node. The update rate is the rate at which files are updated (changed).

We desire the following:

1. To locate single or multiple copies of

*This research was supported by the Office of Naval Research under contract ONR/N00014-77-C-0532 (NR 041-519).

the M files at the N nodes such that the files will be accessible to the commanders who need the files.
2. To locate the files at nodes that will provide the least amount of cost. The cost can be related query communication costs, update communication costs and file storage costs.

1.4 TRADE-OFF OF COSTS

There is a trade-off in costs if we consider the following costs:

1. query communication costs
2. update communication costs
3. storage costs
4. the cost to the BG if a particular commander does not have access to the particular file he desires.

To minimize any one of the four costs we can do the following:

1. To reduce the query communication costs, we can store more redundant copies of each file so that each query can find its file with less communication cost. This is true since we shall assume that each query goes to the nearest node containing the file.
2. To reduce the update communication costs, we can store fewer redundant copies of each file so that each update will update fewer files and incur less communication costs. This is true since we assume each update goes to all the nodes containing the file.
3. To reduce the storage costs, we can store fewer redundant copies of each file so the cost will decrease.
4. To reduce the cost of non-accessibility of particular files to particular commanders, we can store more redundant copies of that particular file so that it has a higher probability of being able to be accessed by the particular commander.

The bottom line is: we cannot increase and decrease the number of redundant copies of a particular file. We would like to find the optimal number of redundant copies for all the files and where to store them.

2. LITERATURE SURVEY

The file allocation problem was first investigated by Chu [1]; a global optimization was considered, consisting in obtaining the minimum overall operating costs subject to two kinds of constraints; first, the expected time to access each file had to be less than a given delay, and secondly the amount of storage needed at each computer had not to exceed the available storage capacity. The number of copies of each file was assumed to be fixed. A generalized model was defined, in which storage and transmission costs were associated to file allocations; channel queues were modeled in order to introduce the constraint on the delay. The resulting linearized integer program was characterized by a very great number of variables even for application of limited dimensions; its solution was extremely hard from a conceptual viewpoint.

Casey [2],[7] considered the problem of allocating single files separately, but the number of copies of each file was not assumed to be fixed. Communication costs and storage costs of allocations were analyzed in order to determine the optimal set of nodes on which the file was to be allocated. The difference between retrieval and update transactions was stressed; while retrieval transactions are routed to only one copy of the file, update transactions are routed to all the copies, in order to preserve consistency of redundant information. Under the assumption of taking equal cost rates for retrieval and updates, theorems were given for limiting the number of replicated copies of the file on the basis of the update/query ratio; obviously the convenience of taking replicated copies decreases while the update/query ratio increases. Although the file allocation problem was analyzed for each file separately, Eswaran [3] proved that Casey's formulation was NP complete and, therefore, suggested to investigate heuristic approaches.

Morgan and Levin [4], examined both the allocation of files and transactions within a generalized, ARPA-like network. They adopted the user's viewpoint, assuming to be under the jurisdiction of a network management providing services at the market price. Because of this characterization of application environment, storage capacity constraints were not included; the provision of sufficient storage was considered a task of the network management. Therefore, by introducing some other simplifying assumption, the authors demonstrated that the multiple file allocation problem could be decomposed into independant (single) file allocation problems; they also developed an heuristic solution technique.

Finally two contributions to the file allocation problem have been very recently presented. Ramamoorthy and Wah [5] analyzed a relational Distributed Data Base; they observed that the general approach of query processing optimization consists in the minimization of communication costs. These communication costs are mostly due to data moves which are necessary for providing the logical correlation, expressed by the query, between files stored on different nodes. A logical operation which is particularly critical is the join operation between remote files; a join between two files can be performed only if the two files are co-located at the same node. Therefore the authors developed a model in which redundant files are introduced in order to avoid distributed joins, on the basis of the frequency of queries.

3. NOTATION

b_i = the available memory size of the i^{th} computer

x_{ij} = the file j is stored at node i

N_i = node i

$\Gamma(N_i)$ = set of nodes that are directly linked to node i

$\Gamma^*(N_i) = \Gamma\Gamma\ldots\Gamma(N_i)$

$R(N_i)$ = set of nodes that are accessible to node i by a path

$R(N_i) = \Gamma(N_i) \cup \Gamma^2(N_i) \cup \ldots$

α_i = the importance of commander i

β_{ji} = the value of file j to commander i

$P_{ij}(I)$ = the probability that file j is accessible to the commander at node i given an assignment I

$$A_i(j) \triangleq \begin{cases} 1 & \text{if } \exists k \text{ s.t. } j \text{ at } N_k \in R(N_i) \\ 0 & \text{if } \exists k \text{ s.t. } j \text{ at } N_k \in R(N_i) \\ 0 & \text{if } \forall k \text{ s.t. } j \text{ at } N_k, N_k \text{ destroyed} \end{cases}$$

r_j = the number of redundant copies of file j stored in the system

λ_{ji} = the volume of query traffic emanating from node j for file l

ψ_{ji} = the volume of update traffic emanating from node j for file l

d_{jk} = the cost of a unit of communication from node j to node k

σ_{kj} = the cost of locating a copy of a file j at the k^{th} node

T_{ij} = the maximum allowable query traffic time of the j^{th} file to the i^{th} node

a_{ijk} = the expected time for the i^{th} node to query the j^{th} file from the k^{th} node

I_ℓ = the set of node indexes representing a given assignment of file ℓ.

4. FORMULATION

Since $x_{i\ell}$ is a zero-one variable, the sum over all nodes i must be equal to the number of redundant copies of file ℓ. Therefore we have:

$$\sum_{i=1}^{N} x_{i\ell} = r_\ell \tag{4.1}$$

We define

$$I_\ell(R(N_i)) = \beta_{\ell i} A_i(\ell) \tag{4.2}$$

which denotes the accessibility of file ℓ to the commander at node i weighted by the importance of file ℓ to the commander at node i.

The initial formulation is as follows:

$$\min_{I_\ell} \sum_{\ell=1}^{M} = \min_{I_\ell} \sum_{\ell=1}^{M} \left[\sum_{j=1}^{N} \sum_{k=1}^{N} \psi_{j\ell} d_{jk} x_{\ell k} + \sum_{j=1}^{N} \min_{k \in I_\ell} \lambda_{j\ell} d_{jk} + \sum_{k=1}^{N} \sigma_{k\ell} x_{\ell k} - E\left[\sum_{k=1}^{L} \alpha_i I_\ell(R(N_i)) \right] \right] \tag{4.3}$$

such that

$$x_{11} + x_{12} + \ldots + x_{1M} \leq b_1;$$

$$x_{21} + x_{22} + \ldots + x_{2M} \leq b_2;$$

$$\vdots$$

$$x_{N1} + x_{N2} + \ldots + x_{NM} \leq b_N .$$

and

$$(1-x_{i1}) x_{k1} a_{i1k} \leq T_{i1};$$

$$(1-x_{i2}) x_{k2} a_{i2k} \leq T_{i2};$$

$$\vdots$$

$$(1-x_{iM}) x_{kM} a_{iMk} \leq T_{iM} .$$

$$x \geq 0$$

where the first term in the minimization corresponds to the cost of updating file ℓ at node k which was requested by node j, where each node k is a node that has file ℓ. The second term denotes the cost of querying file j at node k which was requested by node j, where node k is the closest node containing file ℓ. The third term represents the cost of storing file ℓ at node k. The last term denotes the cost associated with the expected accessibility of file ℓ to the commander at node i weighted by the importance of commander i.

The first set of constraints state that the number of files stored at any node must be less than the capacity at each node. The second constraint states that the expected time to retrieve a query is less than a certain threshold quantity. The last constraint states that all the zero-one variables are nonnegative.

If we now examine the last term in the minimization, we can simplify the expression. The expected value may be brought inside the summation. Since the importance of the commander i and the importance of file ℓ to commander i are not probabilistic, we can simply take the expected value of the accessibility. However, the expected value of the accessibility is simply the probability that commander i can access file ℓ given the allocation of redundant copies of file ℓ in the network. We have:

$$E\left[\sum_{i=1}^{L} \alpha_i I_\ell(R(N_i)) \right] = E\left[\sum_{i=1}^{L} \alpha_i \beta_{\ell i} A_i(\ell) \right];$$

$$= \sum_{i=1}^{L} \alpha_i \beta_{\ell i} E\, A_i(\ell) \;;$$

$$\sum_{i=1}^{L} \alpha_i \beta_{\ell i} P_{i\ell}(I_\ell) \;. \tag{4.4}$$

where

$$P_{i\ell}(I_\ell) = \begin{cases} 1 & \text{if } \Pr(\exists k \text{ s.t. } \ell \text{ at } N_k \in R(N_i)) \\ 0 & \text{if } \Pr(\exists k \text{ x.t. } \ell \text{ at } N_k \in R(N_i)) \\ 0 & \text{if } \Pr(\forall k \text{ s.t. } \ell \text{ at } N_k, N_k \text{ destroyed}) \end{cases} \tag{4.5}$$

Where $P_{i\ell}(I_\ell)$ for one file ℓ is by definition:

$$P_{i\ell}(I_\ell) = \sum_{j=1}^{N} \prod_{\substack{k=1 \\ k\neq j}}^{N} (1-x_k)x_j P_{ij}$$

$$+ \sum_{h=1}^{N} \sum_{\substack{j=1 \\ j>k}}^{N} \prod_{\substack{k=1 \\ k\neq j \\ k\neq h}}^{N} (1-x_k)x_j x_h [P_{ij}+(1-P_{ij})P_{ih}]..$$

$$+ \sum_{a=1}^{N} \sum_{\substack{b=1 \\ b>a}}^{N} \cdots \sum_{\substack{j=1 \\ j>h \\ \vdots \\ j>b \\ j>a}}^{N} \prod_{\substack{k=1 \\ k\neq j \\ k\neq h \\ \vdots \\ k\neq b \\ k\neq a}}^{N} \qquad (4.6)$$

$$x_k [P_{ij}+(1-P_{ij})P_{ih}+(1-P_{ij})(1-P_{ih})P_{ig}\ldots]$$

which simplifies to the following:

$$= \sum_{j=1}^{N} x_j P_{ij} - \ldots$$

$$+ (-1)^{N+1} \sum_{j=1}^{N} \sum_{\substack{k=1 \\ k>j}}^{N} \prod_{\substack{m=1 \\ m\neq j \\ m\neq k}}^{N} x_m P_{im}$$

$$+ (-1)^{N} \sum_{j=1}^{N} \prod_{\substack{m=1 \\ m\neq j}}^{N} x_m P_{im} + (-1)^{N+1} \prod_{m=1}^{N} x_m P_{im} \qquad (4.7)$$

where P_{ij} is the probability of accessibility between nodes i and j.

Substituting this back into the initial formulation we now have:

$$\min_{I_\ell} \sum_{\ell=1}^{M} C(I_\ell) =$$

$$\min_{I_\ell} \sum_{\ell=1}^{M} \left[\sum_{j=1}^{N} \sum_{k=1}^{N} \psi_{j\ell} d_{jk} x_{\ell k} + \min_{k\in I_\ell} \lambda_{j\ell} d_{jk} + \right.$$

$$\left. + \sum_{k=1}^{N} \sigma_{k\ell} x_{\ell k} - \sum_{i=1}^{L} \alpha_i \beta_{\ell i} P_{i\ell}(I_\ell) \right] \qquad (4.8)$$

such that

$$\sum_{j=1}^{M} x_{ij} \leq b_i \; ; \quad 1\leq i\leq N$$

$$(1-x_{ij})x_{kj} a_{ijk} \leq T_{ij} ; \quad 1\leq j\leq M \quad i\neq k$$

$$x\geq 0$$

We know that

$$\sum_{k=1}^{N} (\text{anything}) x_{\ell k} = \sum_{k\in I_\ell} (\text{anything}) \qquad (4.9)$$

So substituting into the previous formulation we have:

$$\min_{I_\ell} \sum_{\ell=1}^{M} C(I_\ell) =$$

$$\min_{I_\ell} \sum_{\ell=1}^{M} \left[\sum_{j=1}^{N} \sum_{k\in I_\ell} \psi_{j\ell} d_{jk} + \min_{k\in I_\ell} \lambda_{j\ell} d_{jk} + \right.$$

$$\left. + \sum_{k\in I_\ell} \sigma_{k\ell} - \sum_{i=1}^{L} \alpha_i \beta_{\ell i} P_{i\ell}(I_\ell) \right] \qquad (4.10)$$

such that

$$\sum_{j=1}^{M} x_{ij} \leq b_i \; ; \quad 1\leq i\leq N$$

$$(1-x_{ij})x_{kj} a_{ijk} \leq T_{ij} ; \quad 1\leq j\leq M \quad i\neq k$$

If we remove the constraints, we are minimizing over disjoint sets, so we have

$$\min_{I_\ell} \sum_{\ell=1}^{M} C(I_\ell) = \sum_{\ell=1}^{M} \min_{I_\ell} C(I_\ell) \qquad (4.11)$$

Lets now try to $\min C(I_\ell)$ for a particular ℓ. The following of theorems will set bounds on how to allocate the files and determine when not to allocate files.

5. THEOREMS

First lets look at allocating one file ℓ optimally by placing redundant copies at different nodes, so without loss of genrality let $I=I_\ell$.

Theorem I: If $\psi=\rho\lambda_j$ for j=1,2,...,n then an r-node assignment cannot be less costly than the optimal one-node assignment if

1)
$$\rho \geq \frac{1}{r-1} , \qquad (5.1)$$

and

2)
$$\gamma \leq \frac{(\rho r-\ell-1)(1+\rho)}{\rho r} \sum_{j=1}^{N} \lambda_j d_{j1} + \frac{(1+\rho)}{r} \sum_{k=2}^{r} \delta_k$$
$$+ \frac{(1+\rho)}{\ell r} \sum_{j=1}^{N} \min_k \lambda_j d_{jk} + \frac{(r-1)}{r} \sigma_1 + \frac{1}{\rho r} \sum_{k=2}^{r} \sigma_k$$

Theorem I states that if we have allocated r-1 copies of a file and the two inequalities hold, then by allocating the rth file, our total cost will not be less than just allocating one file optimally. This will allow us to reduce the possible solution space in which the integer program must search. Now we can exclude all allocations with more than r-1 files from possible file allocations before execution of the integer program.

Theorem II: If for some integer $r\leq n$,

1)
$$\rho > \frac{1}{r-1} , \qquad (5.3)$$

and

2)

$$\rho < \frac{(\rho r-\ell-1)(1+\rho)}{\ell r} \sum_{j=1}^{N} \lambda_j d_{j1} + \frac{(1+\rho)}{r} \sum_{k=2}^{r} \delta_k$$

$$\frac{(1+\rho)}{\rho r} \sum_{j=1}^{N} \min_k \lambda_j d_{jk} + \frac{(r-1)}{r} \sigma_1 + \frac{1}{\rho r} \sum_{k=2}^{r} \sigma_k \qquad (5.4)$$

then any r-node file assignment is more costly than an optimal one-node assignment.

Theorem II states that if we have allocated r-1 copies of a file and certain conditions hold, then by allocating the rth file, our total cost will be more than allocating one file optimally. This will allow us to reduce the possible solution space in which the integer program must search. Now we can exclude all allocations with more than r-1 files from possible file allocations before execution of the integer program.

Define u_k as follows:

$$u_k = \sigma_k + \gamma_k + \sum_{j=1}^{N} \psi_j d_{jk} , \qquad (5.5)$$

where

$$\gamma_k = \gamma_{(I \cup k)} - \gamma_{(I)} \qquad (5.6)$$

Then the cost function for any given assignment I is given by:

$$C(I) = \sum_{k \in I} u_k + \sum_{j=1}^{N} \lambda_j \min_{k \in I} d_{jk} . \qquad (5.7)$$

Theorem III: If

$$C(I) \leq C(I \sim [k]) \quad \text{for } k=1,2 \qquad (5.8)$$

then

$$C(I \sim [k]) \leq C(I \sim [1,2]) \quad \text{for } k=1,2 \qquad (5.9)$$

Theorem III states that our cost graph if the given vertex has a cost less than the cost of two vertices leading to it, then the cost of the predecessor of the two vertices is greater than either of the two vertices.

Theorem IV: Given an index set $X \subseteq I$, containing r elements with the following property:

$$C(I) \leq C(I \sim [x]) \quad \text{for each } x \in X \qquad (5.10)$$

Then for every sequence $R^{(1)}, R^{(2)}, \ldots, R^{(r)}$, which are subsets of X, such that $R^{(k)}$ has k elements and $R^{(k)} \subset R^{(k+1)}$, the following is true:

$$C(1) \leq C(I \sim R^{(1)}) \leq C(I \sim R^{(2)}) \leq C(I \sim R^{(3)}) \leq \ldots C(I \sim R^{(r)}) \qquad (5.11)$$

Theorem IV states that if a given vertex has a cost less than the cost of any vertex along the path leading to it, then the sequence of costs encountered along any one of these paths decrease monotonically. Thus in order to find the optimal allocation policy, it is sufficient to follow every path of the cost graph until the cost increases and no further. This will given a locally optimum allocation of which the global optimum is one of them.

This allow us to reduce the solution space of the integer program. Once we find a local optimum then we know that any more file allocation is not required. Hence the integer program will not have to search for solutions in that part of the solution space.

Theorem V: All optimal allocations will include site i if

$$\lambda_i \min_{j \neq i} d_{ij} > Z_i , \qquad (5.12)$$

where

$$Z_i = \sigma_i + \gamma_i + \sum_{j=1}^{N} \psi_j d_{ji} . \qquad (5.13)$$

Theorem VI: No optimal allocation including more than one site will include site i if the following is true:

$$Z_i > \sum_{j=1}^{N} \lambda_j (\max_k d_{jk} - d_{ji}) . \qquad (5.14)$$

Theorem V states that if the cost of having a local file copy is smaller than the smallest possible cost of sending queries elsewhere, then a local copy should unquestionably be included in the optimal allocation. This will require certain nodes to have files allocated there. Therefore the solution space required to be searched by the integer program will be reduced. The integer program can ignore any possible file allocations that excludes the files that are unquestionably allocated by Theorem V.

Theorem VI states the other extreme of Theorem V. If it costs more to maintain a local copy than we could possible save by having one, then we do not want one. This theorem will allow the integer program to ignore allocating files in locations that are definitely too costly and therefore reduce the possible solution space for the integer programming solution. The integer program can ignore file allocations which include file allocations that are ruled out by Theorem VI.

Define the following:

$$m_i = \lambda_i \min_{j \neq i} d_{ij} , \qquad (5.15)$$

and

$$M_i = \sum_{j=1}^{N} \lambda_j (\max_k d_{jk} - d_{ji}) . \qquad (5.16)$$

Then for each i the real line is partitioned by m_i and M_i into three regions.

If Z_i falls into region I then it should unquestionably be included. If Z_i falls into region III it will be excluded unless all Z_i fall in region III, then just include the largest one. If Z_i falls in region II then it must be further considered.

These theorems are useful because now the region in which the integer program must search for solutions is reduced. We can force files to be allocated in region I and not be allocated in region III.

Theorem VII: By choosing $d_{jk}=1$, $\sigma_{kl}=\sigma_l$ and a completely connected network, then the cost function reduces to the following:

$$C(I_\ell)=\sum_{\ell=1}^{M}\left[\sum_{k=1}^{N}\sigma_\ell x_{k\ell}+\sum_{i=1}^{N}\sum_{k=1}^{N}\psi_{i\ell}x_{k\ell}+\sum_{k=1}^{N}\lambda_{k\ell}(1-x_{k\ell})-\sum_{i=1}^{L}\alpha_i\beta_{\ell i}p_{i\ell}(I_\ell)\right]. \quad (5.17)$$

The decision rule for the initial file assignment is $x_{ij}=1$ if:

$$\lambda_{ij}+\psi_{ij}\geq\psi_{kj}+\lambda_{kj} \quad 1\leq k\leq N \quad k\neq i \quad (5.18)$$

This theorem just states the initial file allocation for this special type of network.

Theorem VIII: Given a node k and a file j, then to store a copy of j in k leads to a reduction of the overall costs if the following holds:

$$\lambda_{kj}+\psi_{kj}>\sum_{i=1}^{N}\psi_{ij}+\alpha_j-\gamma_j. \quad (5.19)$$

For allocating a new copy later, the theorem states that if allocation of a new copy leads to a cost decrement for the host node which is greater than the overcost due to the necessity of updating the additional copy and storage cost, then we should store the file there.

This theorem is useful because the solution to the file allocation problem does not require integer programming and therefore is not NP complete. It enables simple calculations to determine file allocation.

Theorem IX: Define the following allocations: $I'=I\cup[k]$ and $I''=I'\cup[i]$. If site i satisfies

$$Z_i>\sum_{j=1}^{N}\lambda_j\max[(d_{jk}-d_{ji}),0], \quad (5.20)$$

for some site k in the network, then $C(I'')>C(I')$.

Theorem IX states that if site i is sufficiently costly, then by adding site i to an allocation which already includes k increases the total cost.

Theorem X: Define the following allocations $I'''=I\cup[i]$. If sites i and k satisfy

$$Z_i-Z_k>\sum_{j=1}^{N}\lambda_j\max[(d_{jk}-d_{ji}),0], \quad (5.21)$$

then

$$C(I''')>C(I') \quad (5.22)$$

Theorem X states that if:

$$Z_i-Z_k>\sum_{j=1}^{N}\lambda_j\max[d_{jk}-d_{ji},0] \quad (5.23)$$

is satisfied, then replacing site k by site i in an allocation will increase the cost.

Theorem XI: A site i cannot be included in any optimal allocation if there exists another site k in the network such that

$$Z_i-Z_k>\sum_{j=1}^{N}\lambda_j\max[d_{jk}-d_{ji},0]. \quad (5.24)$$

Theorem XI states that instead of determining that no more than one of some group of geographically close sites can be included in an optimal solution, Theorem XI states that certain sites may be excluded from being optimal allocations by the existence of better nearby sites. This theorem is useful because it allows us to reduce the possible solution space of the integer program. The integer program can ignore file allocations that allocate separate copies of the same file at geographically close nodes.

Theorem IX, theorem X, and theorem XI are extensions of work done by Grapa and Belford [6].

Theorem XII: The following are equivalent:

$$P_i(I)=\sum_{j=1}^{N}\prod_{\substack{k=1\\k\neq j}}^{N}(1-x_k)x_jP_{ij}$$

$$+\sum_{h=1}^{N}\sum_{\substack{j=1\\j>h}}^{N}\prod_{\substack{k=1\\k\neq j\\k\neq h}}^{N}(1-x_k)x_jx_h[P_{ij}+(1-P_{ij})P_{ih}]\cdots$$

$$+\sum_{a=1}^{N}\sum_{b>a}^{N}\cdots\sum_{\substack{j=1\\j>h\\\vdots\\j>b\\j>a}}^{N}\prod_{\substack{k=1\\k\neq j\\k\neq h\\\vdots\\k\neq b\\k\neq a}}^{N}x_k[P_{ij}+(1-P_{ij})P_{ih}+(1-P_{ij})(1-P_{ih})P_{ig}\cdots]$$

and

$$= \sum_{j=1}^{N} x_j p_{ij} - \cdots$$

$$+ (-1)^{N+1} \sum_{j=1}^{N} \sum_{\substack{k=1 \\ k>j}}^{N} \prod_{\substack{m=1 \\ m\neq j \\ m=k}}^{N} x_m p_{im}$$

$$(-1)^{N} \sum_{j=1}^{N} \prod_{\substack{m=1 \\ m\neq j}}^{N} x_m p_{im} + (-1)^{N+1} \prod_{m=1}^{N} x_m p_{im} \qquad (5.26)$$

where p_{ij} is the probability of accessibility between nodes i and j.

This theorem essentially states a generalization of the well known probability law:

$$P(ABC)=P(A)+P(B)+P(C)-P(AB)-P(BC)-P(AC)+P(ABC)\,. \qquad (5.27)$$

6. CONCLUSIONS

We have formulated the file allocation problem in a C^3 context where vulnerability is an issue. The formulation considers:

1. The probability of commander accessing files;
2. The importance of commanders;
3. The importance of particular files to particular commanders.

The theorems have provided ways to cut down on the possible file allocations (solution space) in which the integer program has to search. Therefore, we reduce the amount of time required to solve for a solution using integer programming.

We have extended and proved twelve theorem, all applicable to the new formulation.

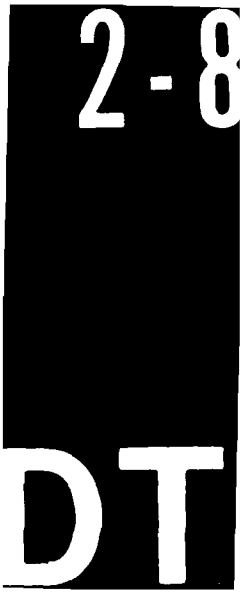
In the C^3 context, we may not need inger programming to solve for a solution if we make the following assumptions:

1. Connected network where all nodes are connected to each other;
2. Cost of communication is same between all nodes;
3. Cost of storing a file is the same.

In the area of further research, we plan to explore the effects of where the data sources are located on the file allocation problem. This would be applicable in a C^3 context, where sensor data may come from only a fixed set of nodes. The data must also pass through a processing node. The location of where the processing node should be is also an optimization problem which can be incorporated into our formulation.

REFERENCES

1. W.W. Chu, "Optimal File Allocation in a Multiple Computer System." IEEE Transactions on Computer, Vol. C-18, No. 10 (1969).
2. R.G. Casey, "Allocation of Copies of a File in an Information Network," AFIPS, SJCC (1972).
3. K.P. Eswaran, "Placement of Records in a File and File Allocation in a Computer Network," Information Processing, North Holland (1974).
4. H.L. Morgan, J.D. Levin, "Optimal Program and Data Locations in Computer Networks," CACM, Vol. 20, No.5 (1977).
5. C.V. Ramamoorthy, B.W. Wah, "The Placement of Relations on a Distributed Relational Data Base," Proc. 1st. Int. Conference on Distributed Computing Systems (1979).
6. E. Grapa and G.G. Belford, "Some Theorems to Aid in solving the File Allocation Problem," CACM, Vol. 20, No. 11, pp. 878-882 (1977).
7. R.G. Casey, "Allocation of Copies of a File in an Information Network," AFIPS, SJCC (1972).


2-8
DT