THE MULTICOMMODITY MULTIPERIOD ASSIGNMENT PROBLEM I: A SPECIALIZED BRANCH AND BOUND ALGORITHM

M. Assad Elnidani*

Jay E. Aronson**

* Assistant Professor of Business Administration
Department of Business Administration
Faculty of Administrative Sciences and Economics
The University of Qatar
P. O. Box 2713
Doha, Qatar

** Associate Professor of Management Sciences and Information Technology
Department of Management Sciences and Information Technology
College of Business Administration
The University of Georgia
Brooks Hall
Athens, GA 30602, U.S.A.

THE MULTICOMMODITY MULTIPERIOD ASSIGNMENT PROBLEM I: A SPECIALIZED BRANCH AND BOUND ALGORITHM

M. Assad Elnidani*

Jay E. Aronson**

ABSTRACT

The multiperiod assignment problem is an important specialization of the three dimensional assignment problem, which is a generalization of the classical (two dimensional) assignment problem. This model describes the optimization problem of assigning people to activities (jobs) over several time periods. In the most general case, there is a cost assigning a person to an activity in each time period, and a cost of transferring a person from one activity in each period to another activity in the following period. The number of time periods is not restricted to equal the number of persons and activities.

We present an integer, multiperiod, multicommodity network flow model formulation of the multiperiod assignment problem. we develop a specialized branch and bound algorithm that exploits the multicommodity structure of the model. We report favorable computational results of an implementation of the algorithm, and compare them to those of a commercial mixed-integer, linear programming code.

KEYWORDS: Programming: Integer
Networks/Graphs
Networks/Graphs: Flow Algorithms
Programming: Assignment
Manpower Planning

1. Introduction

The multiperiod assignment problem is an important specialization of the three dimensional assignment problem, which is a generalization of the classical (two dimensional) assignment problem. This model describes the optimization problem of assigning m activities (jobs) to n persons over T discrete time periods. In this model, the most general case, two types of costs are considered. There is a cost associated with the assignment of person i to job j, in time period t. Also considered is the cost of transferring person i from job j in period t to job k in period t+1. The transfer cost can be set high as a penalty for situations where no employee is allowed to repeat a job assignment for two consecutive time periods. This situation may arise for health concerns, such as jobs that require handling of hazardous materials.

The number of time periods is not restricted to be equal to the number of persons or activities, as is the case in other related models [1, 2, 3, 4]. Additionally, the number of persons is not restricted to equal the number of activities, however, without loss of generality, it is assumed so. Clearly dummy persons or jobs can be added as needed. Costs associated with assigning a dummy person to a job, or a person to a dummy job and associated transfer costs are zero, unless otherwise stipulated.

We refer to the problem of optimally assigning n persons to m jobs over T time periods as the multiperiod assignment problem. Applications occur in the scheduling of parallel activities on concurrent processor computers, the assignment of salesmen to territories, the assignment of consultants to clients, the assignment of groups within an organization to various projects and problems in manpower planning [5].

The multiperiod assignment problem may be mathematically formulated as an integer, multicommodity network flow problem, where persons are the commodities [5]. Because of the problem's structure, it is possible to devise a specialized branch and bound algorithm that solves a set of shortest path problems. Here we discuss a specialized algorithm that exploits the multicommodity network structure directly.

In the next section, we present some general background material on multicommodity network flow problems and the definitions and notation used. In Section 3, we present the formulation of the problem as an integer, multicommodity network flow model. The specialized branch and bound algorithm and an example are presented in Section 4. In Section 5, we describe the implementation of the algorithm and report its computational results on randomly generated test problems. We also present a favorable comparison of our implementation to those of the commercial linear/integer programming package MPSX/MIP/370 [6]. A summary and conclusions are given in Section 6. We assume that the reader is familiar with linear programming [7-9], network models and optimization [7,10], and integer programming [11, 12]. For additional material on dynamic networks, see [13]. See [14] for a recent survey of multidimensional assignment problems. For work on related models, see [1-4, 15-23].

2. General Background Material

2.1 The Multicommodity Network Flow Problem

A multicommodity network flow problem can be thought of as a set of independent, single commodity network flow problem, in which the arcs are linked together by a set of mutual are capacity constraints [24,10]. When the commodities are measured in the same units (weight, volume, etc.), the linking are constraints have the generalized upper bounded (GUB) structure [25].

Algorithms for solving the multicommodity network flow problem have been presented by several authors. See Kennington and Helgason [10] for a complete theoretical development of the primal partitioning algorithm utilized in this paper to solve the linear programming relaxation of the problem. For additional material on the standard and integer multicommodity network flow problems, see [26-29, 30, 31]. The linear programming constraint matrix of a multicommodity network flow problem is not necessarily totally unimodular, except in special cases [26-28]. Thus, the integrality of basic feasible solutions to the linear programming relaxation of the multicommodity, multiperiod assignment problem to be presented is not guaranteed. We show an example in Section 4.

2.2 Definitions and Special Notation

Upper case English letters are used to define sets and matrices; lower case letters are for vectors and indices. The meaning will be clear from the context. Unless otherwise specified, the index i represents persons; the index j represents jobs; and the index t represents time periods.

Let G be a directed network [N,E], consisting of a finite set of nodes N = {1,2...,p} and a finite set of directed arcs, E = {(i,j), (k,h),..., (q,s)} joining pairs of nodes in N. Arc (i,j) is said to be directed away from node i and towards node j [7]. We assume that the network is connected; there exists a path in the network, to be defined shortly, between every pair of nodes i and j. We also assume, without loss of generality, that there exists no *parallel arcs*. Two or more arcs are said to be parallel if they have the same origin and destination nodes.

The integer, multicommodity network flow model may be represented by a collection of linked networks, each with a single source and a single sink. The following is a description of the definitions, terminology and notation:

Path: A path is defined from node h_0 to node h_p, as a sequence of arcs $P = \{(h_0, h_1), (h_0, h_1), ..., (h_{p-1}, h_p)\}$

Cycle: A cycle is a path in which $h_0 = h_p$. Thus a cycle is a closed path [11].

Arc Orientation: The orientation of arc $a = (i,j)$ in the path $P = \{(h_0, h_1), ..., (h_k, h_{k+1}), ..., (h_{p-1}, h_p)\}$, is + 1 if $((h_k, h_{k+1}) = (i,j)$ for some k; and - 1 if $(h_k, h_{k+1}) = (j,i)$ for some k.

Directed Acyclic Network: A network that contains no cycle of the form $C = \{(h_0, h_1), (h_1, h_2), ..., (h_s, h_0)\}$, such that the orientations on arcs (h_k, h_{k+1}) for all $k = 0, ..., s$ are equal, is said to be directed acyclic.

SC Source node. SC will always be represented by 1, or the lowest node number in N.

SK Sink node. SK will always be represented by p, or the highest node number in N (usually $p = |N|$)

n Number of persons (commodities).

m Number of jobs (= n).

T Number of consecutive time periods over which the assignment of the n persons to the m jobs occurs.

$(s,v)^i$ Arc (s,v) of commodity i. Node s is the origin node and node v is the destination node.

i. A singleton set SC, representing the single source node (Figure 1), SC= {1}.

ii. A singleton Set SK, representing the single sink node (Figure 2), SK = {2 + 2n (T - 1)}.

iii. A node set NtB, identifying job assignments at the beginning of period t, for t = 1,...,T; (Figures 1 - 4),
N^1_E = SC, and
$N^t_E = \{j + n(2t - 3) + 1 \mid J = 1,...,n\}$, for t = 2,...,T.

iv. A node set N^t_E, identifying job assignments at the end of period t, for t = 1,...,T: (Figures 1-4),
$N^t_E = \{j+2n(t-1) +1 \mid j =1,...,n\}$, for t =1,...,T-1, and
N^T_E = SK.

The sets N_B and N_E are the set of nodes identifying job assignments at the beginning of all time periods and the set of nodes identifying job assignments at the end of all time periods, respectively. These sets are defined as

$$N_B = \bigcup_{t=1}^{T} N^t_E,$$

$$N_E = \bigcup_{t=1}^{T} N^t_E,$$

The set of nodes, N, is then given by

$$N = N_B \cup N_E.$$

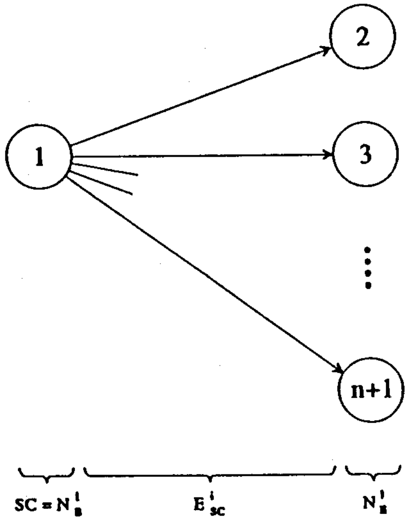

Figure 1: Source Arcs of Commodity i and the Source Node. The source arc set, E^1_{SC} = {(1,j+1) | j=1,...,n}, for i=1,...,n. Source node set SC = N^1_B = {1} and node set N^1_E = {2, 3,..., n+1} are also shown.

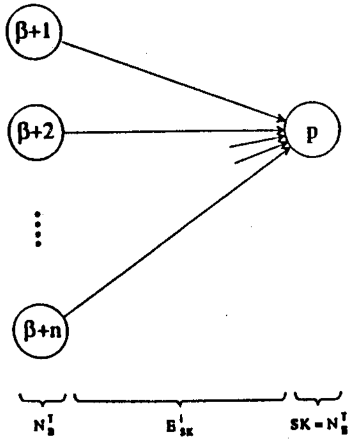

Figure 2: Sink Arcs of Commodity i and the Sink Node. The sink arc set E^1_{SK} = (β+j,p) | j=1,...,n; p=2+2n(T-1)}, for i=1,...,n. Node set N^T_B = {β+j | j=1,...,n} and sink node set SK = N^T_E = {2+2n(T-1)}, where β=n(2T-3)+1, are also shown.

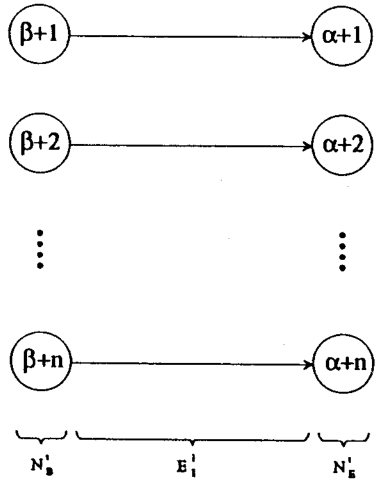

Figure 3: Inner Assignment Arcs of Commodity i. The inner assignment arc set E_I^j = {(β+j,α+j) | j=1,...,n}, for i=1,...,n; the node sets N_B^t = {β+j | j=1,...,n} and N_E^t = {α+j | j=1,...,n}, where α=2n(t-1)+1 and β=n(2t-3)+1, for t=2,...,T-1 are also shown.

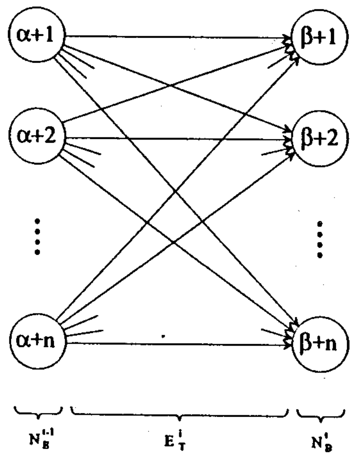

Figure 4: Transfer Arcs of Commodity i, E_T^i= {(α+k,β+j) | k=1,...,j; j=1,...,n} for i=1,...,n. The node sets N_B^t = {β+j | j=1,...,n} and N_E^t = {α+k | k=1,...,n}, where α = 2n(t-1)+1 and β = n(2t-1)+1, for t=1,...,T-1 are also shown.

3.2 The Arc Set

The arc set, E, contains the following distinguished subsets:

i. A *source arc set*: The origin node of a source arc is always node 1; the destination node of a source arc is a node from the set N1E. The source arc set of commodity i is given by Eisc, defined as follows: Eisk = {(j + q - n, q + 1) | j = 1,..., n + 1; q = 2n (T - 1) +1}.
The number of arcs in Eisk is n (Figure 2).

iii. An *inner assignment arc set*: The inner assignment arc set of commodity i is given by E^i_I and defined by:
E^i_I = {(q-n,q) | q = j + 2n (j - 1) + 1; J = 1,...,n; t = 2,..., T - 1}
The number of arcs in E^i_I is n (T - 2) (Figure 3).

iv. A *transfer arc set*: The transfer arc set is given by E^i_T and defined by:
E^i_T = {(j + q, k + q + n) | q = 2n (t - 1) + 1; t = 1,...,T - 1; j = 1,...,n; k =1,...,n}. The number of arcs in E^i_T is n^2 (T-1) (Figure 4).

The union of the above four sets defines the arc set E^i of commodity i,

$$E^i = E^i_{SC} \cup E^i_I \cup E^i_T \cup E^i_{SK}$$

Thus the arc set E for all n commodities is given by

$$E = \bigcup_{i=1}^{n} E^i$$

The set of all assignment arcs for all n commodities is given by:

$$E_A = \bigcup_{i=1}^{n} E^i_A,$$

and the set of all transfer arcs for all n comodities is given by

$$\text{(MCMAP) Min} \quad \sum_{i=1}^{n} \sum_{(s,v)\in E^i} c_{sv}^i x_{sv}^i \qquad (1)$$

subject to:

$$\sum_{s\in N_E^1} x_{1s}^i = 1 \qquad ; i=1,\ldots,n, \qquad (2)$$

$$x_{1j}^i - \sum_{s\in N_B^2} x_{js}^i = 0 \qquad ; i=1,\ldots,n \qquad (3)$$

$$; j \in N_E^1,$$

$$\sum_{s\in N_E^{t-1}} x_{sj}^i - x_{j,j+n}^i = 0 \qquad ; i=1,\ldots,n \qquad (4)$$

$$; j \in N_B^t$$

$$; j=2,\ldots,T-1,$$

$$x_{j-n,j}^i - \sum_{s\in N_B^{t+1}} x_{js}^i = 0 \qquad ; i=1,\ldots,n \qquad (5)$$

$$; j \in N_E^t$$

$$; t=2,\ldots,T-1,$$

$$\sum_{s\in N_E^{T-1}} x_{sj}^i - x_{jp}^i = 0 \qquad ; i=1,\ldots,n \qquad (6)$$

$$; j \in N_B^T$$

$$; p \in SK,$$

$$-\sum_{s\in N_B^T} x_{sp}^i = -1 \qquad ; i=1,\ldots,n \qquad (7)$$

$$; p \in SK,$$

$$\sum_{i=1}^{n} x_{sv}^i \le 1 \qquad ; (s,v) \in E_A, \qquad (8)$$

$$x_{sv}^i \ge 0, \text{ integer} \qquad ; i=1,\ldots,n \qquad (9)$$

$$; (s,v) \in E^i$$

The objective function (1) is to be minimized. Constraint sets (2) through (7) are the conservation of flow relations (see Table 1). Constraint set (8) consists of the mutual capacity relations on the assignment arcs. Constraint set (9) imposes the nonnegativity and integrality conditions associated with flow assignments. There are n [2+2n (t -1)] conservation of flow constraints and nT mutual arc capacity constraints. Note that when T = 2, constraint set (4) is dropped from the problem. The n networks of the n commodities are identical, directed acyclic networks. Every path in the network of commodity i is directed away from the source node and toward the sink node for i = 1,...n.

demand node (sink). For each commodity i, the source node has a requirement of $r^i_1 = +1$, and the sink node has a requirement of $r^i_p = -1$. The remaining nodes s, s = 2,...,p - 1, have requirements of $r^i_s = 0$.

3.5 Mutual Arc Capacity Constraints

Mutual arc capacity constraints are associated only with the assignment arcs. The right hand side is 1 for every mutual arc capacity constraint. This imposes a restriction on the total combined flow on the linked group of assignment arcs of the n commodities. Although the linking constraints are stated as inequalities, they are tight for every feasible solution to the problem (proved following the model).

3.6 Mathematical Programming Model

Let x^i_{sv} be the flow on arc $(s,v)^i$. From the set definitions, the integer, multicommodity, multiperiod, assignment problem (MCMAP) may be stated mathematically follows:

e. Continue with Step 3.

Step 3: Select Next Current Problem (CP).

If the Candidate List is empty, stop, the incumbent is the optimal solution.

Otherwise, choose the next problem from the candidate list to be the next current problem and continue with Step 4.

Step 4: Early Fathoming / LP Solution.

Remove the current problem from the candidate list or branch left, the person is not assigned the job. If the lower bound of the current problem is greater than or equal to Z*, remove it from the candidate list and return to Step 3.

Otherwise:

a. Relax the integrality constraints of CP to produce CP_r.
b. Solve CP_r.
c. Let Z be the value of the optimal objective function of CP_r.
d. Continue with Step 5.

Step 5: First Fathoming Criterion (Infeasibility Check).

If the current problem has no feasible solution, fathom it. Return toStep 3.

Otherwise, continue with Step 6.

Step 6: Second Fathoming Criterion (Compare to Incumbent).

The third fathoming criterion (Step 7), states that a subproblem is fathomed if its linear programming relaxation has an optimal integer basic feasible solution. It is sufficient to test for the integrality of flow through only the basic assignment arcs. By definition of the networks of LMCMAP, x^i_{sv}, the flow through the transfer

If Z is greater than or equal to Z*, fathom the current problem. Return to Step 3. Otherwise, continue with Step 7.

Step 7: Third Fathoming Criterion (Integrality Check).

If the linear programming relaxation of CP has an integer optimal feasible solution then,

a. Set Z* = Z.
b. Record the current problem's solution.
c. Return to Step 3.

Otherwise, return to Step 2.

4.2 General Remarks on the Algorithm with Regard to Implementation

The linear programming relaxation, LMCMAP, is the first candidate problem at the root node of the branch and bound tree. The arcs of the problem are stored in a single array where commodity 1 arcs are followed by commodity 2 arcs and so on. Let x^h be the flow assigned to the h^{th} arc in the arc array. A problem is separated into two candidate problems by fixing the value of a variable, say x^h_{sv}, to 0 or 1. A variable is fixed to zero by assigning a relatively large positive value to the corresponding objective function coefficient. A variable is fixed to one by assigning a negative value for which its absolute value is relatively large, to the corresponding objective function coefficient. The cost array is reset to its original values at the end of each iteration of the algorithm.

A depth first strategy was implemented. It is relatively straightforward to implement, and based on the computational results discussed in the next section, did not pose a massive combinatorial hurdle. Few branch and bound nodes sere evaluated.

Prob. No.	MAP: MCNF Reinv Freq.	Best Solution Found	% Within IP	MCNF LP Iter-ations	LP Solutions (B&B Nodes)	MCNF Reinvs	CPU Solution Time (Sec.)
1	100	2286	0.00%	264	1	2	1.90
2	100	2427	0.37%	295	1	2	2.21
3	100	4159	0.00%	479	0	4	5.65
4	25	568	1.61%	141	4	5	0.39
5	25	605	0.00%	139	2	5	0.38
6	100	694	0.00%	174	2	1	0.75
7	100	700	0.00%	144	0	1	0.48
8	100	737	0.55%	227	1	2	1.01
9	25	868	0.23%	217	1	8	0.82
10	25	856	0.00%	215	0	8	0.86
11	25	1088	0.00%	248	0	9	1.10
12	25	1118	0.00%	311	0	12	1.64
13	100	1223	0.00%	263	0	2	1.59
14	100	1362	0.00%	329	2	3	2.24
15	100	3439	0.38%	1040	5	11	21.70
16	100	3375	1.17%	1027	10	10	21.49
17	100	5708	0.23%	1565	9	15	52.49
18	25	373	0.00%	100	0	4	0.23
19	25	518	0.19%	178	2	7	0.63
20	25	642	0.00%	195	0	7	0.73
21	25	765	0.00%	369	1	14	1.96
22	100	260	0.00%	360	2	3	2.43
23	100	1527	0.00%	377	1	3	2.86
24	100	1863	0.92%	484	5	4	3.95
25	25	1045	1.85%	609	3	24	4.69
26	25	1206	1.60%	712	4	28	6.51
27	25	1296	0.00%	519	0	20	4.15
28	25	1510	0.00%	814	2	32	8.79
29	100	1688	0.42%	1082	6	10	15.49
30	100	4153	0.75%	4551	29	182	177.12
31	25	414	0.00%	752	14	30	3.48
32	25	595	0.00%	283	0	11	1.51
33	100	807	2.28%	709	3	7	6.36
34	100	745	0.00%	373	0	3	2.66
35	25	906	0.00%	789	3	31	7.65

Table 4: (Part 1) Summary of Computational Results of MAP on the IBM 3081-24 Using the VS FORTRAN 77 Compiler with the Optimization Level Set to 3. MAP and MCNF parameters: INTCHK=3, FBOUND=1.03, ISPAN=25, LB=10, UB=35.

	MAP						
Prob. No.	MCNF Reinv Freq.	Best Solution Found	% Within IP	MCNF LP Iter- ations	LP Solutions (B&B Nodes)	MCNF Reinvs	CPU Solution Time (Sec.)
36	25	1113	0.72%	2465	4	98	30.27
37	25	1247	1.88%	1611	12	64	20.56
38	25	1473	0.00%	1581	8	63	24.45
39	25	1648	0.10%	1762	4	70	29.44
40	25	1777	0.00%	2724	11	108	49.63
41	100	1977	1.91%	4875	9	48	103.16
42	DNR						
43	25	279	0.00%	117	0	4	0.45
44	25	487	0.00%	411	0	16	2.91
45	25	687	1.63%	1042	7	41	10.54
46	100	892	1.59%	1168	2	11	13.53
47	100	868	0.81%	2031	6	21	26.85
48	100	912	1.90%	1372	4	13	18.14
49	25	1121	2.19%	3739	19	149	59.85
50	DNR						
51	25	314	0.00%	162	2	6	0.81
52	25	559	2.01%	1726	12	69	16.13
53	25	784	0.00%	1988	8	79	28.22
54	25	986	0.00%	5422	11	216	100.90
55	25	1224	1.75%	9960	13	398	221.74
56	25	1478	0.96%	12565	17	506	331.47
57	25	1738	1.28%	11990	14	479	796.80
58	DNR						
59	25	328	0.00%	198	0	7	1.20
60	25	609	0.00%	645	1	25	7.16
61	25	885	1.14%	2155	5	86	37.70
62	25	1131	1.34%	3571	9	142	82.09
63	DNR						
64	25	407	0.99%	264	2	10	2.07
65	25	687	1.03%	1143	3	56	21.34
66	25	938	0.54%	3149	6	125	72.36
67	DNR						
68	25	426	0.00%	329	0	13	3.14
69	25	768	1.45%	6857	18	274	136.95
70	DNR						

DNR = Problem did not run

Table 4: (Part 2) Summary of Computational Results of MAP on the IBM 3081-24 Using the VS FORTRAN 77 Compiler with the Optimization Level Set to 3. MAP and MCNF parameters: INTCHK=3, FBOUND=1.03, ISPAN=25, LB=10, UB=35.

6. Summary and Conclusions

We presented a complete mathematical formulation of the integer multicommodity, multiperiod assignment problem formulation of Aronson [5]. Instead of following Aronson's technique of exploiting linked shortest path problems, we concentrated on exploiting the special, multicommodity structure of the linear programming relaxation of the problem. We developed and implemented a specialized branch and bound algorithm in which is embedded a modified version of MCNF, a multicommodity network flow code. Computational tests comparing our implementation to the commercial code MPSX/MIP/370 indicate that our methodology performs quite favorable. The range of the ratio of the solution CPU times of MAP to the solution CPU times of MPSX/MIP/370 for all problems solved to optimality by MAP was 0.19 to 2.85. MAP was faster in solving approximately 55% of the test problems.

A problems extension of the work on MAP includes constructing and experimenting with a specialized version of the primal partitioning multicommodity network flow algorithm implementation, MCNF. It is expected that the new method would perform much better because the majority of the solution time is spent in solving linear programming relaxations. Another modification that may be applied to the branch and bound algorithm is in the management of the candidate list.

Further enhancements would exploit the multiperiod structure by solving the subproblems in a forward manner [13, 41-43]. Others involve using fictitious bounds [44]. One may also use a heuristic, as in [5], to find an initial incumbent solution in Step lg of the Algorithm. Given this solution, an initial basis Subgradient optimization [33,27,11] may also be applied. In our companion papers, we discuss special properties of the model [45] and variations of the model for facility location and personnel planning [46,47].

REFERENCES

[1] Balas, E. and Saltzman, M.J., "**Facets of the Three-Index Assignment Polytope**", Discrete Applied Mathematic, 23, (1989), 201-229.

[2] Balas, E. and Saltzman, M. J., "**An Algorithm for the Three-Index Assignment Problem,**" Operations Research, prob. 39, 1 (forthcoming).

[3] Evans, J. R., "**The Multicommodity Assignment Problem: A Network Aggregation Heuristic,**" Computers and Mathematics with Applications, 7, 2, (1981), 187 - 194.

[4] Pierskalla, W. P., "**The Tri-Substitution Method for the Three-Dimensional Assignment Problem,**" CORS Journal, 5, (1967), 71 - 81.

[5] Aronson, J. E., "**The Multiperiod Assignment Problem: A Multicommodity Network Flow Model, a Specialized Branch and Bound Algorithm,**" European Journal of Operational Research, 23, (1986), 367 - 381.

[6] International Business Machines Corporation, "**IBM Mathematical Programming Systems Extended/370.**" White Plains, NY, 1979.

[7] Bazaraa, M. S. and Jarvis, J. J., **Linear Programming and Network Flows,** John Wiley and Sons, New York, NY, 1977.

[8] Gass, S. I., **Linear Programming: Methods and Applications**, 5th edition, McGraw-Hill, New York, NY, 1985.

[9] Simmonard, M., **Linear Programming**, Prentice-Hall, Englewood Cliffs, NJ, 1966.

[10] Kennington, J. L. and Helgason, R. V., **Algorithms for Network Programming**, John Wiley and Sons, Inc., New York NY 1980.

[11] Parker, R. G. and Rardin, R. L., **Discrete Optimization,** Academic Press, Inc., San Diego, CA, 1988.

[12] Salkin, H. M. and Mathur, K., **Foundation of Integer Programming,** North-Holland, New York, NY, 1989.

[13] Aronson, J. E., "**A Survey of Dynamic Network Flows,**" Annals of Operations Research (Shetty, B., ed.), 20, (1989), 1-66.

[14] Gilbert, K. C. and Hofstra, R. B., "**Multidimensional Assignment Problems,** " Decision Sciences, 19, 2, (1988), 306 - 321.

[15] Charnes, A., Cooper, W. W., Niehaus, R. J. and Stedry A., "**Static and Dynamic Assignment Models with Multiple Objectives, and Some Remarks on Organization design,**" Management Science, 15, 8, (1969), B365 - B375.

[16] Haley, K. B., "**The Multi-Index Problem,**" Operations Research, 11, (1963), 368-379.

[17] Haley, K. B., "**The Existence of a Solution to the Multi-Index Problem,**" Operational Research Quarterly, 16, (1965), 471 - 474.

[18] Little, J. D. C., Murty, K. G., Sweeney, D. W. and Karel, C., "**An Algorithm for the Traveling Salesman Problem,**" Operations Research, 11, 6, (1963).

[19] Pierskalla, W. P., "**The Multi-Dimensional Assignment and**

Quadratic Assignment Problems," Technical Memorandum No. 93, Case Western Reserve University, Operations Research Department, School of Management, Cleveland, OH, (1967).

[20] Pierskalla, W.P., "**The Multi-Dimensional Assignment Problem,**" Operations Research, 15, 2, (1968), 422-431.

[21] Schell, E., "**Distribution of a Product by Several Properties,**" Directorate of Management Analysis, Proceedings of the Second Symposium in Linear Programming, (Antosiewicz, H. ed.), 2, DCS/Comptroller H.Q. U.S. Air Force, Washington, DC, (1955), 615 - 642.

[22] Shamma, M. M., **A Generalized Assignment Problem**, Unpublished Ph.D. Dissertation, Computer Science/Operations Research Center, Southern Methodist University, Dallas, TX, 1971.

[23] Vlach, M., "**Branch and Bound Method for the Three Index Assignment Problem,**" Ekonomicko-Matematicky Obzor (Czechoslovakia), 3, (1967), 181 - 191.

[24] Kennington, J. L., "**A Survey of Linear Cost Multicommodity Network Flows,**" Operations Research, 26, 2, (1978), 209 - 236.

[25] Dantzig, G. B. and Van Slyke, R. M., "**Generalized Upper Bounding Techniques,**" Journal of Computer and System Sciences, 1, (1967), 213-226.

[26] Evans, J. R., **Theoretical and Computational Aspects of Integer Multicommodity Network Flow Problems,** Unpublished Ph.D. Dissertation, Department of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 1975.

[27] Evans, J. R., "**A Combinatorial Equivalence Between a Class of Multicommodity Flow Problems and the Capacitated Transportation Problem,**" Mathematical Programming, 10, 3, (1976), 401 - 404.

[28] Evans, J. R., "A **Single Commodity Transformation for Certain Multicommodity Networks,**" Operations Research, 26, 4, (1978), 673 - 681.

[29] Evans, J. R., "**The Simplex Method for Integral Multicommodity Networks,**" Naval Research Logistics Quarterly, 25, 1, (1978), 31 - 38.

[30] Evans, J. R. and Jarvis, J. J., "**Network Topology and Integral Multicommodity Flow Problems,**" Networks, 8,2, (1978), 107 - 120/

[31] Sakarovitch, M., "**Two Commodity Network Flows and Linear Programming,**" Mathematical Programming, 4, (1973), 1 - 20.

[32] Telgen, J., "**Identifying Redundant Constraints and Implicit Equalities in Systems of Linear Constraints,**" Management Science, 29, 10, (1983), 1209 - 1221.

[33] Bean J. C., "**A Lagrangian Algorithm for the Multiple Choice Integer Program,**" Operations Research, 32, 5, (1984), 1185 - 1193.

[34] Geoffrion, A. M. and Marsten, R. E., "**Integer Programming Algorithms: A Framework and State-of-the-Art Survey,**" Management Science, 18, 9, (1972), 465 - 49.

[35] Balinski, M. L., "**Integer Programming Methods, Uses, Computation,**" Management Science, 12, 3, (1965), 253-313.

[36] Balinski, M. L. and Spielberg, K., "**Methods for Integer Programming: Algebraic, Combinatorial and Enumerative,**" in J. S. Aronofsky (editor), Progress in Operations Research, 3, Wiley, New York, NY (1969).

[37] Beale, E., "**Survey of Integer Programming,**" Operations Research Quarterly, 16, 2, (1965), 219 - 228.

[38] Geoffrion, A. M., "**Integer Programming by Implicit Enumeration and Balas' Method,**" SIAM review, 9, 2, (1967), 178 - 190.

[39] Geoffrion, A. M., "**An Implicit Enumeration Approach for Integer Programming,**" Operations Research, 17, 3, (1969), 437 - 454.

[40] Kennington, J. L., "A **Primal Partitioning Code for Solving Multicommodity Network Flow Problems (Version 1),**" Technical Report OR 79008, Department of Operations Research, Southern Methodist University, Dallas, TX 1979.

[41] Aronson, J. E. and Chen, B. D., "**A Forward Network Simplex Algorithm for Solving Multiperiod Network Flow Problems,**" Naval Research Logistics Quarterly, 33, (1986), 445-467.

[42] Aronson, J. E., Morton, T. E. and Thompson, G. L., "A **Forward Simplex Method for Staircases Linear Programming Problems,**" Management Science, 31, 6, (1985), 664-679.

[43] Aronson, J. E. and Thompson, G. L., "**A Survey of Forward Methods in Mathematical Programming,**" Large Scale Systems, (1984), 1 - 16.

[44] Bazaraa, M. R. and Elshafei, A. N., "**On the Use of Fictitious Bounds in Tree Search Algorithms,**" Management Science, 23, 8, (1977), 904-908.

[45] Elnidani, M. A. and Aronson, J. E., "**The Multicommodity, Multiperiod Assignment Problem II: Theoretical Results,**" Working Paper 89-277, Department of Management Sciences and Information Technology, College of Business Administration, The University of Ge

[46] Elnidani, M. A. and Aronson, J. E., "**The Multicommodity Multiperiod Assignment Problem III: Variations for Facility Location and Personnel Planning**" Working Paper 89 - 278, Department of Management Sciences and Information Technology, College of Business A

[47] Elnidani, M. A. **The Multiperiod Assignment Problem**, Doctoral Dissertation, Department of Operations Research and Engineering Management, Southern Methodist University, Dallas, TX, 1986.

APPENDIX: Proof of the Proposition

Proposition:

The assignment arc mutual capacity constraints of LMCMAP (8) are implicit equality constraints.

Proof:

Assume contradicting the desired result, that there exists a feasible solution (not necessarily basic) such that at least one constraint from (8) is not tight. Let that constraint be the one associated with assignment arcs (s,v). By the assumption:

$$\sum_{i=1}^{n} x_{sv}^{i} < 1 \qquad \text{(A.1)}$$

holds. Let t be the time period which corresponds to the assignment arcs (s,v), i.e. $s \in N^t_B$ and $v \in N^t_E$. To satisfy the flow conservation constraints, (A.2) must be true:

$$\sum_{i=1}^{n} \left[\sum_{(e,f) \in E_A} x_{ef}^{i} \right] = n \ , \qquad \text{(A.2)}$$

where $e \in N_B^t$, $f \in N_E^t$.

I. e., the sum of the total combined flow on all assignment arcs having their origin nodes in the set N^t_B and their destination nodes in the set N^t_E must equal n, because the directed acyclic networks have a total supply and demand of n. There are n such assignment arcs (s,v) of the n commodities in period t, i.e.

$$y = \sum_{i=1}^{n} x_{sv}^{i} \ . \qquad \text{(A.3)}$$

Therefore, n-y units must flow through the remaining n- 1 unassigned assignment arcs of the n commodities in period t. By the nonnegativity constraints (9), (A.1) and (A.3)

$$0 \leq y < 1 \tag{A.4}$$

holds. Therefore,

$$n \geq n-y > n-1 \tag{A.5}$$

holds, implying that there is at least one set of assignment arcs, say (e,f) of the n commodities, for which the arcs are assigned a total combined flow that is strictly greater than 1. Therefore, the solution cannot be feasible. Thus, the mutual arc capacity constraints are tight for all feasible solutions to the problem.

Q.E.D.