January 1992 LIDS-P-2090

SOME SIMPLE DESIGN PROBLEMS FOR DISTRIBUTED ORGANIZATIONS[1]

Daniel C. Lee[2]
John N. Tsitsiklis[2]

Abstract

We consider the problem of designing an organization that can support the execution of certain decision–making activities. We model an organization by a graph that describes its communication capabilities, and we assume that the desired decision–making activity defines another graph describing the required communication. We then formulate a number of variants of the problem of optimally choosing the organizational graph so as to satisfy the communication requirements. For some of these variants we provide algorithms that solve the organizational design problem, while for some others we show that they are computationally intractable.

---

1. Research supported by the ONR under grant N00014–85–K–0519 (NR 649–003) and by the NSF under grant ECS–8552419

2. Laboratory for Information and Decision Systems and the Operations Research Center, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

1. INTRODUCTION

A divisionalized organization, designed to perform a certain task, often accomplishes its objectives by partitioning that task into subtasks, and by assigning subtasks to its divisions. Generically, some of these subtasks interact, that is, they cannot be carried out by the corresponding divisions in isolation. This introduces the need for communication between certain pairs of divisions. In this paper, we focus on such communicational aspects of organizations. In particular, we describe an organizational structure by specifying "who talks to whom" or, mathematically, by means of an undirected graph $G_O = (V_O, A_O)$, called the ***organizational graph***, that specifies the communication capabilities available to the organization. In particular, the nodes of G_O correspond to the divisions and the presence of an arc $(i,j) \in A_O$ signifies that divisions i and j can communicate with each other. We will be always assuming that $(i,i) \in A_O$ for all $i \in V_O$, which expresses the natural fact that any division can communicate with itself. Note that $(i,k) \notin A_O$ indicates that division i and k cannot communicate, even if $(i,j) \in A_O$ and $(j,k) \in A_O$ for some i.

Certain tasks might require communication between all divisions of the organization, in which case the most suitable organization would corrrespond to a complete graph. On the other hand, there are numerous situations in which the task to be executed has a special structure, in which case fewer communication links suffice. This paper deals with the problem of designing the "best" (in a sense to be defined later) organizational structure that can accomodate the communication requirements.

For our problem to be well–defined, we need a mathematical representation of the communication requirements of the task to be executed. This is done in terms of another undirected graph $G_T = (V_T, A_T)$, called the ***task graph.*** The nodes of G_T correspond to subtasks, while the presence of an arc $(i,j) \in A_T$ signifies that subtasks i and j are interdependent. Each subtask $i \in V_T$ is to be assigned to a division $\sigma_i \in V_O$, the division primarily responsible for that task. In our model, the interdependence between two subtasks i and j is handled by assigning to a particular division, denoted by σ_{ij}, the responsibility of keeping track of this interdependence. It is then natural to require that σ_{ij} can communicate to both σ_i and σ_j.

Formally, we have the following definition. Given a task graph G_T, a ***valid*** organizational structure is defined as a graph G_O, together with a mapping $\sigma : V_T \cup A_T \mapsto V_O$ such that $(\sigma_{ij}, \sigma_i) \in A_O$ and $(\sigma_{ij}, \sigma_j) \in A_O$ for every $(i,j) \in A_T$.[3]

The organizational design problems to be considered will all be of the following form: given the task graph G_T, find a valid organizational structure (G_O, σ), subject to some additional constraints that remain to be specified, so as to optimize a given performance measure. The following are some possible additional constraints on (G_O, σ):

a) We can impose a constraint on the cardinality of V_O, that is, on the number of available divisions.

3. We will mostly use the notation σ_i and σ_{ij} instead of the more standard functional notation $\sigma(i)$ and $\sigma(i,j)$.

b) We could assume that the graph G_O is given, which would correspond to the case where we are dealing with a preexisting organization. In this case, all that remains to do is to design the mapping σ in some desirable way. This would resemble to an assignment problem whereby subtasks, and their interactions, are to be assigned to divisions. An implicit assumption here is that all divisions of the preexisting organization are equally capable and versatile so that any subtask could be assigned to any division.

c) Going one step further, we could assume that the graph G_O is given and that the division σ_i in charge of subtask i is also prespecified for each i. In this case, we only have to choose which division would be responsible for the handling of each subtask interaction. That is, we only need to choose the values of σ_{ij}, for every $(i,j) \in A_T$. Such a problem would correspond to a situation where each subtask is of a specific nature, intimately linked to a particular division of the organization which is the only division capable of handling it. On the other hand, the implicit assumption is that the handling of the interactions between subtasks i and j does not involve any particular expertise and can by handled by any division, as long as the necessary communication links are in place.

Next, we have to specify some relevant performance criteria. Our first criterion pertains to load balancing. The divisions of any organization have limited resources and there is a limit on the number of their responsibilities. It is plausible that the division assigned the largest number of responsibilities could be a bottleneck, and that its load should be minimized. Formally, we define the load ℓ_i of division $i \in V_O$ to be the cardinality of the set $\sigma^{-1}(i)$. This is equal to the number of subtasks plus the number of interactions that this division is responsible for. By defining the load this way, we are implicitly assuming that handling a subtask takes the same amount of resources with the handling of an interaction. The maximum load L is defined by $L = \max_{i \in V_O} \ell_i$.

Another performance criterion relates to the amount of communication resources employed by the organization. This is a natural measure given that communication is often a constrained resource. In fact, we will be considering two alternative ways of measuring communication resources.

A. Given an organization G_O, let C_1 be the number of arcs $(i,j) \in A_O$ for which $i \neq j$. Thus, C_1 measures the number of communication links that have to be in place when setting up the organization.

B. In an alternative method of measuring communication, we can measure the total amount of communication traffic in the organization. In particular, for every $(i,j) \in A_T$, division σ_{ij} has to exchange messages with divisions σ_i and σ_j, which leads, in general, to 2 units of communication traffic. However, if σ_{ij} coincides with σ_i, then we shouldn't "charge" for communication between σ_{ij} and σ_i. Thus, the total communication traffic between all pairs of (distinct) divisions, to be denoted by C_2, can be defined as being equal to $2|A_T|$ minus the number of elements (i,j) of A_T for which $\sigma_{ij} \in \{\sigma_i, \sigma_j\}$.

It should be clear that the objectives of load balancing and low communication requirements compete with each other. For example, communication requirements are lowest if all subtasks are

assigned to a single division, resulting to a most unbalanced load. In our problem formulations, we will deal with this tradeoff by attempting to optimize one of the performance measures while constraining the other. So, for example, we might wish to minimize C_1 subject to a constraint that L be bounded by some given L^*.

Let us close by noting that the design problems that we have formulated are reminiscent of the "mapping" problem [B] that arises when subroutines are to be mapped to a parallel processing architecture. However, our problems have some distinctive features of their own, which make them different from the mapping problems that have been considered in the computer science literature. Also, while there is some literature on mathematical formulations of the organizational design problem [BT, SL], our formulation appears to be new.

A Motivating Example

We will now describe, in some detail, an example that provides a more concrete illustration of the general framework we have introduced.

Consider an organization whose objective is to come up with an n–dimensional decision vector $x = (x_1, \ldots, x_N)$. Let σ_i denote the division of the organization that will be responsible for the decision x_i. We assume that the performance of a decision vector x is judged according to a cost function $J : \Re^N \mapsto \Re$ and that the organization's aim is to choose a decision vector x that minimizes J. Let us further assume that the organization strives towards this objective by mimicking a gradient algorithm. That is, a preliminary decision vector x is chosen, which is then updated by making a correction along a direction of cost improvement, as in the gradient algorithm $x := x - \gamma \nabla J(x)$.

Let us now assume that the cost function J has the structure

$$J(x) = \sum_{i=1}^{N} J^i(x_i) + \sum_{(i,j) \in A_T} J^{ij}(x_i, x_j).$$

Here, J^i captures the immediate cost to division σ_i due to its own decision, whereas J^{ij} reflects the coupling of the decisions of divisions σ_i and σ_j.[4] The set A_T indicates the set of all pairs of interacting divisions. We assume that for every pair of interacting divisions (σ_i, σ_j), with $(i,j) \in A_T$, there is some division, denoted by σ_{ij}, that will have the responsibility of measuring and suitably communicating the effects of these interactions. This can be done most conveniently by using the following distributed implementation of the gradient algorithm [T]:

Assuming that the current value of x_i is stored at division σ_i:

1. Each division σ_i evaluates $\lambda_i^i = (\partial J^i / \partial x_i)(x_i)$.
2. The values of x_i and x_j are communicated to division σ_{ij}.

4. In this example, we are assuming only pairwise interactions between divisions. The example goes through with more general interactions as well, at the expense of havier notation.

3. For each $(i,j) \in A_T$, division σ_{ij} evaluates $\lambda_i^{ij} = (\partial J^{ij}/\partial x_i)(x_i, x_j)$ and $\lambda_j^{ij} = (\partial J^{ij}/\partial x_j)(x_i, x_j)$, and transmits the result to divisions σ_i and σ_j, respectively.
4. Each division σ_i uses the received messages to compute $(\partial J/\partial x_i)(x)$ and update x_i.

Clearly, the communication requirements of this algorithm are that σ_{ij} should be able to communicate to divisions σ_i and σ_j, in conformance to our general model. Note that C_1 measures the number of pairs of divisions that need to communicate with each other. On the other hand, C_2 is proportional to the number of partial derivatives that would have to be communicated during each iteration; both are equally meaningful measures of communication. Furthermore, according to our general definition, the load ℓ_i of a division i reflects the number of partial derivatives that have to be evaluated by that division during a typical iteration. In many optimization problems, derivative evaluation can be the most time consuming step, so this definition of the load makes sense.

The remainder of this paper is organized as follows. Each one of Sections 2, 3, and 4 considers the problem under different assumptions on "how much" of G_O and of the mapping σ is assumed to be predetermined. For each choice of assumptions, we consider a few different problems depending on the particular choice of performance measure (L, C_1, or C_2).

2. THE CASE WHERE THE ORGANIZATIONAL STRUCTURE IS FIXED

Let there be given a task graph G_T. In this section, we consider the organizational design problem under the assumption that the organizational graph G_O is also given, has the same number n of nodes as the task graph G_T, and we also have the constraint $\sigma_i = i$ for all i. Thus, it only remains to choose the value of σ_{ij} for every $(i,j) \in A_T$.

Note that it is easy to determine whether a valid organization exists. In particular, we only need to check whether for every $(i,j) \in A_T$ there exists some k for which $(i,k) \in A_O$ and $(j,k) \in A_O$.

Minimizing the maximum load L

The first problem we consider is the following. We wish to find a valid organization which minimizes the maximum load L, subject to the constraints mentioned in the introduction to this section. This is equivalent to minimizing the maximum, over all divisions k, of the number of pairs $(i,j) \in A_T$ assigned to that division; equivalently, the number of pairs $(i,j) \in A_T$ for which $\sigma_{ij} = k$.

Theorem 2.1: The above defined problem can be solved in polynomial time by solving a sequence of linear network flow problems.

Proof: We start by considering the following related problem: given a value L^*, does there exist a valid organization, satisfying all of our constraints and such that $L \leq L^*$? This problem is equivalent to determining the feasibility of a network flow problem defined on bipartite graph, as we now proceed to show.

For each element (i,j) of A_T, we create a "supply" node s_{ij}, and for each element of V_O, we create a "demand" node d_i. We introduce a variable $x_{ij,k}$ which is equal to 1 if $\sigma_{ij} = k$, and

which is equal to 0 otherwise. We have the constraints $\sum_{k=1}^{n} x_{ij,k} = 1$, reflecting the fact that each interacting pair (i,j) must be assigned to some division k. (Thus, the supply at each supply node is equal to 1.) Furthermore, since σ_{ij} must be able to communicate to i and j, we have the following additional constraint: if either $(i,k) \notin A_O$ or $(j,k) \notin A_O$, then $x_{ij,k} = 0$. We finally have the constraint $1 + \sum_{(i,j) \in A_T} x_{ij,k} \leq L^*$, for all k. The question of the existence of a valid organization satisfying all of our constraints is equivalent to choosing the variables $x_{ij,k}$ so as to satisfy the above introduced constraints. The latter is easily seen to be equivalent to the question of existence of an integer feasible solution in a network flow problem with integer coefficients, and can be solved in polynomial time [PS].

In order to find the optimal value of L, we could solve the above network flow problem for all values of L^* from 1 to n^2 and this would be a polynomial time algorithm for the original problem. In fact a faster algorithm is obtained if we perform binary search for the optimal value of L; in particular, it would suffice to solve $O(\log n)$ network flow problems. **Q.E.D.**

Minimizing a communication measure

The problem of minimizing the number C_1 of arcs is vacuous because G_O is assumed to be given and therefore C_1 is predetermined. The problem of minimizing C_2 is also very simple, as we now discuss. If $(i,j) \in A_T$ and $(i,j) \in A_O$, then we should let σ_{ij} be equal to either σ_i or σ_j; if on the other hand, $(i,j) \notin A_O$, then we have to let σ_{ij} be equal to an arbitrary element k of V_O such that $(i,k) \in A_O$ and $(j,k) \in A_T$. It should be clear that this method results in the minimal possible value of C_2.

A more interesting problem is dealt with in the following result.

Theorem 2.2: Consider the problem of minimizing C_2 subject to an upper bound L^* on the maximum load L. This problem can be formulated as a linear network flow problem and can be therefore solved in polynomial time.

Proof: The main ideas are similar to the proof of Theorem 2.1. For each element $(i,j) \in A_T$, we create a supply node s_{ij}, and for each element of V_O, we create a demand node d_i. We introduce a variable $x_{ij,k}$ which is equal to 1 if $\sigma_{ij} = k$ and is 0 otherwise. As in the proof of Theorem 2.1, we have the constraint $x_{ij,k} = 0$ if either $(i,k) \notin A_O$ or $(j,k) \notin A_O$ and the constraint $\sum_{(i,j) \in A_T} x_{ij,k} \leq L^*$. The objective is to minimize the cost function $\sum_{i,j,k} c_{ij,k} x_{ij,k}$, where $c_{ij,k}$ is equal to 1 if k is equal to i or j, and $c_{ij,k} = 2$, otherwise. It is clear that, for any choice of the variables $x_{ij,k}$, the value of the cost function is equal to the value of C_2 in the resulting organization. The optimization problem we have just defined is a min–cost linear network flow problem and can be solved in polynomial time. **Q.E.D.**

3. THE CASE WHERE THE STRUCTURE OF THE ORGANIZATION IS GIVEN UP TO ISOMORPHISM

Let there be given a task graph G_T. In this section we also assume that the graph G_O is given

and has the same number of nodes as G_T. However, in contrast to the preceding section, we do not impose the requirement that $\sigma_i = i$ for all i. Instead, we impose the milder requirement that each division is assigned exactly one subtask, that is, the mapping $i \mapsto \sigma_i$ is a permutation. Our main result states that even the problem of existence of a valid organization is difficult.

Theorem 3.1: The problem of deciding whether there exists a mapping σ such that the organization (G_O, σ) is valid with respect to a given task graph G_T is NP–complete.

Proof: That the problem belongs to NP is evident: if we have a YES instance, the mapping σ provides a certificate.

We now note that the problem of interest is equivalent to the following:

Problem P: Does there exist a permutation $i \mapsto \sigma_i$ such that whenever $(i,j) \in A_T$, then the distance of σ_i and σ_j (in the graph G_O) is at most 2.

For any graph G, let $T(G)$ be a graph with the same set of nodes and such that (i,j) is an arc of $T(G)$ if and only if the distance of i and j in the graph G is at most two. We then see that we are dealing with the following problem:

Problem P': Given two graphs G_T and G_O with the same number of nodes, is G_T isomorphic with a subgraph of $T(G_O)$?

We recall the problem CLIQUE which is known to be NP–complete [GJ] and which is the following: Given a graph G, and an integer k, does G have a clique of size k?

Lemma 1: CLIQUE remains NP–complete even if we restrict to instances for which $k \geq n/2 + 2$ and for which the degree of each node is at least $n/2 + 1$, where n is the number of nodes in the graph G.

Proof: Let there be given an instance (G,k) of the CLIQUE problem and let m be the number of nodes of G. We construct a new graph G', as follows. The graph G' consists of the graph G together with $m + 4$ additional nodes. All of these additional nodes are connected by means of an arc to every other node in G'. Note that G' has $n = 2m + 4$ nodes. It is clear that G has a clique of size k if and only if G' has a clique of size $k' = m + 4 + k$. Note that $k' \geq m + 4 = n/2 + 2$. Also, the degree of each node in G' is at least $m + 4 \geq n/2 + 1$. We have thus reduced the general CLIQUE problem to the special case for which $k \geq n/2 + 2$ and the degree of each node is at least $n/2 + 1$, thus establishing the desired result. **Q.E.D.**

Recall now the SUBGRAPH ISOMORPHISM problem: given two graphs G and G', is G isomorphic to a subgraph of G'? Since CLIQUE is a special case of SUBGRAPH ISOMORPHISM, and in view of Lemma 1, we see that SUBGRAPH ISOMORPHISM is NP–complete even if we restrict to graphs for which the degree of each node is at least $n/2 + 1$.

We will be needing another graph transformation. Given a graph G, we denote by $Q(G)$ the graph which is the same as G except that each arc of G is replaced by a sequence of 3 arcs, as shown in Fig. 1.

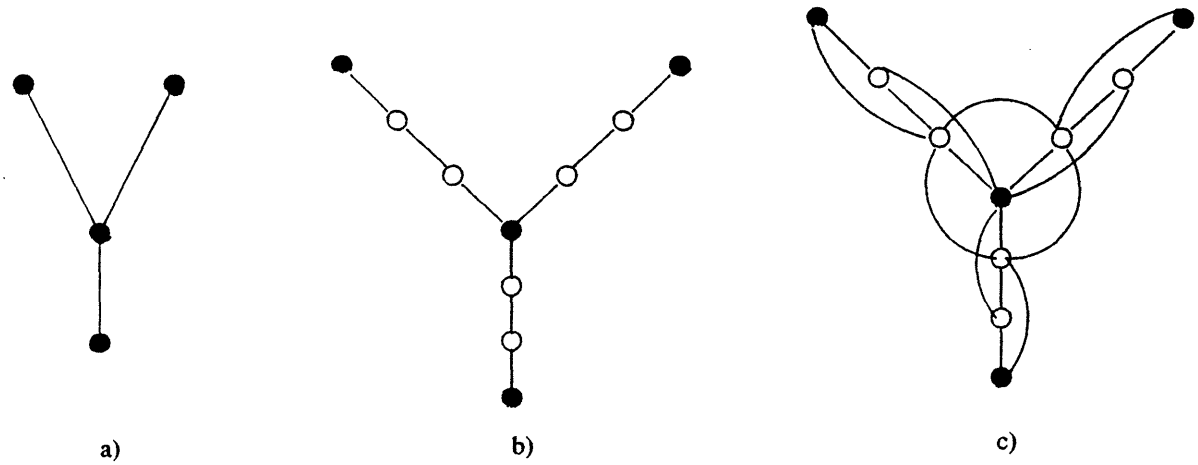

Figure 1: a) A graph G; b) the graph $Q(G)$; c) the graph $T(Q(G))$.

We introduce some more notation. If G is a graph and i is a node of that graph, we use $T(Q(i))$ to denote the image of node i when the transformations Q and T are applied in succession.

Lemma 2: Let G be a graph in which all nodes have degree at least d.
a) If i is a node of G, then $T(Q(i))$ has degree at least $2d$; all nodes of $T(Q(G))$, not of the form $T(Q(i))$ for some i, have degree bounded by $n+1$.
b) If (i,j) is an arc of G, then the distance [in the graph $T(Q(G))$] between $T(Q(i))$ and $T(Q(j))$ is equal to 2; if (i,j) is not an arc of G, then the distance between $T(Q(i))$ and $T(Q(j))$ is larger than 2.

Proof: a) If a node in G has degree $\delta \geq d$, then the corresponding node in $T(Q(G))$ is connected to its neighbors in $Q(G)$ (there are δ of them) and to the neighbors of these neighbors (there are δ of them as well, for a total of $2\delta \geq 2d$.

If a node in $T(Q(G))$ is not of the form $T(Q(i))$, then it has only 2 neighbors in the graph $Q(G)$. One of these neighbors has a single extra neighbor; the other one corresponds to a node of the original graph G and has at most $n-2$ extra neighbors. Thus, the degree of the node of $T(Q(G))$ under consideration is at most $2+1+n-2=n+1$.
b) Evident from Figure 1. **Q.E.D.**

Note that if all nodes of G have degree at least $n/2+1$, then nodes, of the form $T(Q(i))$ will

have degree at least $n+2$. All other nodes of $T(Q(G))$ will have degree at most $n+1$. Thus, for each node of $T(Q(G))$, it can be immediately determined whether it is of the form $T(Q(i))$ or not.

Lemma 3: Let G and G' be graphs in which all nodes have degree at least $n/2+1$. Then, G is isomorphic to a subgraph of G' if and only if $T(Q(G))$ is isomorphic to a subgraph of $T(Q(G'))$.

Proof: If G is isomorphic to a subgraph of G', it is evident that $T(Q(G))$ is isomorphic to a subgraph of $T(Q(G'))$. It only remains to prove the reverse implication.

Suppose that $T(Q(G))$ is isomorphic to a subgraph of $T(Q(G'))$. Consider any node of $T(Q(G))$ which has degree larger than $n+1$. Such a node is of the form $T(Q(i))$ for some node i of G, by the preceding discussion. Since $T(Q(G))$ is isomorphic to a subgraph of $T(Q(G'))$, node $T(Q(i))$ is mapped to some node of $T(Q(G'))$ that has also degree larger than $n+1$, and is therefore of the form $T(Q(i'))$, where i' is a node of G'.

Suppose that (i,j) is an arc of G. Then, $T(Q(i))$ and $T(Q(j))$ have both degree larger than $n+1$ and their distance in $T(Q(G))$ is equal to 2 (Lemma 2). Since $T(Q(G))$ is a subgraph of $T(Q(G'))$, the nodes $T(Q(i))$ and $T(Q(j))$ are mapped to some (distinct) nodes in $T(Q(G'))$ which are of degree larger than $n+1$. In particular, these latter nodes of $T(Q(G'))$ must be of the form $T(Q(i'))$ and $T(Q(j'))$, for some nodes i' and j' of G'. Since the distance of $T(Q(i))$ and $T(Q(j))$ is equal to 2, the distance of $T(Q(i'))$ and $T(Q(j'))$ must be at most 2. Using Lemma 2(b), we conclude that (i',j') is an arc of G' (Lemma 1). So, by mapping i and j to i' and j', respectively, and by mapping similarly all other nodes of G to nodes of G', we see that G is isomorphic to a subgraph of G', which concludes the proof of the lemma. **Q.E.D.**

We notice that Lemma 3 reduces a special case of SUBGRAPH ISOMORPHISM (shown earlier to be NP–complete) to problem P', with the identification $G_O = Q(G')$ and $G_T = T(Q(G))$, except that we have not enforced the requirement that the two graphs in an instance of problem P' have the same number of nodes. This is easily taken care of, by adding a number of zero–degree nodes, and we conclude that problem P' is NP–complete, and the proof of Theorem 3.1 has been completed. **Q.E.D.**

We have shown that it is difficult to even determine whether a valid organization does exist. It follows that the problem of determining an optimal valid organization is also difficult (NP–hard), for any nontrivial choice of the performance criterion.

4. THE CASE WHERE ONLY THE NUMBER OF NODES OF G_O IS FIXED

We now consider the case where G_T is given and we require that G_O has the same number of nodes as G_T; no other constraints are imposed on G_O. We also impose the requirement that each node i of G_T is mapped to a different node σ_i of G_T.

Under the above constraints, the problem of designing a valid organization that minimizes C_1 is trivial: assuming that G_T is connected with n nodes, let $G_O = (\{1,\ldots,n\},\ \{(1,2),\ldots,(1,n)\})$ (a "star" graph), and let $\sigma_{ij} = 1$ for all $(i,j) \in A_T$. We then have $C_1 = n-1$. Since G_T is connected,

it is clear that G_O must also be connected and therefore no valid organization could have less than $n-1$ arcs.

If we impose a load balancing constraint $L \leq L^*$ and attempt to minimize C_1 subject to that constraint, we obtain an apparently more difficult problem. We conjecture that this problem is NP–hard, although we have not been able to establish this result.

The last problem to be considered is dealt with by the following result.

Theorem 4.1: Under the assumptions of this section, the problem of designing a valid organization in which C_2 is minimized subject to the constraint $L \leq L^*$, can be formulated as a min–cost linear network flow problem and can be solved in polynomial time.

Proof: Instead of assuming that nothing is given about the graph G_O, let us assume instead that G_O is a complete graph, and minimize C_2 subject to the constraint $L \leq L^*$. By noting that the criterion C_2 penalizes only those arcs that are "used", we realize that this is the same problem. However, this problem is a special case of the problem considered in Theorem 2.2, and the result follows from Theorem 2.2. **Q.E.D.**

5. CONCLUSIONS

We have formulated a new class of design problems for decentralized organizations. We have derived solution procedures for some of these design problems, and we have showed that another variation leads to NP–hard problems. We believe that our formulation captures some generic features of organizational design problems.

REFERENCES

[B] Bokhari, S. H., "On the mapping problem," *IEEE Trans, on Computers,* Vol. 30, pp. 550–557, 1981.

[BT] Boettcher, K. L. and Tenney, R.R., "On the analysis and design of human information processing organizations," *Proceedings of the 8th MIT/ONR Workshop on C^3 Systems,* M. Athans and A. Levis, Editors, pp. 69–74, December 1985.

[GJ] Garey, M.R., and Johnson, D.S., *Computers and Intractability: a Guide to the Theory of NP–completeness,* W.H. Freeman, New York, 1979.

[PS] Papadimitriou, C.H., and Steiglitz, K., *Combinatorial Optimization: Algorithms and Complexity,* Prentice Hall, New Jersey, 1982.

[SL] Stabile, D.A., and Levis, A.H., "The design of information structures: basic allocation strategies for organizations", *Proceedings of the 6th MIT/ONR Workshop on C^3 Systems,* M. Athans, E. Ducot, A. Levis, and R.R. Tenney, Editors, pp. 219–225, December 1983.

[T] Tsitsiklis, J.N., "Problems in Decentralized Decision Making and Computation", Ph.D. thesis, Dept. of EECS, MIT, Cambridge, MA, 1984.