BIPARTITE GRAPHS AND AN OPTIMAL BORDERED TRIANGULAR FORM OF A MATRIX

by

A. Sangiovanni-Vincentelli and T.A. Bickart

Memorandum No. UCB/ERL M78/74

22 July 1977

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

BIPARTITE GRAPHS AND AN OPTIMAL BORDERED TRIANGULAR FORM OF A MATRIX

Alberto Sangiovanni-Vincentelli and Theodore A. Bickart†

Department of Electrical Engineering and Computer Sciences
and the Electronics Research Laboratory
University of California, Berkeley, California 94720

ABSTRACT

The problem of determining row and column permutations to transform a nonsingular (not necessarily symmetric) matrix to a minimum k k-bordered lower triangular form is shown to be an NP-complete (intrinsically difficult) problem by treating an equivalent bipartite graph problem — determine a minimum essential dumbbell set. A (sequential, rather than backtrack oriented) algorithm is described by which to obtain a minimal (local minimum, rather than minimum) essential dumbbell set, hence, also a minimal k k-bordered lower triangular form of a matrix. The performance of an APL realization of the algorithm is illustrated and data to justify an embedded heuristic is provided.

Research sponsored in part by the National Science Foundation under Grant ENG74-06651-A02 and by a special grant from the Syracuse University Senate Research Committee to the second author.

†On leave from the Department of Electrical and Computer Engineering, Syracuse University, Syracuse, New York 13210.

I. INTRODUCTION

In the analysis of a large scale system, such as an electrical network, it is usually advantageous to take account of such a system's structure. This structure is manifested by the equations describing the mathematical model of the physical system and can (usually) be exploited by a reordering of the equations and variables. One structure, made manifest as a k-bordered lower triangular Jacobian matrix of the reordered equations, has been recently exploited.† This is the case in algorithms developed to solve sets of nonlinear equations [1], to solve linear programming problems [2], (with tearing methodologies) to solve linear algebraic system equations [3,4], and to study input-output stability of interconnected systems [5].

In general, many different k-bordered lower triangular forms $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ of a matrix $\underset{\sim}{A}$ exist, depending upon the row permutation — matrix $\underset{\sim}{P}$ — and the column permutation — matrix $\underset{\sim}{Q}$ — selected. Of note, though, is: In all the applications just mentioned, the solution process becomes increasingly efficient as k decreases. By this we then infer the following optimization problem [4]: *Given the matrix* $\underset{\sim}{A}$ *determine permutation matrices* $\underset{\sim}{P}$ *and* $\underset{\sim}{Q}$ *such that* $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ *is a k-bordered lower triangular matrix with minimum k.* As a constraint on the problem, we would, as has been necessary in all of the above noted applications, impose: *The matrix* $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ *must have non-zero diagonal elements.* This now constrained optimization problem has been investigated

---

†A square matrix $\underset{\sim}{A} = [a_{ij}]$ of order n is *k-bordered lower triangular* (k-BLT) if $a_{ij} = 0$ $(i = 1,\ldots,n-k \quad (j = i+1,\ldots,n-k))$. See Fig. 1.

under the assumption that $\underset{\sim}{Q} = \underset{\sim}{P}'$ — symmetric permutations on $\underset{\sim}{A}$ — with some graph theoretic algorithms for establishing $\underset{\sim}{P}$ having been given [1,4,6,7,8].† Without this assumption this problem stands unresolved — an open question [9] — with but few applicable results.

In [10], the first author proposed a graph theoretic interpretation of the problem and a resolution of it, based on directed graphs and simple operations on them. Some heuristic algorithms were subsequently proposed in [11]. In this paper, we will provide a careful interpretation and treatment of this problem based on bipartite graph theory, culminating with presentation of a rigorously justified heuristic algorithm.

The organization of the paper is as follows: Some graph theoretic terms are defined and the correspondence between matrices and bipartite graphs is shown in Section II; the equivalence of the task of establishing an optimum — minimum k — solution of the (non-symmetric permutation) problem to the task of obtaining a dumbbell — graph theoretic term to be defined — set of minimum cardinality is established in Section III; a backtrack algorithm by which to accomplish the latter task is produced in Section IV and then there shown to be intrinsically hard (NP-complete); a heuristic algorithm to invoke in place of and exhibiting greater efficiency than the backtrack algorithm is described and validated in Section V; illustrations of the heuristic algorithm, realized as a collection of *APL* functions (listed in the Appendix), are given in Section VI; the concluding discussion is the subject of Section VII.

---

† See also the papers cited in these referenced papers, especially the citation of [8].

II. DEFINITIONS

Any graph theoretic terms not hereafter defined are to be understood as defined by Harary [12]. Let $G = (X,U)$ be a *graph* [alternatively, $G = (X,E)$ be a *digraph*] with a set of *nodes* X and a set of *edges* $U = \{\{x_i,x_j\} : x_i,x_j \in X\}$ [alternatively, a set of *directed edges* $E = \{(x_i,x_j) : x_i,x_j \in X\}$]. A *simple path* [alternatively, *simple directed path*] $\mu(x_i,x_j)$ is a sequence of distinct nodes — thus, $x_i \neq x_j$ — denoted $\langle p_0,\ldots,p_\ell \rangle$ such that $p_0 = x_i$, $p_\ell = x_j$, and $\{p_{k-1},p_k\} \in U$ $(k = 1,\ldots,\ell)$ [alternatively, and $(p_{k-1},p_k) \in E$ $(k = 1,\ldots,\ell)$]. The path is said to be of *length* ℓ. A *simple cycle* [alternatively, a *simple directed cycle*] η is a sequence of nodes $\langle p_0,\ldots,p_{\ell-1},p_0 \rangle$ such that $\langle p_0,\ldots,p_{\ell-1} \rangle$ and $\langle p_1,\ldots,p_0 \rangle$ are simple paths [alternatively, simple directed paths]. The cycle is said to be of length ℓ. A simple [alternatively, a simple directed] path or cycle is said to contain an edge if that edge is $\{p_{k-1},p_k\}$ [alternatively, (p_{k-1},p_k)] for some $k = 1,\ldots,\ell$, with p_ℓ interpreted as p_0 for a cycle. The *section graph* [alternatively, *section digraph*] defined on the node set $Y \subset X$ is $G(Y) \triangleq (Y,U(Y))$ where $U(Y) = \{\{x_i,x_j\}: \{x_i,x_j\} \in U \wedge x_i,x_j \in Y\}$ [alternatively, $G(Y) \triangleq (Y,E(Y))$ where $E(Y) = \{(x_i,x_j): (x_i,x_j) \in E \wedge x_i,x_j \in Y\}$].

A directed graph $G = (X,E)$ is said to be strongly connected if, for every pair of vertices $x_i,x_j \in X$, there exist a simple path $\mu_1(x_i,x_j)$ and a simple path $\mu_2(x_j,x_i)$. Note: A trivial digraph — with just one node — is considered to be strongly connected. Let

$\pi = \{X_1,\ldots,X_q\}$ be a partition of the nodes X. If the section digraph $G_i = G(X_i)$ $(i = 1,\ldots,q)$ is strongly connected and is not a proper subgraph of some strongly connected section digraph of G, then the G_i's are called the *strongly connected components* of G.

A *bipartite graph* B = (S,T,U) is a graph B = (X,U) [alternatively, a *bipartite digraph* B = (S,T,E) is a digraph B = (X,E)] such that $X = S \cup T$, [alternately, $E = S \cup T$], $S \cap T = \phi$, and the section graphs [alternatively, digraphs] B(S) and B(T) are both *node graphs* — edge free graphs. A set of edges $I \subset U$ of a bipartite graph B = (S,T,U) [alternatively, $\subset E$ of a bipartite digraph B = (S,T,E)] is said to be a *matching* if no two edges of I are incident at the same node. A node is said to be *covered* if an edge of I is incident at it. A *complete matching* is a matching such that all nodes are covered. A *maximum cardinality matching* is a matching having a maximum number of edges. Note: A bipartite graph may not have a complete matching, but it always has a maximum cardinality matching; on the other hand, a complete matching is a maximum cardinality matching. These several categories of matchings are illustrated in Fig. 2. A simple path — node sequence $\langle p_0,\ldots,p_\ell\rangle$ — is said to be a *simple alternating path wrt I* (with respect to I), denoted $\lambda_I(x_i,x_j)$, if $\{p_{k-1},p_k\} \in I$ for k odd or for k even. A simple cycle — node sequence $\langle p_0,\ldots,p_{\ell-1},p_0\rangle$ — is said to be a *simple alternating cycle wrt I*, denoted ρ_I, if $\langle p_0,\ldots,p_{\ell-1}\rangle$ and $\langle p_1,\ldots,p_{\ell-1},p_0\rangle$ are alternating simple paths wrt I. Clearly the length of an alternating cycle must be even. If a bipartite graph B has no alternating simple cycles with respect to some matching I (of B), then B is said to be *acyclic wrt I*.

Graphs have often been used to represent the zero/non-zero element structure of matrices. In this paper we will do likewise for square matrices using bipartite graphs. The relationship of a bipartite graph to a matrix $\underset{\sim}{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$ is quite simple: Let[†] $B[\underset{\sim}{A}] = (S[\underset{\sim}{A}], T[\underset{\sim}{A}], U[\underset{\sim}{A}])$, denoting the bipartite graph associated with $\underset{\sim}{A}$, be defined by the conditions[††] $|S[\underset{\sim}{A}]| = |T[\underset{\sim}{A}]| = n$ and $\{s_i, t_j\} \in U[A]$ iff[†††] $a_{ij} \neq 0$ $(i,j = 1,\ldots,n)$. Now, bipartite graphs are particularly well matched to the task of establishing a non-symmetric permutation strategy. In particular, a non-symmetric permutation on $\underset{\sim}{A}$, realizing $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$, has as its effect the separate reordering of the rows and columns of $\underset{\sim}{A}$. Thus, by the definitions of $B[\underset{\sim}{A}]$ and $B[\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}]$ it follows — because $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ is but a row and column reordered copy of $\underset{\sim}{A}$ — that $B[\underset{\sim}{A}]$ and $B[\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}]$ are isomorphic. The structural properties are maintained. Moreover, it must be noted that the existence of a complete matching $I[\underset{\sim}{A}]$ in $B[\underset{\sim}{A}]$ is equivalent to the existence of non-zero elements of $\underset{\sim}{A}$ which can be brought by row and column permutations to the main diagonal of $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$. Thus, I characterizes a "coupling" between rows — corresponding to the S set — and columns — corresponding to the T set — of $\underset{\sim}{A}$.

Throughout the paper it will be necessary to consider, in addition to an edge in I, the nodes at which the edge is incident. This is a special section graph — of a bipartite graph $B = (S,T,U)$ — which we call a *dumbbell*. In particular, let $s \in S$ and $t \in T$; then the section graph defined on these two nodes is a dumbbell, denoted s—t, if $\{s,t\} \in U$. Now, given a bipartite graph B and a complete matching I, the

[†]When there can be no confusion in doing so $[\underset{\sim}{A}]$ will not be appended to B, S, etc.

[††]The symbol for a set enclosed by vertical rules — e.g., $|S[\underset{\sim}{A}]|$ — denotes the cardinality — number of elements — of the set.

[†††]Throughout, iff denotes if and only if.

fundamental dumbbell set wrt I, denoted D_I, consists of the dumbbells associated with the edges of I. Given a bipartite graph B = (S,T,U), a set of dumbbells D, and a dumbbell $\hat{d} = \hat{s}\text{—}\hat{t} \in D$, the set of *dumbbells adjacent to* $\hat{d}$ *in* D *wrt* S, denoted $SA_D(\hat{d})$, is

$$SA_D(\hat{d}) = \{d\colon d = s\text{—}t \in D - \{\hat{d}\} \wedge t = \hat{t}\}$$

and the set of *dumbbells adjacent to* $\hat{d}$ *in* D *wrt* T, denoted $TA_D(\hat{d})$, is

$$TA_D(\hat{d}) = \{d\colon d = s\text{—}t \in D - \{\hat{d}\} \wedge s = \hat{s}\}$$

The *S-degree* of $\hat{d}$ denoted $S_D^o(\hat{d})$, is the cardinality — $|SA_D(\hat{d})|$ — of the set $SA_D(\hat{d})$, and the *T-degree* of $\hat{d}$, denoted $T_D^o(\hat{d})$, is the cardinality — $|TA_D(\hat{d})|$ — of the set $TA_D(\hat{d})$. A set of dumbbells D distinguishes the following sets of nodes:

$$S(D) = \{s\colon s \in S \wedge \exists\ s\text{—}t \in D\},$$

$$T(D) = \{t\colon t \in T \wedge \exists\ s\text{—}t \in D\},$$

and

$$X(D) = S(D) \cup T(D).$$

Let B be a bipartite graph with a complete matching, then a set of dumbbells F is an *essential dumbbell set* of B if the section graph B(Y), where Y = X-X(F), admits a complete matching and is acyclic with respect to that matching. An essential dumbbell set of minimum cardinality, denoted MF, is called a *minimum essential dumbbell set* (of B). These several constructs associated with dumbbells are illustrated in Fig. 3. Each of these dumbbell related concepts associated with a bipartite graph have their counterparts when a bipartite digraph is considered. In the definitions the word *directed* would be added

and U would be replaced by E. Notationally, a directed dumbbell is denoted as $s \to t$ (or $t \to s$, as the case might be).

III. OPTIMUM BORDERED (LOWER) TRIANGULAR FORM AND BIPARTITE GRAPHS

The graph theoretic interpretation of the constrained optimization problem will emerge as a consequence of the following propositions and lemmas. We start with

PROPOSITION 3.1 *Suppose a bipartite graph B has two complete matchings I_a and I_b. Then there exists a set of disjoint simple alternating cycles wrt I_a and wrt I_b $\{\rho^i$ $(i = 1,\ldots,m)\}$ such that $(I_a-\hat{I}) \cup (I_b-\hat{I})$ is the edge set associated with $\{\rho^i\}$, where $\hat{I} = I_a \cap I_b$.*

PROOF: This proposition is an immediate consequence of a theorem in [13, p.123]. ¤

The relationship between a pair of complete matchings specified by this proposition is illustrated in Fig. 4. We now present

LEMMA 3.1 *Let B be a bipartite graph with the complete matching I. Then the following statements are equivalent:*

(a) I is unique.

(b) B is acyclic wrt I.

PROOF This lemma is an immediate consequence of Proposition 3.1. ¤

The next result we seek to establish is expressed in

LEMMA 3.2 *Let B be a bipartite graph with complete matching I and let* D_I *be its fundamental dumbbell set wrt I. Then, if B is acyclic wrt I, there exists a dumbbell* $d_\tau \in D_I$ *such that* $SA_{D_I}(d_\tau) = \phi$ *and a dumbbell* $d_\sigma \in D_I$ *such that* $TA_{D_I}(d_\sigma) = \phi$.

The proof of this lemma rests upon

CONSTRUCTION 3.1 *Let* $B = (S,T,U)$ *be a bipartite graph with the complete matching I. Let* $\bar{B}^I = (S,T,E^I)$ *be the derived bipartite digraph wherein*

$$(s,t) \in E^I \text{ iff } \{s,t\} \in U \wedge \{s,t\} \notin I$$

and

$$(t,s) \in E^I \text{ iff } \{s,t\} \in U \wedge \{s,t\} \in I$$

and upon

PROPOSITION 3.2 *Let B be a bipartite graph with the complete matching I and let* $\bar{B}^I$ *be the bipartite digraph derived according to Construction 3.1. Then, there is one-to-one correspondence between simple directed cycles of* $\bar{B}^I$ *and simple alternating cycles wrt I of B.*

PROOF: This proposition is an obvious consequence of the definitions of simple directed and simple alternating cycles. ¤

Because of the correspondence established by this theorem, we may refer to a bipartite digraph which has no simple directed cycles as acyclic.

We now return to

PROOF: [Lemma 3.2] Create $\bar{B}^I$ according to Construction 3.1. Because B is acyclic wrt I (by hypothesis), it follows (by Proposition 3.2) that $\bar{B}^I$ has no simple directed cycles. Hence, by a well-known theorem (see Proposition 6.2 in [14, p.29]), there is a node which has 0 out-degree. By Construction 3.1 this node must be in S; denote it by s_σ. Similarly, there is a node which has 0 in-degree that must be in T; denote it by t_τ.

Clearly, there is only one edge, denoted (t_σ, s_σ), incident at s_σ and only one edge, denoted (t_τ, s_τ), incident at t_τ. By the correspondence between edges of $\bar{B}^I$ and B and by the fact — established by hypothesis — that I is a complete matching the dumbbell $d_\sigma = s_\sigma - t_\sigma \in D_I$ is such that $TA_{D_I}(d_\sigma) = \phi$. Similarly, the dumbbell $d_\tau = s_\tau - t_\tau \in D_I$ is such that $SA_{D_I}(d_\tau) = \phi$. ¤

The next result we need is set forth in

LEMMA 3.3 *Let B be a bipartite graph with the complete matching I. Then, B is acyclic wrt I iff it is possible to order the fundamental dumbbell set D_I as a sequence $\langle d_1, \ldots, d_n \rangle$ such that $d_i \in TA_{D_I}(d_j)$ implies $i < j$.*

PROOF: [*only if*] Set $B^0 = B$ and $I^0 = I$. (Obviously, $D_{I^0} = D_I$.) By Lemma 3.2 there exists a $\hat{d} \in D_{I^0}$ such that $TA_{D_{I^0}}(\hat{d}) = \phi$. Set $d_1 = \hat{d}$. It follows that $\{D_I - \{d_1\} \cap TA_{D_{I^0}}(d_1) = \phi$ and (hence) that $\{D_I - \{d_1\}\} \cap TA_{D_I}(d_1) = \phi$. Derive B^1 from B^0 and I^1 from I^0 by deteleting d_1. Clearly, I^1 is a complete matching for B^1 and B^1 is acyclic wrt I^1. Again by Lemma 3.2 — here applied to B^1 with I^1 — there exists a $d \in D_{I^1}$ such that $TA_{D_{I^1}}(\hat{d}) = \phi$. Set $d_2 = \hat{d}$. It follows that $\{D_I - \{d_1, d_2\}\} \cap TA_{D_{I^1}}(d_2) = \phi$ and (hence) that $\{D_I - \{d_1, d_2\}\} \cap TA_{D_I}(d_2) = \phi$. Continuing in this manner a complete ordering of the dumbbells of D_I is obtained such that $\{D_I - \{d_1, \ldots, d_k\}\} \cap TA_{D_I}(d_k) = \phi$. Equivalently, $d_i \in TA_{D_I}(d_j)$ implies $i < j$. [*if*] Again, set $B^0 = B$ and $I^0 = I$. By $d_i \in TA_{D_{I^0}}(d_j)$ implies $i < j$ $(j, i = 1, \ldots, n)$ we infer that $TA_{D_{I^0}}(d_1) = \phi$.

Thus, the vertices of d_1 cannot be in a simple alternating cycle. It then follows that any simple alternating cycles of B^0 remain as simple alternating cycles of B^1, derived from B^0 by deleting d_1. Similarly, I^1 is derived I^0 by deleting d_1. Clearly, I^1 is a complete matching for B^1. Furthermore, by $d_i \in TA_{D_{I^0}}(d_j)$ implies $i < j$ $(i,j = 1,\ldots,n)$ we deduce that $d_i \in TA_{D_{I^1}}(d_j)$ implies $i < j$ $(i,j = 2,\ldots,n)$. Continuing in this manner — next for B^1 — we arrive at the bipartite graph B^{n-1} consisting of just d_n. This graph is obviously acyclic. Hence, B must also be acyclic. ¤

The last of the preliminary results we need is stated in

LEMMA 3.4 *A matrix* $\underset{\sim}{A} \in \mathbb{R}^{n\times n}$ *can be transformed by row and column permutations to a lower triangular matrix*† *with non-zero diagonal elements iff* $B[\underset{\sim}{A}]$ *has a complete matching I and is acyclic wrt I.*

PROOF: [*if*] By Lemma 3.3, order D_I as a sequence $\langle d_1,\ldots,d_n\rangle$ such that $d_i \in TA_{D_I}(d_j)$ implies $i < j$. Let $d_k = s_k - t_k$. Now there always exist permutation matrices $\underset{\sim}{P}$ and $\underset{\sim}{Q}$ such that row s_i $(i = 1,\ldots,n)$ of $\underset{\sim}{A}$ becomes row i of $\underset{\sim}{P}\underset{\sim}{A}$ and column t_j $(j = 1,\ldots,n)$ of $\underset{\sim}{P}\underset{\sim}{A}$ becomes column j of $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$. Becuase each dumbbell of D_I corresponds to a non-zero diagonal element of $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$, it follows, by $d_i \in TA_{D_I}(d_j)$ implies $i < j$, that $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ is lower triangular. [*only if*] This *only if* portion of the proof can be obtained by reversing the arguments and steps of the preceding *if* portion. ¤

Let us observe that, by Lemma 3.1, the last part of this lemma could have read ...*has a unique complete matching.*

---

†A lower triangular matrix is a 0-bordered lower triangular matrix.

We now present the principal result of this section as

THEOREM 3.1 *The determination of a transformation of* $\underset{\sim}{A} \in \mathbb{R}^{n\times n}$ *by row and column permutations — $\underset{\sim}{A}$ into $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ — such that $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ is k-BLT with minimum k and with non-zero diagonal elements is equivalent to (⇔) the determination of a minimum essential dumbbell set MF of* $B[\underset{\sim}{A}]$.

PROOF [⇒] Suppose there exist permutation matrices $\underset{\sim}{P}$ and $\underset{\sim}{Q}$ such $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q} = \hat{A}$ is k-BLT with minimum k and with non-zero diagonal elements. Then

$$\hat{\underset{\sim}{A}} = \begin{bmatrix} \hat{\underset{\sim}{A}}_{11} & \hat{\underset{\sim}{A}}_{12} \\ \hat{\underset{\sim}{A}}_{21} & \hat{\underset{\sim}{A}}_{22} \end{bmatrix},$$

where $\hat{\underset{\sim}{A}}_{11}$ is lower triangular and order n-k. By Lemma 3.4, $B[\hat{\underset{\sim}{A}}_{11}]$ has a complete matching $I[\hat{\underset{\sim}{A}}_{11}]$ and is acyclic wrt $I[\hat{\underset{\sim}{A}}_{11}]$. Furthermore, because $\hat{\underset{\sim}{A}}_{22}$ has non-zero diagonal elements $B[\hat{\underset{\sim}{A}}_{22}]$ has a complete matching $I[\hat{\underset{\sim}{A}}_{22}]$ corresponding to those diagonal elements. (Obviously, $I[\hat{\underset{\sim}{A}}] = I[\hat{\underset{\sim}{A}}_{11}] \cup I[\hat{\underset{\sim}{A}}_{22}]$ is a complete matching for $B[\hat{\underset{\sim}{A}}]$.) Now, $B[\hat{\underset{\sim}{A}}_{11}] = B[\hat{\underset{\sim}{A}}](X-X(D_{I[\hat{\underset{\sim}{A}}_{22}]}))$. Therefore, $D_{I[\hat{\underset{\sim}{A}}_{22}]}$ is an essential dumbbell set of $B[\hat{\underset{\sim}{A}}]$ and, as $\hat{\underset{\sim}{A}}$ is of minimum order (by hypothesis), it is a minimum essential dumbbell set of $B[\hat{\underset{\sim}{A}}]$. Since $B[\underset{\sim}{A}]$ and $B[\hat{\underset{\sim}{A}}]$ are isomorphic, there exists a corresponding minimum essential dumbbell set MF of $B[\underset{\sim}{A}]$. [⇐] Suppose there exists a minimum essential dumbbell set MF of $B[\underset{\sim}{A}]$. It also then follows that $B[\underset{\sim}{A}]$ has a complete matching (containing MF). So, there exist row and column permutations — corresponding to relabelings of the vertices of S and T — such that the diagonal elements of the transformed $\underset{\sim}{A}$, expressed as $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$, are non-zero and such that those associated with the dumbbells of MF occupy the last $k = |MF|$ positions. By the definition of a minimum essential dumbbell set it

also follows that $B[\underset{\sim}{A}](X-X(MF))$ has a complete matching and is acyclic. Thus, by Lemma 3.4, the above noted permutations can be chosen such that the first n-k rows and columns are a lower triangular submatrix of $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$. Therefore, $\underset{\sim}{P}\underset{\sim}{A}\underset{\sim}{Q}$ is k-BLT with — because MF (by definition) is of minimum cardinality — minimum k. ¤

IV. BACKTRACK ALGORITHM FOR MINIMUM ESSENTIAL DUMBBELL SET: VALIDATION AND COMPUTATIONAL COMPLEXITY

By Theorem 3.1 we have deduced from the constrained optimization problem the following equivalent problem: *Given the bipartite graph B, determine a minimum essential dumbbell set MF (of B).* For the solution of this problem we shall offer our backtrack (search) algorithm shortly.

In a search for a minimum essential dumbbell set it is necessary to preclude, insofar as is possible, dead-end searches. Note: Dead-end searches are made highly likely by the need to also establish a complete matching. Therefore, we must exploit the relationship between a complete matching and dumbbells to avoid dead-end searches. To that end, we will invoke the criterion: The dumbbells considered at each step of the search must belong to a set of dumbbells whose edges can be included in a complete matching. Now, such dumbbells are identified through Proposition 3.1. We shall, however, turn to an alternate means — one that appears computationally more attractive — of identifying them. By invoking Proposition 3.2, it is possible to prove, as in [15].

THEOREM 4.1 *Suppose the bipartite graph $B = (S,T,U)$ has a complete matching I_a. Consider the edge $\{\hat{s},\hat{t}\} \notin I_a$. Now, there exists a complete matching I_b such that $\{\hat{s},\hat{t}\} \in I_b$ iff $(\hat{s},\hat{t})$ belongs to a strongly connected component of $\bar{B}^{I_a}$, the bipartite digraph derived from B by Construction 3.1.*

The backtrack search algorithm we shall propose is based upon this theorem. Before doing so, however, it is convenient to the wording of the algorithm to alter the category of components said to be strongly connected in a trivial sense. Thus, if $\hat{s} \in S$ and $\hat{t} \in T$ are trivial strongly connected components of $\bar{B}^I$ and $(\hat{t},\hat{s}) \in E^I$, then $d = \hat{t} \to \hat{s}$ is said to be a strongly connected component in a trivial sense.

We now present the

BACKTRACK (SEARCH) ALGORITHM †

COMMENT $\bar{B}^I = (S,T,E)$ is assumed to be strongly connected and non-trivial ($E \neq \phi$).

BEGIN Make the assignments $\bar{B}_1^0 = (S_1^0,T_1^0,E_1^0) \leftarrow \bar{B}^I = (S,T,E)$, $D_1^0 \leftarrow \phi$ (D_1^0 is a set of directed dumbbells, initially empty), $MF \leftarrow \phi$, $\alpha \leftarrow \langle \bar{B}_1^0 \rangle$ (α is an ordered set — stack — initialized with $\bar{B}_1^0$), $j_0 \leftarrow 1$, $j_1 \leftarrow \ldots \leftarrow j_n \leftarrow 0$, $k \leftarrow \infty$, and $\hat{k} \leftarrow \infty$.

{1} REPEAT Let $\bar{B}_{j_K}^K$ be the first element in the stack α. Make the assignments $k \leftarrow K$.

IF $k + 1 \geq \hat{k}$

{2} THEN Delete $\bar{B}_{j_k}^k$ from α

ELSE

{3} REPEAT Determine the strongly connected components of $\bar{B}_{j_k}^k$. Let $TSC_{j_k}^k$ denote the trivial strongly connected components — a set of dumbbells. Make the assignment $\bar{B}_{j_k}^k \leftarrow \bar{B}_{j_k}^k - TSC_{j_k}^k$. Establish p, the number of strongly connected components of $\bar{B}_{j_k}^k$

IF $p + k \geq \hat{k}$

---

†This algorithm, and that to follow, follow the conventions set forth by Wirth [20]. In addition, line reference numbers are placed at the left margin within braces, as {1} for line 1.

{4} THEN Delete $\bar{B}^k_{j_k}$ from α. Make the assignment $k \leftarrow \infty$.

{5} ELSE Let $\{d_1,\ldots,d_{\ell^k_{j_k}}\}$ be the set of directed dumbbells associated with $\bar{B}^k_{j_k}$, one for each edge of $\bar{B}^k_{j_k}$. Make the assignments $n \leftarrow 1$ and $k \leftarrow k + 1$

REPEAT Make the assignments $j_k \leftarrow j_k + 1$ and $D^k_{j_k} \leftarrow D^{k-1}_{j_{k-1}} \cup \{d_n\}$.

IF $D^k_{j_k} = D^k_m$ for some $m = 1,\ldots,j_k-1$

THEN Put $\bar{B}^k_{j_k}$ on top of α. Make the assignment $n \leftarrow n + 1$.

ELSE

IF $d_n \neq t_n \rightarrow s_n$

THEN Find a path $\mu(t_n,s_n)$.

Reverse the edges of $\mu(t_n,s_n)$ in $\bar{B}^{k-1}_{j_{k-1}}$ and the edge $\{s_n,t_n\}$ in $\bar{B}^{k-1}_{j_{k-1}}$ and d_n.

{6} BEGIN Make the assignment $\bar{B}^k_{j_k} \leftarrow \bar{B}^{k-1}_{j_{k-1}} - d_n$.

END

IF $\bar{B}^k_{j_k}$ is acyclic.

{7} THEN Make the assignments $\hat{k} \leftarrow k$ and $MF \leftarrow D^k_{j_k}$.

ELSE

{8} IF $k + 1 < \hat{k}$

THEN Put $\bar{B}^k_{j_k}$ on top of α. Make the assignment $n \leftarrow n + 1$.

UNTIL $n > \ell^k_{j_k}$ or $k + 1 \geq \hat{k}$

UNTIL $k + 1 \geq \hat{k}$

UNTIL $\alpha = \phi$ or $\hat{k} = 1$ (MF is, this case, a minimum essential dumbbell set of B.)

END

The validity of this algorithm to achieve a solution of our problem is expressed in

THEOREM 4.1 *The backtrack algorithm establishes a minimum essential dumbbell set.*

PROOF: At line 7, MF is set equal to $D^k_{j_k}$ when $\bar{B}^k_{j_k}$ is acyclic, where $\bar{B}^k_{j_k}$ has been obtained from $\bar{B}^0_1 = \bar{B}^I$ by deletion of dumbbells at lines 3 and 6. The dumbbells deleted at line 6, recorded as the elements of $D^k_{j_k}$, are those which when deleted from $\bar{B}^0_1$ realize an acyclic bipartite digraph, equal to $\bar{B}^k_{j_k}$ upon deletion of trivial strongly connected components — those dumbbells deleted at line 3. That is, B(X-X(MF)) is acyclic and, furthermore, has a complete matching (by Theorem 4.1). At lines 2, 4, and 8 the stack α is purged of or precluded from acquiring any reduced graph which would lead to a $D^k_{j_k}$ of cardinality greater than that, $\hat{k}$, of the thusfar established MF (at line 7). As all essential dumbbell sets are considered, MF must be on exit from the algorithm (satisfaction of the conditions of line 9) the minimum essential dumbbell set. Note: The alternate stop by the test $\hat{k} = 1$ in

recognition of the fact that the cardinality of MF must be at least 1 for a strongly connected bipartite digraph as assumed for $\bar{B}^I$. ¤

Let us here observe that $\bar{B}^k_{j_k}$ is the j_k-th graph at the k-th level of the search process (or, tree), as illustrated in Fig. 5. The algorithm implements a depth first search strategy — at every phase of the process a path to greatest possible depth in the tree is sought and after each backtrack, return to line 1, such a path is again sought. Of course it is possible to implement a breadth first search strategy, but it would not then be possible to use the bound $\hat{k}$ to "prune" the tree, as is done at lines 2, 4, and 8 of our algorithm. We also note that the ordering of the dumbbells in the set of dumbbells at line 5 is arbitrary. Hence, the selected dumbbell d_n of subsequent steps is also arbitrary. One might select d_n from among those of the set not previously selected by some (heuristically established) criterion which engenders (with a high likelihood) an acyclic graph at a least depth in the tree.

As is evident (after a little thought), the number of twigs of the pruned tree is exponential — not polynomial — in the number of edges of B. Thus, even when the number of edges is modest, the bound on the number of passes through the algorithm is so large as to preclude using it. So we are led to ask: Is this the best we can expect to do? Unfortunately, because the problem is NP-complete† the answer is yes.

†Some combinatorial problems — such as the traveling salesman problem, the map coloring problem, and the feedback vertex set problem — are more difficult than others, in the sense that for these problems there is no available algorithm to solve any one of them in operations and with storage bounded by a polynomial in the number of edges and/or nodes of the problem's graph. Karp has shown [16] that many of these difficult graph problems are equivalent in the sense that for each of them, or none of them, there is a polynomial bounded algorithm by which to generate a solution. This result strongly suggests that these problems will remain intractable.

To prove that this problem is NP-complete, we recast the minimum dumbbell essential set problem into a decision problem [16,17,18] consisting of:

INPUT: a bipartite graph $B = (S,T,U)$ and an integer k_{MF}.

PROPERTY: There exists a set MF of k_{MF} DUMBBELLS IN $B = (S,T,U)$ such that $B(X-X(MF))$, with $X = S \cup T$, has a complete matching I, and is acyclic wrt I.

By invoking a theorem of Cook [18], it can be shown that a decision problem M is an NP-complete problem by showing that M is an NP problem and by showing that any one of the decision problems already known to be an NP-complete problem, such as the feedback node set decision problem [17], can be transformed† into M. We shall proceed in this way to validate.

THEOREM 4.2 *The minimum essential dumbbell set (decision) problem is NP-complete.*

PROOF: The backtrack algorithm presented previously is of polynomial bounded depth — equal to half the number of vertices, or $|S| = |T|$ — and can be used to solve MDS; therefore, MDS is an NP problem. Let FNS denote the feedback node set decision problem.

INPUT a digraph $G = (X,E)$ and an integer k_{MV},

PROPERTY there exists a set MV of k_{MV} nodes in $G = (X,E)$ such that $G(X-MV)$ is acyclic.

†A decision problem L is said to be transformable into a decision problem M if there exists a relation f such that f maps the input of L into the input of M, there is an algorithm to compute f in polynomial bounded operations, and f preserves the problem, that is, the input of L satisfies the property of L iff the corresponding input of M satisfies the property of M.

The transformation of the input of FVS into that of MDS is defined as follows:

(i) $k_{MF} = k_{MV}$,

(ii) $s_i \in S$, $t_i \in T$, and $\{s_i,t_i\} \in U$ of $B = (S,T,U)$ when $x_i \in X$ of $G = (X,E)$, and

(iii) $\{s_i,t_j\}_1 \in U$ of B and $\{s_i,t_j\}_2 \in U$ of B — a pair of parallel edges — when $(x_i,x_j) \in E$ of G.

An input for MDS can be realized from one for FVS in polynomial bounded operations $(3|X| + |E|)$. Now, we must show that S for FVS with G and k_{MV} is true iff S for MDS with B and k_{MF}, derived from G and k_{MV} by the transformation, is true.

[*Only if*] Let $MV = \{x_1,\ldots,x_\ell\}$ be a feedback vertex set of G of cardinality $k_{MV}(=\ell)$. Now, $MF = \{s_1-t_1,\ldots,s_\ell-t_\ell\}$ is an essential dumbbell set of B cardinality $k_{MF} = k_{MV}$. (This follows easily from the fact that every alternating cycle wrt $I = \{\{s_1,t_1\},\ldots,\{s_{|X|},t_{|X|}\}\}$ of $\hat{B}$ corresponds by (ii) and (iii) to a directed cycle of G. Hence, S for FVS with G and k_{MV} implies S for MDS with B and k_{MF}.

[*if*] Let $MF = \{d_1,\ldots,d_\ell\}$ be an essential dumbbell set of cardinality $k_{MF}(=\ell)$. Now, $B(X-X(MF))$ must have a complete matching I such that $B(X-X(MF))$ is acyclic wrt to I. By (iii) it follows, with appropriately assigned index values, that $I = \{s_{\ell+1}-t_{\ell+1},\ldots,s_{|X|}-t_{|X|}\}$, for otherwise there would exist one or more simple alternating cycles wrt I in $B(X-X(MF))$. Then, by (ii) and (iii) it must be that $G(\{x_{\ell+1},\ldots,x_{|X|}\})$ is devoid of directed cycles. This implies that $MV = X - \{x_{\ell+1},\ldots,x_{|X|}\}$ is a feedback vertex set of G of cardinality $k_{MV} = k_{MF}$. Hence, S for MDS with B and k_{MF} implies S for FVS with G and k_{MV}. ¤

The fact that the minimum essential dumbbell set problem is an NP-complete problem was our justification for developing an algorithm to solve the less difficult problem: *Given a bipartite graph B, determine a minimal — not necessarily minimum — essential dumbbell set.* This algorithm is presented next.

V. ALGORITHM FOR MINIMAL ESSENTIAL DUMBBELL SET

We can solve this new problem only after establishing our meaning of minimal — local minimum — essential dumbbell set. To that end we tender the following: A minimal essential dumbbell set, denoted LMF, is an essential dumbbell set such that no proper subset (of LMF) is also an essential dumbbell set.

In seeking a computationally efficient algorithm to establish a LMF of some bipartite graph B = (S,T,U), we must avoid a backtrack procedure. Rather, we want a sequential procedure which assigns dumbbells which constitute a complete matching to LMF or to NF — a set of dumbbells such that, upon exit from the algorithm, X(NF) = X-X(LMF) with X = S ∪ T. Information about alternating cycles passing through already assigned dumbbells is essential if subsequent assignments are to be made efficiently. This information is retained in a pair of labels for each edge established and updated in a pair of procedures invoked in making dumbbell assignments. Thus, we offer

PROCEDURE 5.1 *The* labeled elimination *of a dumbbell* $\check{d} = \check{t} \to \check{s}$ *from a bipartite digraph* $\bar{B}^i = (S^i, T^i, E^i)$ *— related to a bipartite graph* B_i *with a complete matching by Construction 3.1 — is accomplished by, in turn, (i) modifying the labels of the edges of the set*

$$E^i_+ = \{(s_m, t_n): (s_m, t_n) \in E^i \wedge (s_m, \check{t}) \in E^i \wedge (\check{s}, t_n) \in E^i\}$$

by setting

$$v(s_m, t_n) = +1 \quad \forall\ (s_m, t_n) \in E^i_+$$

and of the edges of the set

$$E^i_- = \{(t_n, s_m):\ (t_n, s_m) \in E^i \wedge (s_m, \check{t}) \in E^i \wedge (\check{s}, t_n) \in E^i\}$$

by setting

$$v(t_n, s_m) = -1 \quad \forall\ (t_n, s_m) \in E^i_-,$$

(ii) adding a set of new edges

$$E^i_a = \{(s_m, t_n):\ (s_m, t_n) \notin E^i \wedge (t_n, s_m) \notin E^i \wedge (s_m, \check{t}) \in E^i \wedge (\check{s}, t_n) \in E^i\}$$

with labels

$$u(s_m, t_n) = 1 \text{ and } v(s_m, t_n) = +1\ \forall\ (s_m, t_n) \in E^i_a;$$

and

(iii) deleting $\check{d}$ *from* $\bar{B}^i$.

$$v(t_n, s_m) = -1 \quad \forall\ (t_n, s_m) \in E^i_-.$$

The label u = 0 indicates that the edge was in the original bipartite digraph; alternatively, u = 1 indicates that the edge was inserted during some labeled elimination. The label v ≠ 0 indicates that between the edge's vertices there was a directed path (of length 3) through some eliminated dumbbell; otherwise, v = 0. Specifically, v = +1 [alternatively, v = -1] indicates the labeled edge is oriented from S to T [alternatively, from T to S] — as the path was [alternatively, was not]. This leads to a second interpretation of the v ≠ 0 label: An edge with v = -1 was in a directed cycle containing some eliminated dumbbell. This information on orientation of edges for which v ≠ 0 is redundant; however, it is convenient to have in stating the algorithm to follow shortly.

Labeled elimination is defined for a dumbbell oriented from T to S. For a dumbbell oriented from S to T we need an additional procedure to reverse the orientation. Thus, we tender

PROCEDURE 5.2 *The* labeled reversion *of an edge* (s_m, t_n) *in a bipartite digraph is accomplished by (i) replacing* (s_m, t_n) *with* (t_n, s_m) *and (ii) set* $v(t_n, s_m) = -v(s_m, t_n)$.

The ingredients now exist for our algorithm whereby we select a maximal set of dumbbells whose edges are a matching with respect to which the graph has no simple alternating cycle. A minimal essential dumbbell set is then obtained as those dumbbells whose edges are needed to augment the matching so as to arrive at a complete matching. The matching is obtained in two stages: (1) By examining a succession of strongly connected, but not trivially so, subgraphs a maximal set of dumbells is determined such that the associated edges are a matching with respect to which the graph has no simple alternating cycle (2) This set of dumbbells is then augmented by a maximal number of dumbbells, drawn from a succession of trivially strongly connected subgraphs such that the same criteria are met. We herewith tender our

MINIMAL ESSENTIAL DUMBBELL SET (MEDS) ALGORITHM

BEGIN Make the assignments $\bar{B}^0 = (S^0, T^0, E^0) \leftarrow \bar{B}^I = (S, T, E)$, $u(x,y) \leftarrow v(x,y) \leftarrow 0 \quad \forall (x,y) \in E^0$, $NF \leftarrow \phi$, and $i \leftarrow 0$.

{1} REPEAT Let $\tilde{E}^i = \{(x,y) : (x,y) \in E^i \wedge u(x,y) = 0\}$ and $\tilde{B}^i = (S^i, T^i, \tilde{E}^i)$. Determine the strongly connected components, but not trivially so, of $\tilde{B}^i$ and let $\tilde{E}^i_s$ denote their edge set. Set $\tilde{D}^i = \{x \to y : (x,y) \in \tilde{E}^i_s \wedge v(x,y) = 0\}$.

IF $\tilde{D}^i \neq \phi$

{2} THEN Select any dumbbell, denote it as $\tilde{d}^i$, in $\tilde{D}^i$.

IF $\tilde{d}^i$ is oriented from S^i to T^i

{3} THEN Find a directed cycle in $\tilde{B}^i$ containing $\tilde{d}^i$. Do a labeled reversion of each edge of the cycle.

{4} BEGIN Make the assignment $i \leftarrow i + 1$. Do a labeled elimination of $\tilde{d}^{i-1}$ from $\bar{B}^{i-1}$ to obtain $\bar{B}^i = (S^i, T^i, E^i)$. Add $\tilde{d}^{i-1}$ to NF.

END

UNTIL $\tilde{D}^i = \phi$

REPEAT Let $\tilde{D}^i = \{x \rightarrow y : (x,y) = E^i \wedge u(x,y) = 0 \wedge v(x,y) = 0\}$.

IF $\tilde{D}^i \neq 0$

{5} THEN Select any dumbbell oriented from T^i to S^i, denote it as $\tilde{d}^i$, in $\tilde{D}^i$. Make the assignment $i \leftarrow i + 1$. Do a labeled elimination of $\tilde{d}^{i-1}$ from $\bar{B}^{i-1}$ to obtain $\bar{B}^i(S^i, T^i, E^i)$. Add $\tilde{d}^{i-1}$ to NF.

UNTIL $\tilde{D}^i = \phi$

{6} BEGIN Set LMF = $\{x \rightarrow y : (x,y) \in E^i \wedge u(x,y) = 0 \wedge v(x,y) = -1\}$.

END

END

The validity of this algorithm to obtain a solution of our less difficult, modified problem is expressed in

THEOREM 5.1 *The MEDS algorithm establishes a minimal essential dumbbell set.*

PROOF: Let us first establish the fact that LMF is an essential dumbbell set of B which by Construction 3.1 engendered $\bar{B}^I$. A dumbbell added to NF at line 4 is drawn from a strongly connected component of $\bar{B}^{i-1} \subset B^I$. Therefore, its associated, non-directed edge by Theorem 4.1 is to be found in a complete matching for B. The associated non-directed edge of a dumbbell added to NF at line 5 must also be in a complete matching for B, as the dumbbell is oriented from T to S and is a strongly connected component in a trivial sense of $\bar{B}^{i-1} \subset B^I$. Let $\hat{i}$ denote the value of i when line 6 is reached. Clearly, the associated, non-directed edge of a dumbbell in LMF must too be in a complete matching for B, as the dumbbell is oriented from T to S — $v(x,y) = -1$ — and is in a strongly connected component (possibly, in a trivial sense) of $\bar{B}^{\hat{i}} \subset B^I$. It is evident in fact that the non-directed edges associated with the dumbbells of NF $\cup$ LMF constitute a complete matching for B. It is also evident that NF and LMF are disjoint. Therefore, it follows, that B(X-X(LMF)) = B(X(NF)) has a complete matching. A dumbbell to be added to NF at line 4 or line 5 is drawn from candidates for which v = 0. Perforce, there can be no simple alternating cycle wrt the edges of the dumbbells in NF. Furthermore, the edges removed with the dumbbell during the labeled elimination being oriented from S to T (a non-zero v of only +1) cannot share a directed cycle with dumbbells in NF; with impunity they may subsequently be edges of B(X(NF)). Therefore, B(X-X(LMF)) = B(X(NF)) must be acyclic. As B(X-X(LMF)) has a complete matching and is acyclic, LMF must be an essential dumbbell set of B.

It remains but to show that LMF is minimal. For each dumbbell in LMF it is true that $v = -1$. That means that LMF diminished by any dumbbell $\tilde{d}$ cannot be an essential dumbbell set as B cannot be acyclic wrt NF augmented by $\tilde{d}$. Hence LMF is minimal.

¤

In line 2, the dumbbell selection is completely arbitrary. Obviously there exists a particular choice which upon exit from the algorithm would have $|LMF| = |MF|$. That is LMF would be a minimum, not just minimal, essential dumbbell set. Unfortunately, there is no known apriori criterion — Theorem 4.2 assures us of that — for the dumbbell selection to achieve the desired end. However, a heuristic selection guide might diminish the disparity between $|LMF|$ and $|MF|$.

In [11], a heuristic dumbbell selection guide was incorporated in an algorithm to establish an essential dumbbell set. The algorithm was efficient but suffered from the fact that there could be no guarantee that the essential dumbbell set was minimal. That could be achieved only by augmenting the algorithm with an additional, termed refinement, step.

We propose that the same heuristic selection guide should be used in line 2 of the MEDS algorithm, which (as Theorem 5.1 testifies) does not exhibit the limitation of the just above cited algorithm; the MEDS algorithm establishes a minimal essential dumbbell set. We shall refer to the so modified MEDS algorithm as the MMEDS algorithm. The modified line 2 is

{2'} THEN Select from $\tilde{D}^i$ a dumbbell, denote it as $\tilde{d}^i$, which first has minimum T-degree in $\tilde{D}^i$ *wrt* $\tilde{B}^i$ and then (among those of minimum T-degree) has maximum S-degree.

The rationale for this guide is: By selection of a dumbbell of minimum S-degree, the number of dumbbells being committed to LMF is being made small, as small as possible with such a simple measure. From among equal choices, selection of a dumbbell of maximum T-degree results in the number of simple alternating cycles passing through the dumbbells committed to LMF being made large, as large as possible with such a simple measure. By this latter choice it can be reasoned that the number of dumbbells that must be subsequently committed to LMF to open the remaining simple alternating cycles is being made small.

The number of operations for the MMEDS algorithm (also, the MEDS algorithm) is polynomial bounded — not exponential bounded. In fact, the number of operations in the worst case — a maximally connected bipartite graph — is of order n^3. This has elsewhere been established [11] for lines 1 and 3 and is easily shown to be the case for line 2'. For the remaining steps the number of operations are of order not exceeding n^2 [11].

VI. MMEDS ALGORITHM REALIZATION AND ILLUSTRATION

The MMEDS algorithm has been realized as a collection of *APL* functions, the principal function being named *MMEDS*. (A listing of these functions is to be found in the Appendix. A somewhat expansive description of this realization is to be found in [19].) To illustrate the MMEDS algorithm thus realized we will next present some examples.

EXAMPLE 1 Consider the matrix

S vertex numbers \ T vertex numbers	8	9	10	11	12	13	14
1	1	1	1	1	0	1	1
2	0	1	1	1	0	1	1
3	0	1	1	1	0	1	0
4	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1
6	0	0	1	1	0	1	0
7	1	1	1	1	1	1	1

which for subsequent purposes is the *APL* variable *M*. Vertex numbers are assigned first to the S vertices — 1 through 7, one for each row — and then to the T vertices — 8 through 14, one for each column — as shown above adjacent to the rows and columns of the matrix. The next step is to type†

MMEDS ⍝

The subsidiary function *INPUT* (see the Appendix.) responds with the query

ENTER NUMBER OF NODES:

to which 14⍝ is the correct response. The exchange would appear as

ENTER NUMBER OF NODES: 14⍝

Then by a sequence of queries and responses the edge data is entered as††

†The ⍝ denotes a (non-printing) carriage return.

††The missing edge data — edges 6 through 34 — may be found in the following (packed) edge list:

edge	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
S vertex	3	5	4	4	1	7	5	3	3	1	4	1	4	1	2	3	5	5
T vertex	10	10	9	10	14	9	9	9	13	10	14	8	13	13	10	11	11	14

edge	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
S vertex	7	7	4	2	7	1	5	5	2	6	7	5	7	7	4	6	1	2	4	6	2
T vertex	11	10	12	14	13	11	13	8	13	10	14	12	8	12	11	11	9	11	8	13	9

```
queries      responses
EDGE 1  :  3  10⍲
EDGE 2  :  5  10⍲
EDGE 3  :  4  9 ⍲
EDGE 4  :  4  10⍲
EDGE 5  :  1  14⍲
   .
   .
   .
   .
EDGE 35 :  1  9 ⍲
EDGE 36 :  2  11⍲
EDGE 37 :  4  8 ⍲
EDGE 38 :  6  13⍲
EDGE 39 :  2  9 ⍲
EDGE 40 :⍲
```

After all the edges have been entered then, as shown, a simple ⍲ is the correct response to the last query for an edge. Note: Each edge is entered as: S vertex and (then) T vertex. When execution of *MMEDS* is complete, the set of dumbbells in NF and LMF are the elements of correspondingly labeled *APL* variables and can be printed as follows:

```
        NF ⍲
28   8  22 12  21  ←— dumbbell edges
        LMF ⍲
17  23            ←— dumbbell edges
```

Note: The edges 8, 22, 24, 25, 28, 32, and 37 comprised the complete matching for the initial graph and during execution five labeled reversions were accomplished (after selection of dumbbells 28, 8, 22, 12, and 21) and line 1 was encountered six times.

Since LMF has just two dumbbells, there must exist row and column permutations which transform *M* to a 2-BLT matrix. By executing the function *PERM* as

```
PERM ⎕
```

the row and column permutations — lists of row and column indices — are composed from the dumbbell lists *NF* and *LMF*. Those permutations are the elements of the first and second, respectively, rows of the (matrix valued) variable *P*. The 2-BLT form of *M* can be displayed as

```
M[P[1;];P[2;]] ⎕
```

```
1 0 0 0 0 1 1
1 1 0 0 0 1 1
1 1 1 0 0 1 1
1 1 1 1 0 1 1
1 1 1 1 1 1 1
1 1 1 1 1 1 1
1 1 1 1 1 1 1
```

Obviously, the least possible value for the border width has been found.

Note: The permutation lists — rows of *P* — can be displayed as

```
P ⎕
6 3 2 1 7 4 5
4 2 7 1 5 3 6
```

EXAMPLE 2 The MMEDS algorithm engendered row and column permutations

```
 1  4  6  7  9  3 13 19 11 20 16 14 22  2 15 23 24 25 12 21 10 18 17  5  8
11  8 23  4 22 15 17 13 10  2  1  7  6 19  9 16 18 20 24  5 21  3 25 14 12
```

to transform the 25x25 matrix

0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0
0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1
0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0
0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	1

Note: The original matrix was obtained by randomly selected row and column permutations of a sparse — 12% fill — 8-BLT matrix.

To provide an illustration of the polynomial bound on the number of operations we obtained the central processor time consumed by this realization of the MMEDS algorithm applied to a number of maximally connected bipartite graphs.† A log-log plot of the time, τ_{cp}, versus n, half the number of (graph) vertices, is presented in Fig. 6. The slope for moderate values of n is slightly greater than 3, where 3 is the expected value for large n, based upon the number of operations being of order n^3 under worst case conditions

VII. CONCLUDING DISCUSSION

We have herein shown that the problem of finding a minimum k k-BLT form of a nonsingular (non necessarily symmetric) matrix is equivalent to finding a minimum essential dumbbell set of a bipartite graph with a complete matching.†† We then established the fact that the latter — hence, also the former — problem is an NP-complete (intrinsically difficult) problem. We have proposed, as an alternative to a backtrack algorithm by which this NP-complete problem could be solved, an algorithm to solve the somewhat less ambitious problem of finding a minimal — local minimum, rather than minimum — essential dumbbell set. That (sequential, rather than backtrack oriented) algorithm, the MEDS algorithm with an adopted heuristic for dumbbell selection at line 2 was renamed the MMEDS algorithm. An *APL* realization of that algorithm was then illustrated. In the first example it was

†Central processor time is for *VS APL* under CMS on an IBM system 370 model 145 computer.

††An efficient algorithm for evaluating a complete matching is described in [21].

seen to lead to the minimum k-BLT form of the initial matrix. In the second example it lead to a value of k less than a known upper bound on the least value of k. Furthermore, we illustrated the polynomial bound on the number of operations when invoking the MMEDS algorithm.

We have also created an *APL* realization of the MEDS algorithm (with arbitrary dumbbell selection at line 2) and found as anticipated that, in general the MMEDS algorithm performed better. This we support by the data for fourteen cases presented in Table 1. In each case: The original matrix was obtained by randomly selected row and column permutations on a not particularly sparse k-BLT matrix; then edge lables were assigned randomly to the edges, each associated with a non-zero matrix element.

We conclude with the comment: We believe that the MMEDS algorithm represents an efficient procedure by which to find a minimal essential dumbbell set — hence, a minimal k k-BLT form of a matrix.

ACKNOWLEDGEMENTS

The authors gratefully acknowedge the many helpful discussions and comments by A. K. Kevorkian, E. S, Kuh, C. A. Desoer, and F. G. Gustavson, and the constructive criticisms of the reviewers.

REFERENCES

[1] A. K. Kevorkian and J. Snoek, "Decomposition in large scale systems: theory and applications in solving large sets of nonlinear simultaneous equations," in *Decomposition of large scale problems* (D. M. Himmemblau, ed.), North Holland, Amsterdam, 1973.

[2] M. A. Saunders, "A fast, stable implementation of the simplex method using Bartels-Golub updating," in *Sparse Matrix Computations* (J. R. Bunch and D. J. Rose, eds.), Academic Press, New York, 1976.

[3] A. K. Kevorkian, "A decompositional algorithm for the solution of large systems of linear algebraic equations," *Proceedings of the 1975 IEEE International Symposium on Circuits and Systems,* 116-120.

[4] L. K. Cheung and E. S. Kuh, "The bordered triangular matrix and minimum essential set of digraph," *IEEE Trans. on Circuits and Systems,* CAS-21 (1974), 633-639.

[5] F. M. Callier, W. T. Chan, and C. A. Desoer, "Input-output stability of interconnected systems using decomposition techniques," *IEEE Trans. on Circuits and Systems,* CAS-23 (1976), 714-729.

[6] G. Guardabassi, "A note on minimal essential sets," *IEEE Trans. on Circuit Theory*, CT-18 (1971), 557-560.

[7] G. W. Smith and R. B. Walford, "The identification of a minimal feedback vertex set of a directed graph," *IEEE Trans. on Circuits and Systems,* CAS-22 (1975), 9-15.

[8] A. K. Kevorkian, "On bordered triangular or lower N forms of an irreducible matrix," *IEEE Trans. on Circuits and Systems,* CAS-23 (1976), 621-624.

[9] L. K. Cheung and E. S. Kuh, "The bordered triangular matrix and minimum essential set of a digraph," University of California, Electronics Research Laboratory, Memorandum M 375, Berkeley, CA. 1973.

[10] A. Sangiovanni-Vincentelli, "A graph theoretical interpretation of nonsymmetric permutations on sparse matrices," *Int. Jour. on Circuit Theory and Applications*, 5 (1977), 139-147.

[11] A. Sangiovanni-Vincentelli, "An optimization problem arising from tearing methods," in *Sparse matrix computations* (J. R. Bunch and D. J. Rose, eds.), Academic Press, New York, 1976.

[12] F. Harary, *Graph theory*, Addison-Wesley, Reading, MA, 1969.

[13] C. Berge, *Graphs and hypergraphs*, North-Holland, Amsterdam, 1973.

[14] E. Lawler, *Combinatorial optimization, networks and matroids*, Holt, Rinehart and Winston, New York, 1976.

[15] A. Sangiovanni-Vincentelli, "A note on bipartite graphs and pivot selection in sparse matrices," *IEEE Trans. on Circuits and Systems*, CAS-23 (1976), 783-791.

[16] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations* (R. E. Miller and J. W. Thatcher, eds.), Plenum, New York, 1972.

[17] R. M. Karp, "On the computational complexity of combinatorial problems," *Networks*, 5 (1975), pp. 45-68.

[18] S. A. Cook, "The complexity of theory-proving procedures," *Proceedings of the Third ACM Symposium on Theory of Computing*, 1971.

[19] T. A. Bickart and A. Sangiovanni-Vincentelli, "Minimal essential dumbbell set algorithm: *APL* realization," Syracuse University Electrical and Computer Engineering Department Report TR 78-Syracuse, N.Y., 1978.

[20] N. Wirth, Systematic programming, Prentice-Hall, Englewood Cliffs, N.J. 1973.

[21] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," SIAM Jour. on Computing, 2 (1973), 225-231.

APPENDIX

The MMEDS algorithm is realized as the niladic *APL* function *MMEDS* next listed.

```
      ∇ MMEDS;CC;CNMS;DB;DBN;DLST;K
[1]     INPUT
[2]     ATR←(4,B)↑(1,B)⍴1
[3]     CM
[4]     NF←⍳0
[5]    L1:SCMPT
[6]     →L3⌈⍳0=SLCT1A
[7]     →L2⌈⍳0=ATR[1;DB]
[8]     LR
[9]    L2:K←⎕EX 2 4 ⍴'DLSTCC'
[10]    LE
[11]    NF←NF,DB
[12]    →L1
[13]   L3:→L4⌈⍳0=SLCT2
[14]    LE
[15]    NF←NF,DB
[16]    →L3
[17]   L4:LMF←(0=∨⌿ATR[1 4 ;])/⍳B
[18]    →0⌊⍳0=⍋NC 'CC'
[19]    K←⎕EX CC
      ∇
```

The hierarchy of functions, *MMEDS* being at the top, is displayed in Fig. A-1. Note: The functions *CYCL*, *CNCT*, and *CCNCT* execute themselves — are recursive. All of these functions, excluding *MMEDS*, are listed next.

```
      ∇ INPUT;E
[1]     ⍞←⎕TC[3], 'ENTER NUMBER OF NODES: '
[2]     LST←⍳N←⍎23↓⍞
[3]     B←0
[4]    L1:⍞←⎕TC[3], '    EDGE ' ,(3↑⍕B←B+1), ': '
[5]     →L2⌈⍳0=⍴E←13↓⍞
[6]     LST[E]← ¯3 ¯1 +⍴LST←LST,,⌽LST[E],[1.5]⌽E←⍎E
[7]     →L1
[8]    L2:B←B-1
      ∇
```

```
      ∇ CM;AE;AN;FN;FSN;FTN;I;IL;ME;NI;PE;PEU;PN;PNU;R;RE;RN;S;SE;SN
[1]    S∆CM←STOP
[2]    FN←⍳N,ME←⍳0
[3]   L1:FTN←(-S←(⍴FN)÷2)↑FN
[4]    SN←AN←FSN←(S)↑FN
[5]    SE←AE←⍳S←0
[6]   L2:R← 2 0 ⍴RN←RE←⍳I←0
[7]    S←~S
[8]    IL←⍴AN
[9]   L3:→L4⌈⍳IL<I←I+1
[10]   R←R,S ADJE AN[I]
[11]   →L3
[12]  L4:→L6⌈⍳0=⍴RN←(I←~R[1;]∊SN)/R[1;]
[13]   RE←I/R[2;]
[14]   →L5⌈⍳0=S
[15]   →L7⌈⍳0<+/I←RN∊FTN
[16]  L5:SN←SN,AN←RN
[17]   SE←SE,AE←RE
[18]   →L2
[19]  L6:⎕←⎕TC[3], 'THERE IS NO COMPLETE MATCHING.  ENTER  →  TO'
[20]   ⎕← 'TERMINATE EXECUTION.' ,⎕TC[3]
[21]  STOP:
[22]  L7:ATR[4;]←B⍴1
[23]   ATR[4;SE,I/RE]←0
[24]   NI←N⍴I←0
[25]   IL←⍴FTN
[26]   PNU←PEU←⍳0
[27]  L8:→L10⌈⍳IL<I←I+1
[28]   →L9⌈⍳1=1 REC FTN[I]
[29]   →L8
[30]  L9:PEU←PEU,PE,(¯1↑PN)EDGE FTN[I]
[31]   PNU←PNU,PN,FTN[I]
[32]   →L8
[33]  L10:ATR[ 1 4 ;]←(2,B)↑(1,B)⍴1
[34]   ATR[1;ME←((~ME∊PEU)/ME),(~PEU∊ME)/PEU]←0
[35]   →0⌊⍳N=2×⍴ME
[36]   FN←(~FN∊PNU)/FN
[37]   →L1
      ∇
```

```
      ∇ Z←S ADJE I;E;J
[1]    Z← 2 0 ⍴0
[2]    J←I-1
[3]   L1:→0⌊⍳I=J←LST[J+1]
[4]    →L1⌈⍳S≠ATR[1;E←⌈(J-N)÷4]
[5]    Z←Z,LST[J],E
[6]    →L1
      ∇
```

```
      ∇ Z←S REC A;B;I
[1]    Z←0
[2]    B←S ADJN A
[3]   L1:→0⌊⍳(⍴B)<I←NI[A]←NI[A]+1
[4]    →L1⌈⍳1=B[I]∊PNU
[5]    →L2⌈⍳0=S
[6]    →L3⌈⍳1=Z←B[I]∊FSN
[7]   L2:→L4⌈⍳1=Z←(~S)REC B[I]
[8]    →L1
[9]   L3:PE←⍳0
[10]   PN←,B[I]
[11]   →0
[12]  L4:PE←PE,(¯1↑PN)EDGE B[I]
[13]   PN←PN,B[I]
      ∇
```

```
      ∇ Z←S ADJN I;J
[1]    Z←⍳0
[2]    J←I-1
[3]   L1:→0⌊⍳I=J←LST[J+1]
[4]    →L1⌈⍳∨/(S,0)≠ATR[ 1 4 ;⌈(J-N)÷4]
[5]    Z←Z,LST[J]
[6]    →L1
      ∇
```

```
      ∇ Z←I EDGE J;L
[1]    Z←0
[2]    L←I-1
[3]   L1:→0⌊⍳I=L←LST[L+1]
[4]    →L1⌈⍳1=ATR[4;⌈(L-N)÷4]
[5]    →L1⌈⍳J≠LST[L]
[6]    Z←⌈(L-N)÷4
      ∇
```

```
      ∇ SCMPT;LL;NF;NN;S
[1]    CNMS← 0 0 ⍴' '
[2]    DLST←STR
[3]    LL←NN←N⍴K←0
[4]    S←⍳0
[5]   L1:→0⌊⍳N<NF←NN⍳0
[6]    CNCT NF
[7]    →L1
      ∇
```

```
      ∇ Z←STR;I;J;L
[1]    Z←0,N⍴I←0
[2]   L1:→L4⌈⍳N<I←I+1
[3]    Z[I]←1+⍴Z
[4]    J←LST[I]
[5]   L2:→L1⌈⍳I=J
[6]    →L3⌈⍳1=∨/ATR[ 2 4 ;L←⌈(J-N)÷4]
[7]    →L3⌈⍳~((I≤N÷2)∧ATR[1;L]=1)∨(I>N÷2)∧ATR[1;L]=0
[8]    Z←Z,LST[J],L
[9]   L3:J←LST[J+1]
[10]   →L2
[11]  L4:Z[I]←1+⍴Z
      ∇
```

```
      ∇ CNCT A;C;I;M
[1]    LL[A]←NN[A]←1++/0≠NN
[2]    S←S,A
[3]    I←DLST[A+ 0 1 ]- 2 0
[4]   L1:→L3⌈⍳=/I←I+ 2 0
[5]    C←DLST[1↑I]
[6]    →L2⌈⍳0≠NN[C]
[7]    CNCT C
[8]    LL[A]←LL[A]⌊LL[C]
[9]    →L1
[10]  L2:→L1⌈⍳(NN[C]>NN[A])∨~C∊S
[11]   LL[A]←LL[A]⌊NN[C]
[12]   →L1
[13]  L3:→0⌊⍳LL[A]≠NN[A]
[14]   CMP(M←NN[S]≥NN[A])/S
[15]   S←(~M)/S
      ∇
```

```
      ∇ CMP A;C;E;I;J;L
[1]    →0⌊⍳2>⍴A
[2]    C←'C',(' '≠L)/L←⍒K←K+1
[3]    CNMS←(K,⍴C)↑CNMS
[4]    CNMS[K;]←C
[5]    ⍎C, '←⍳I←0'
[6]   L1:→0⌊⍳(⍴A)<I←I+1
[7]    J←DLST[A[I]+ 1 0 ]-1
[8]    E←((L←(-/J)÷2),2)⍴(1↓J)↓(1↑J)↑DLST
[9]    ⍎C,'←',C, ',E[(E[;1]∊A)/⍳L;2]'
[10]   →L1
      ∇
```

```
    ∇ Z←SLCT1A;D;DG;I;J;L;M
[1]   Z←0
[2]   →0⌊⍳K=0
[3]   →0⌊⍳0=J←⍴L←(0=ATR[3;L])/L←⍎',',CNMS
[4]   DG← 2 0 ⍴I←0
[5]  L1:→L2⌈⍳J<I←I+1
[6]   DG←DG,(('T' DEG D),'S' DEG D←LST[¯1 ¯3 +N+4×L[I]])
[7]   →L1
[8]  L2:M←⌊/DG[1;]
[9]   I←(M=DG[1;])/⍳J
[10]  DB←L[I[DG[2;I]⍳⌈/DG[2;I]]]
[11]  DBN←LST[¯3 ¯1 +N+4×DB]
[12]  Z←1
[13]  I←0
[14] L3:→L3⌈⍳0=DB∊⍎CNMS[I←I+1;]
[15]  CC←CNMS[I;]
[16]  K←+/~⎕EX CNMS[(I≠K)/K←⍳K;]
    ∇
```

```
    ∇ Z←F DEG D;A;B;I;M
[1]   A←D['ST'⍳F]
[2]   B←D['TS'⍳F]
[3]   Z←⍴(A≠M)/M←ADJ B
    ∇
```

```
    ∇ Z←ADJ I;E;J
[1]   Z←⍳0
[2]   J←I-1
[3]  L1:→0⌊⍳I=J←LST[J+1]
[4]   →L1⌈⍳(1≠E∊L)∨0≠ATR[3;E←⌈(J-N)÷4]
[5]   Z←Z,LST[J]
[6]   →L1
    ∇
```

```
    ∇ LR;I;J;LL;NN;S
[1]   →L1⌈⍳(I←DLST[1↓DBN])=J←N+2×((((-1+N-⍴DLST)÷2),2)⍴1↓N↓DLST)[;2]⍳DB
[2]   DLST[I,J]←DLST[(J←J+ 0 1 ),I←I+ 0 1 ]
[3]  L1:LL←NN←N⍴0
[4]   S←⍳0
[5]   CCNCT 1↓DBN
[6]   REV(I/⍳N)[⍋(I←LL=1)/NN]
    ∇
```

```
      ∇ CCNCT A;C;I;M
[1]    LL[A]←NN[A]←1++/0≠NN
[2]    S←S,A
[3]    I←DLST[A+ 0 1 ]- 2 0
[4]   L1:→0⌊⍳=/I←I+ 2 0
[5]    →L1⌈⍳~(1↓C←DLST[ 0 1 +1↑I])∊⍯CC
[6]    C←1↑C
[7]    →L2⌈⍳0≠NN[C]
[8]    CCNCT C
[9]    LL[A]←LL[A]⌊LL[C]
[10]   →L1
[11]  L2:→L1⌈⍳(NN[C]>NN[A])∨~C∊S
[12]   LL[A]←LL[A]⌊NN[C]
[13]   →L1
      ∇
```

```
      ∇ REV A
[1]    A←1↓A,1↑A
[2]    ATR[1;DB]←~I←1
[3]   L1:ATR[1;J]←~ATR[1;J←A[I]EDGE A[I+1]]
[4]    →L1⌈⍳(⍴A)>I←I+1
      ∇
```

```
      ∇ LE;E;I;J;S;SLST;T;TLST
[1]    I←¯1+S←1↓DBN
[2]    SLST←⍳0
[3]   L1:→L2⌈⍳S=I←LST[I+1]
[4]    →L1⌈⍳1=ATR[4;E←⌈(I-N)÷4]
[5]    ATR[4;E]←1
[6]    SLST←SLST,LST[I]
[7]    →L1
[8]   L2:I←¯1+T←1↑DBN
[9]    TLST←⍳0
[10]  L3:→L4⌈⍳T=I←LST[I+1]
[11]   →L3⌈⍳1=ATR[4;E←⌈(I-N)÷4]
[12]   ATR[4;E]←1
[13]   TLST←TLST,LST[I]
[14]   →L3
[15]  L4:→0⌊⍳(0=I←⍴SLST←(T≠SLST)/SLST)∨0=J←⍴TLST
[16]  L5:→L6⌈⍳0=E←(S←TLST[J])EDGE T←SLST[I]
[17]   ATR[3;E]←1
[18]   →L7
[19]  L6:LST[E]← ¯3 ¯1 +⍴LST←LST,,⌽LST[E],[1.5]⌽E←S,T
[20]   B←1↓⍴ATR←ATR, 1 1 1 0
[21]  L7:→L5⌈⍳0<J←J-1
[22]   J←⍴TLST
[23]   →L5⌈⍳0<I←I-1
      ∇
```

```
      ∇ Z←SLCT2
[1]   Z←0
[2]   →0⌊⍳(1+⍴ATR)<DB←(∨⌿ATR)⍳0
[3]   DBN←LST[ ¯3 ¯1 +N+4×DB]
[4]   Z←1
      ∇
```

The row and column permutations necessary to transform the matrix engendering $\bar{B}^I$ (See step 0 of the MMEDS algorithm.) are composed from the elements of (the *APL* global variables) *NF* and *LMF* and listed as the entries of row 1 and row 2, respectively, of (the *APL* global variable) *P* by the function *PERM* with the functions *INC* and *ADJP* next listed.

```
     ∇ PERM;D;HN;I;J;L;NLST;R;V
[1]    ATR←((1,B)⍴1),[1] ATR[2 2 2 ;]
[2]    B←+/~ATR[2;]
[3]    NLST←LST[,(¯1 ¯3 +N)∘.+4×NF]
[4]    I←0
[5]   L1:→L2⌈⍳B<I←I+1
[6]    →L1⌈⍳1=∧/LST[¯1 ¯3 +N+4×I]∊NLST
[7]    ATR[4;I]←1
[8]    →L1
[9]   L2:D←⍳0
[10]   HN←N÷2
[11]   V←HN↑INC
[12]   R←⍳0
[13]  L3:→L4⌈⍳HN<I←V⍳1
[14]   V[L]←V[L←ADJP J←(~J∊R)/J←ADJP I]-1
[15]   D←D,I EDGE J
[16]   R←R,I,J
[17]   →L3
[18]  L4:P←((2,HN)⍴LST[,(¯1 ¯3 +N)∘.+4×(NF←D),LMF])-⍉(HN,2)⍴0,HN
     ∇
```

```
     ∇ Z←INC;I
[1]    Z←⍳I←0
[2]   L1:→0⌊⍳N<I←I+1
[3]    Z←Z,⍴ADJP I
[4]    →L1
     ∇
```

```
    ∇ Z←ADJP I;J
[1]   Z←⍳0
[2]   J←I-1
[3]  L1:→0⌊⍳I=J←LST[J+1]
[4]   →L1⌈⍳1=ATR[4;⌈(J-N)÷4]
[5]   Z←Z,LST[J]
[6]   →L1
    ∇
```

original nonsingular sparse matrix				established value of k	
order	upper bound on minimum k	number of non-zero elements	maximum number of non-zero elements	MEDS algorithm	MMEDS algorithm
10×10	1	36	64	1	1
10×10	1	49	64	2	1
10×10	2	36	72	4	3
10×10	2	48	72	4	2
10×10	3	26	79	1	1
10×10	3	47	79	6	2
12×12	1	30	89	0	0
12×12	1	43	89	5	3
12×12	2	28	99	1	1
12×12	2	41	99	4	3
12×12	3	28	108	1	1
12×12	3	44	108	3	2
12×12	4	32	116	2	2
12×12	4	49	116	5	3

Table 1 Performance data for MEDS and MMEDS algorithms.

FIGURE CAPTIONS

Fig. 1 k-bordered lower triangular matrix

Fig. 2 Bipartite graph and matchings. a) Bipartite graph (i) and two matchings (ii and iii), b) Bipartite graph (i) and a complete — also, maximum cardinality — matching (ii), c) Bipartite graph (i) and a maximum cardinality matching (ii).

Fig. 3 Bipartite graph and dumbbells a) Bipartite graph B, b) Complete matching I, c) Fundamental dumbbell set D_I, d) Dumbbell set D, e) Dumbbell set $SA_D(\hat{d})$, f) Dumbbell set $TA_D(\hat{d})$, g) Essential dumbbell set F, h) Section graph B(X-X(F)), i) Minimum essential dumbbell set MF, j) Section graph (BX-X(MF)).

Fig. 4 Illustration of Proposition 3.1. a) Bipartite graph B, b) Complete matching I_a with $\hat{I}$ darkened, c) Complete matching I_b with $\hat{I}$ darkened, d) Cycle with edges in common with I_b darkened.

Fig. 5 Search tree.

Fig. 6 τ_{cp} versus n.

Fig. A-1 Hierarchy of *APL* functions realizing the MMEDS algorithm.

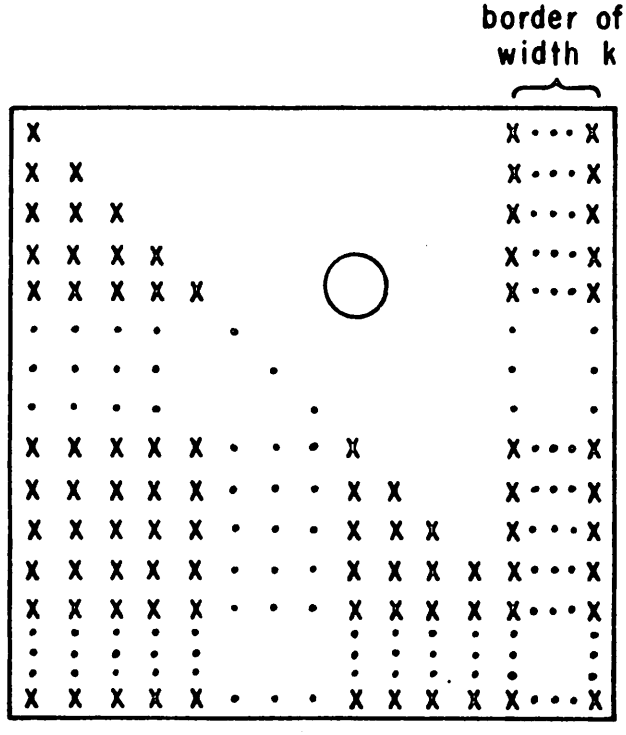

Fig. 1

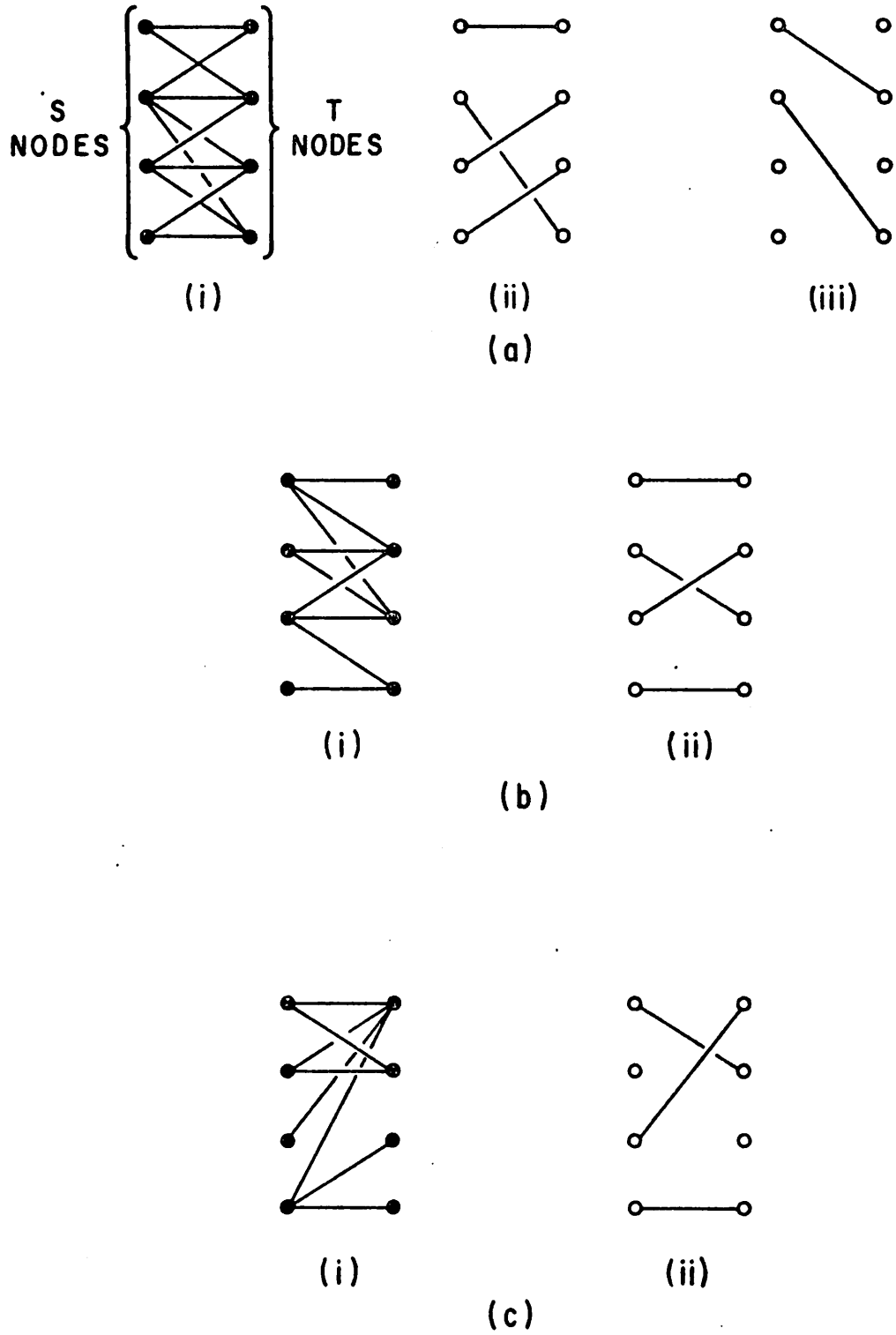

Fig. 2

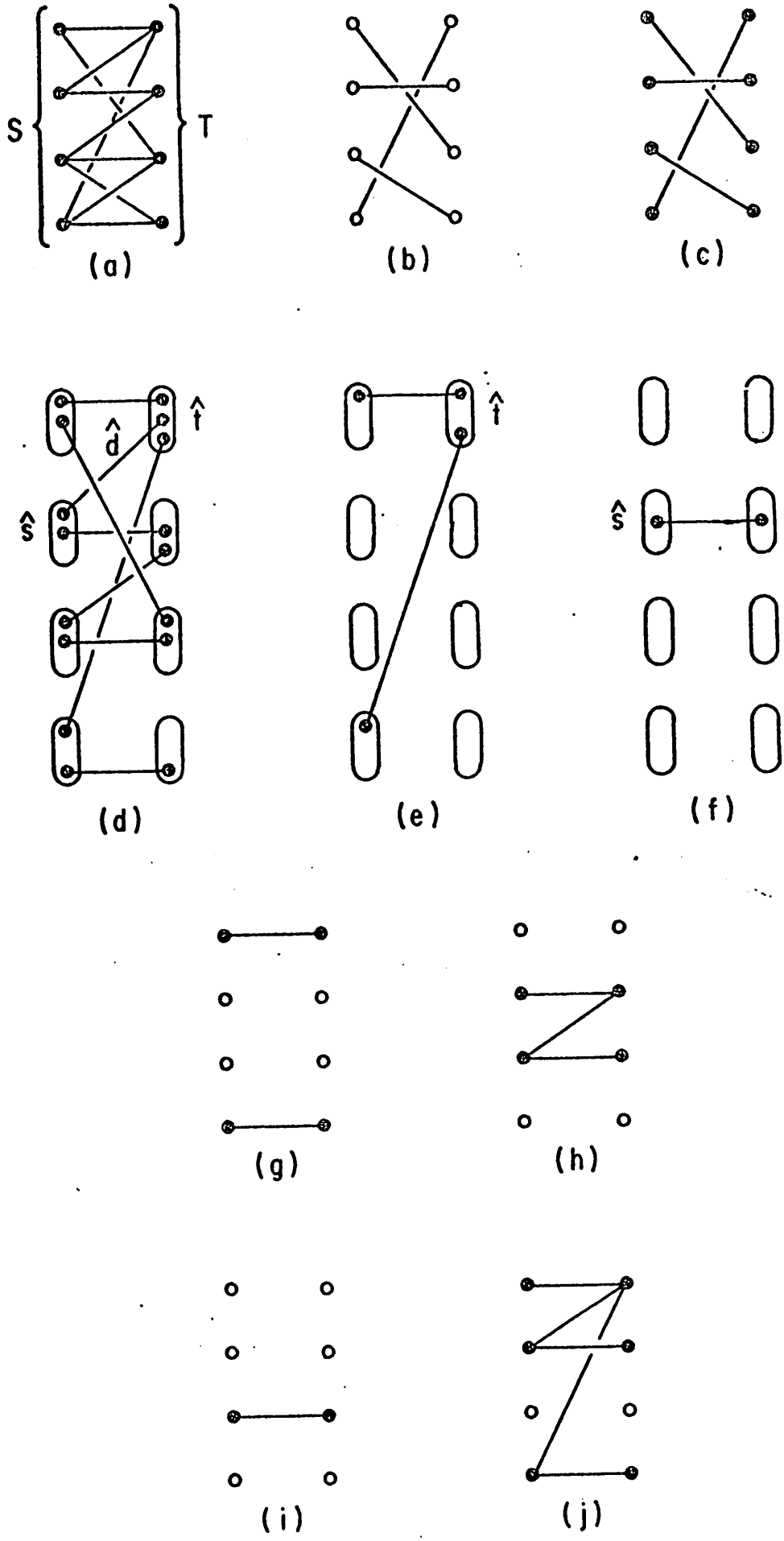

Fig. 3

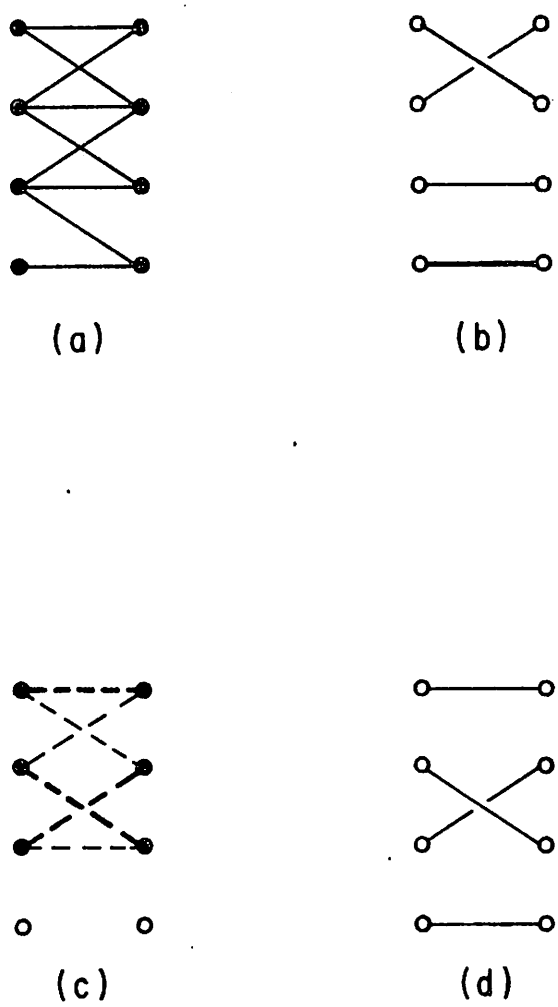

Fig. 4

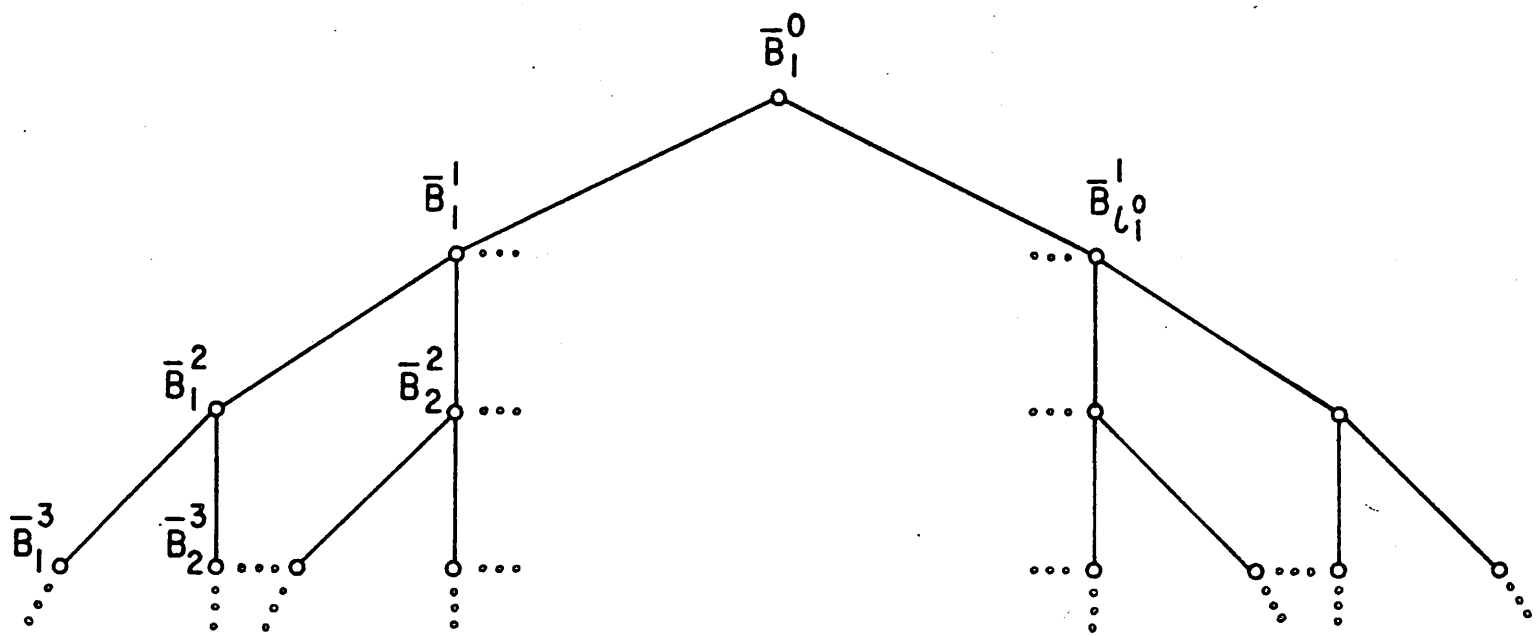

Fig. 5

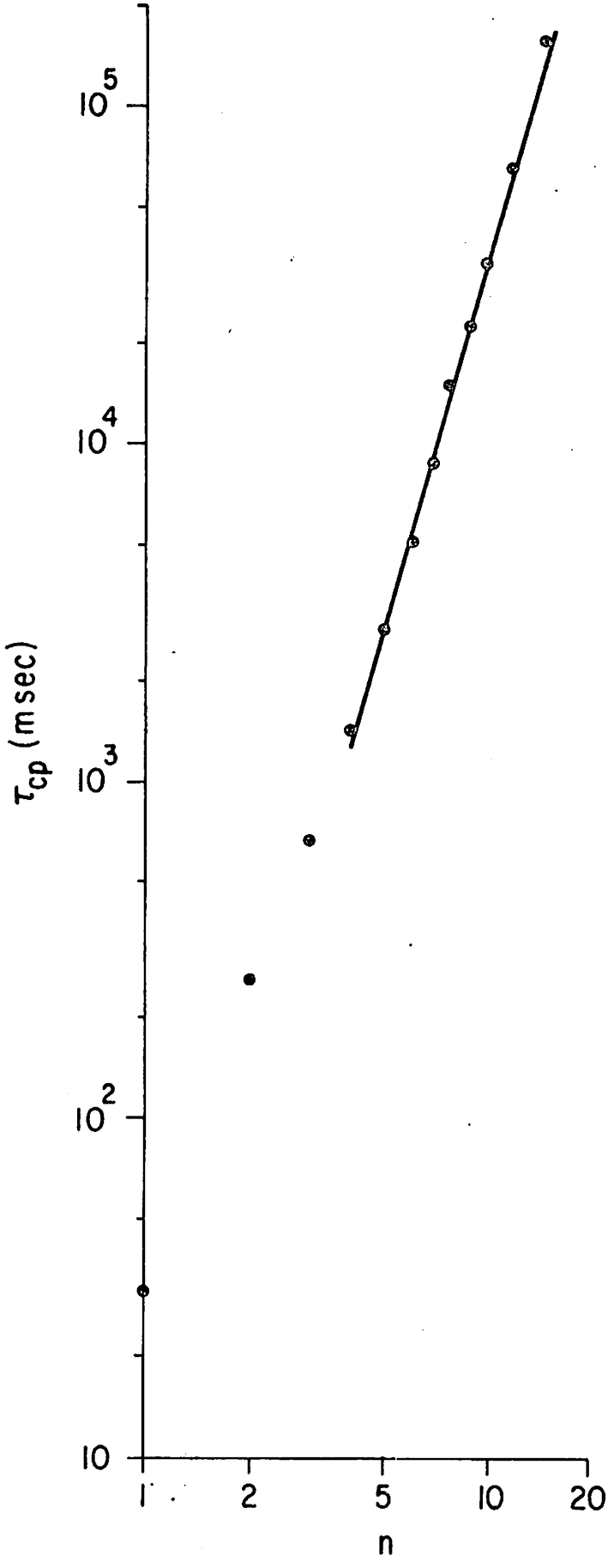

Fig. 6

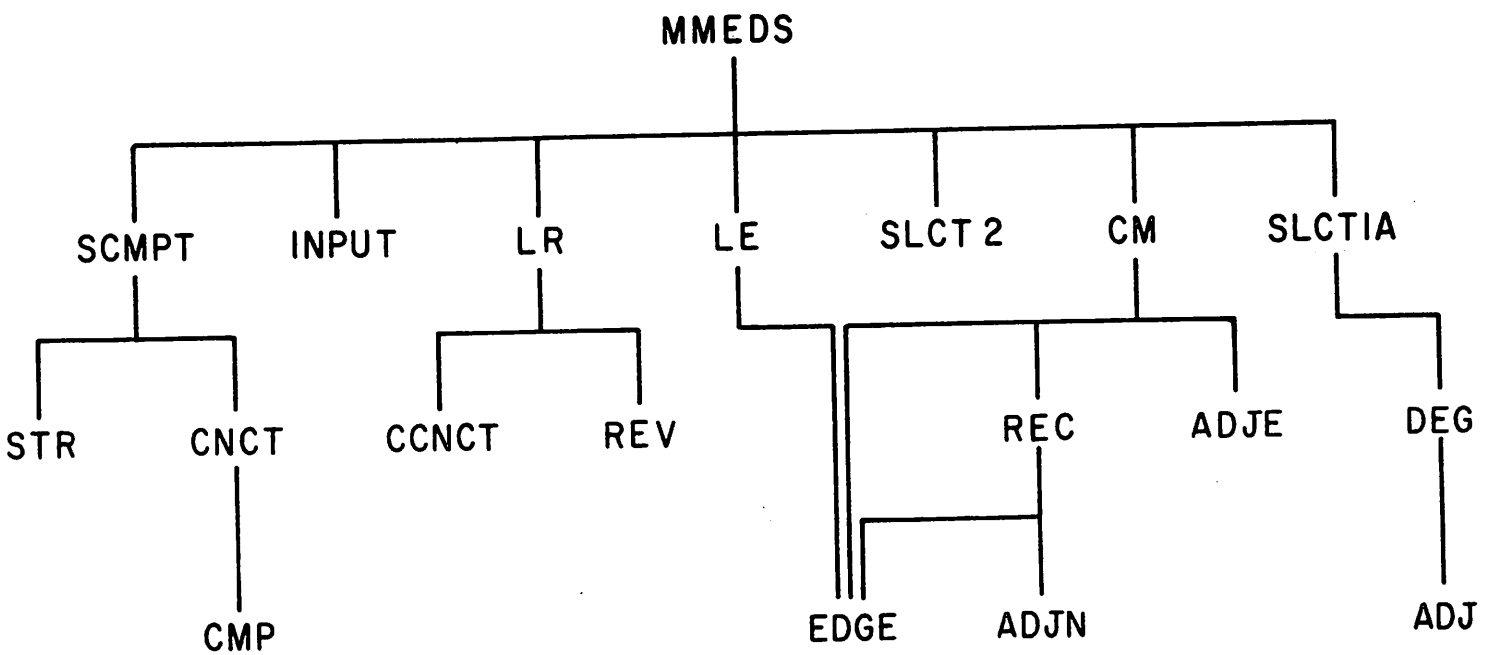
MMEDS
SCMPT
INPUT
LR
LE
SLCT 2
CM
SLCTIA
STR
CNCT
CCNCT
REV
REC
ADJE
DEG
CMP
EDGE
ADJN
ADJ

Fig. A-1