SIAM J. COMPUT.
Vol. 2, No. 4, December 1973

AN $n^{5/2}$ ALGORITHM FOR MAXIMUM MATCHINGS IN BIPARTITE GRAPHS*

JOHN E. HOPCROFT† AND RICHARD M. KARP‡

Abstract. The present paper shows how to construct a maximum matching in a bipartite graph with n vertices and m edges in a number of computation steps proportional to $(m + n)\sqrt{n}$.

Key words. algorithm, algorithmic analysis, bipartite graphs, computational complexity, graphs, matching

1. Introduction. Suppose we are given a rectangular array in which each cell is designated as "occupied" or "unoccupied". A set of cells is *independent* if no two of the cells lie in the same row or column. Our object is to construct an independent set of occupied cells having maximum cardinality.

In one interpretation, the rows of the array represent boys, and the columns represent girls. Cell i, j is occupied if boy i and girl j are compatible, and we wish to match a maximum number of compatible couples.

An alternate statement of the problem is obtained by representing the rows and columns of the array as the vertices of a bipartite graph. The vertices corresponding to row i and column j are joined by an edge if and only if cell i, j is occupied. We then seek a maximum matching; i.e., a maximum number of edges, no two of which meet at a common vertex.

This problem has a wide variety of applications ([3], [4], [5]). These include the determination of chain decompositions in partially ordered sets, of coset representatives in groups, of systems of distinct representatives, and of block-triangular decompositions of sparse matrices. The problem also occurs as a subroutine in the solution of the Hitchcock transportation problem, and in the determination of whether one given tree is isomorphic to a subtree of another.

In view of this variety of applications, the computational complexity of the problem of finding a maximum matching in a bipartite graph is of interest. The best previous methods ([1], [3], [4], [5]) seem to require $O(mn)$ steps, where m is the number of edges, and n the number of vertices. The present method requires only $O((m + n)\sqrt{n})$ steps.

We hope to extend our results to the nonbipartite case (cf. [2]). With this in mind, all the results in § 2 are derived for general graphs. The specialization to the bipartite case occurs in § 3.

2. Matchings and augmenting paths. Let $G = (V, E)$ be a finite undirected graph (without loops, multiple edges, or isolated vertices) having the vertex set V and the edge set E. An edge incident with vertices v and w is written $\{v, w\}$. A set $M \subseteq E$ is a *matching* if no vertex $v \in V$ is incident with more than one edge in M. A matching of maximum cardinality is called a *maximum matching*.

We make the following definitions relative to a matching M. A vertex v is *free* if it is incident with no edge in M.

* Received by the editors November 14, 1972, and in revised form April 23, 1973. This research was supported in part by the National Science Foundation under Grants NSF GJ96, GP-25081 and GJ474.

† Department of Computer Science, Cornell University, Ithaca, New York, 14850.

‡ Computer Science Department, University of California, Berkeley, California 94720.

A path (without repeated vertices)

$$P = (v_1, v_2), (v_2, v_3), \cdots, (v_{2k-1}, v_{2k})$$

is called an *augmenting path* if its endpoints v_1 and v_{2k} are both free, and its edges are alternatively in $E - M$ and in M; i.e.,

$$P \cap M = \{(v_2, v_3), (v_4, v_5), (v_6, v_7), \cdots, (v_{2k-2}, v_{2k-1})\}.$$

When no ambiguity is possible, we let P denote the set of edges in an augmenting path P as well as the sequence of edges which is the path itself. If S and T are sets, then $S \oplus T$ denotes the symmetric difference of S and T, and $S - T$ denotes the set of elements in S which are not in T. If S is a finite set, then $|S|$ denotes the cardinality of S.

LEMMA 1. *If M is a matching and P is an augmenting path relative to M, then $M \oplus P$ is a matching, and $|M \oplus P| = |M| + 1$.*

Figure 1 denotes a graph G with a matching M and augmenting path P along with the matching $M \oplus P$.

THEOREM 1. *Let M and N be matchings. If $|M| = r$, $|N| = s$ and $s > r$, then $M \oplus N$ contains at least $s - r$ vertex-disjoint augmenting paths relative to M.*

Proof. Consider the graph $\bar{G} = (V, M \oplus N)$ with vertex set V and edge set $M \oplus N$. Since M and N are matchings, each vertex is indicent with at most one edge from $N - M$ and at most one edge from $M - N$; hence each (connected) component of $\bar{G}$ is either

(i) an isolated vertex,

(ii) a cycle of even length, with edges alternatively in $M - N$ and in $N - M$, or

(iii) a path whose edges are alternatively in $M - N$ and in $N - M$.

Let the components of $\bar{G}$ be $C_1, C_2, \cdots, C_g$, where $C_i = (V_i, E_i)$. Let $\delta(C_i) = |E_i \cap N| - |E_i \cap M|$. Then $\delta(C_i) \in \{-1, 0, 1\}$, and $\delta(C_i) = 1$ if and only if C_i is an augmenting path relative to M.

$$\sum_i \delta(C_i) = |N - M| - |M - N| = |N| - |M| = s - r.$$

Hence there are at least $s - r$ components C_i of $\bar{G}$ such that $\delta(C_i) = 1$. These components are vertex-disjoint, and each is an augmenting path relative to M.

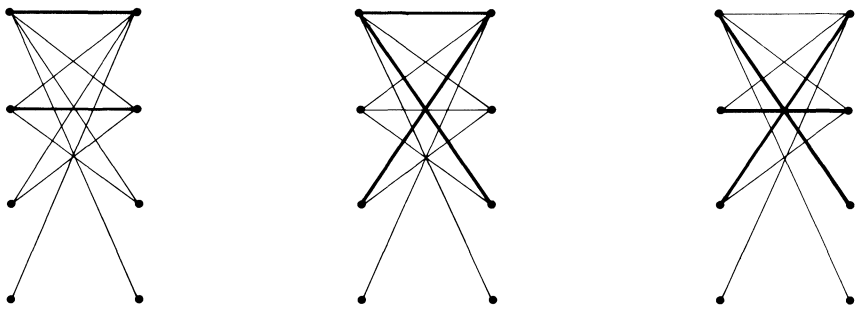

FIG. 1. *Graph G with* (a) *matching M,* (b) *augmenting path P and* (c) *new matching $M \oplus P$ in dark edges*

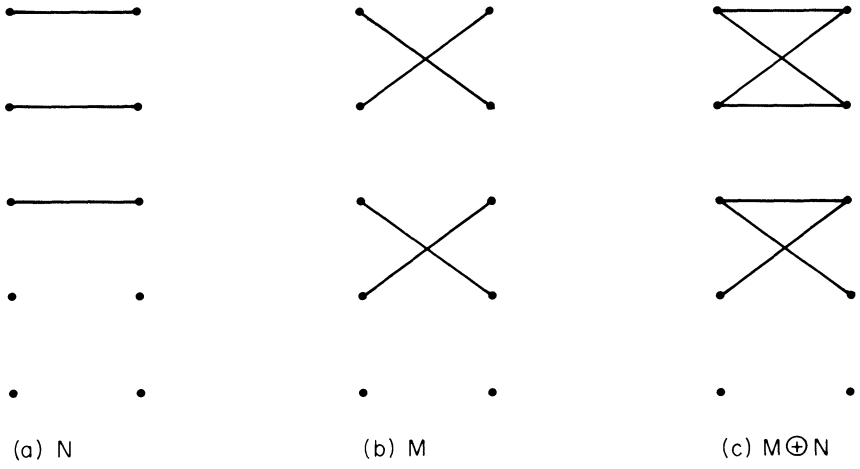

FIG. 2. *Matchings. N, M and M ⊕ N in a graph G. Edges of G are not shown.*

An example of matchings N and M in a graph $G = (V, E)$ along with the graph $(V, N \oplus M)$ is given in Fig. 2.

COROLLARY 1 (Berge). *M is a maximum matching if and only if there is no augmenting path relative to M.*

COROLLARY 2. *Let M be a matching. Suppose $|M| = r$, and suppose that the cardinality of a maximum matching is s, $s > r$. Then there exists an augmenting path relative to M of length $\leqq 2\lfloor r/(s-r)\rfloor + 1$.*[1]

Proof. Let N be a maximum matching. Then $M \oplus N$ contains $s - r$ vertex-disjoint (and hence edge-disjoint) augmenting paths relative to M. Altogether these contain at most r edges from M, so one of them must contain at most $\lfloor r/(s-r)\rfloor$ edges from M, and hence at most $2\lfloor r/(s-r)\rfloor + 1$ edges altogether.

Let M be a matching. The augmenting path P is called *shortest* relative to M if P is of least cardinality among augmenting paths relative to M.

THEOREM 2. *Let M be a matching, P a shortest augmenting path relative to M, and P' an augmenting path relative to $M \oplus P$. Then*

$$|P'| \geqq |P| + |P \cap P'|.$$

Proof. Let $N = M \oplus P \oplus P'$. Then N is a matching and $|N| = |M| + 2$, so $M \oplus N$ contains two vertex-disjoint augmenting paths relative to M; call them P_1 and P_2. Since $M \oplus N = P \oplus P'$, $|P \oplus P'| \geqq |P_1| + |P_2|$. But $|P_1| \geqq |P|$ and $|P_2| \geqq |P|$, since P is a shortest augmenting path. So $|P \oplus P'| \geqq |P_1| + |P_2| \geqq 2|P|$, and also we have the identity $|P \oplus P'| = |P| + |P'| - |P \cap P'|$. Hence $|P'| \geqq |P| + |P \cap P'|$.

We envisage the following scheme of computation: starting with a matching $M_0 = \varnothing$, compute a sequence $M_0, M_1, M_2, \cdots, M_i, \cdots$, where P_i is a shortest augmenting path relative to M_i, and $M_{i+1} = M_i \oplus P_i$.

[1] The symbol $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x, and $\lceil x \rceil$ denotes the least integer greater than or equal to x.

COROLLARY 3. $|P_i| \leqq |P_{i+1}|$.

COROLLARY 4. *For all i and j such that $|P_i| = |P_j|$, P_i and P_j are vertex-disjoint.*

Proof. Suppose, for contradiction, that $|P_i| = |P_j|$, $i < j$, and P_i and P_j are not vertex-disjoint. Then there exist k and l such that $i \leqq k < l \leqq j$, P_k and P_l are not vertex-disjoint, and for each m, $k < m < l$, P_m is vertex-disjoint from P_k and P_l. Then P_l is an augmenting path relative to $M_k \oplus P_k$, so $|P_l| \geqq |P_k| + |P_k \cap P_l|$. But $|P_l| = |P_k|$, so $|P_k \cap P_l| = 0$. Thus P_k and P_l have no edges in common. But if P_k and P_l had a vertex v in common, they would have in common that edge incident with v which is in $M_k \oplus P_k$. Hence P_k and P_l are vertex-disjoint, and a contradiction is obtained.

THEOREM 3. *Let s be the cardinality of a maximum matching. The number of distinct integers in the sequence*

$$|P_0|, |P_1|, \cdots, |P_i|, \cdots$$

is less than or equal to $2\lfloor\sqrt{s}\rfloor + 2$.

Proof. Let $r = \lfloor s - \sqrt{s}\rfloor$. Then $|M_r| = r$ and, by Corollary 2,

$$|P_r| \leqq 2\lfloor s - \sqrt{s}\rfloor/(s - \lfloor s - \sqrt{s}\rfloor) + 1 \leqq 2\lfloor\sqrt{s}\rfloor + 1.$$

Thus, for each $i < r$, $|P_i|$ is one of the $\lfloor\sqrt{s}\rfloor + 1$ positive odd integers less than or equal to $2\lfloor\sqrt{s}\rfloor + 1$. Also $|P_{r+1}|, \cdots, |P_s|$ contribute at most $s - r = \lceil\sqrt{s}\rceil$ distinct integers, and the total number of distinct integers is less than or equal to $\lfloor\sqrt{s}\rfloor + 1 + \lceil\sqrt{s}\rceil \leqq 2\lfloor\sqrt{s}\rfloor + 2$, and the proof is complete.

In view of Corollaries 3 and 4 and Theorem 3, the computation of the sequence $\{M_i\}$ breaks into at most $2\lfloor\sqrt{s}\rfloor + 2$ phases, within each of which all the augmenting paths found are vertex-disjoint and of the same length. Since these paths are vertex-disjoint, they are all augmenting paths relative to the matching with which the phase is begun. This gives us an alternative way of describing the computation of a maximum matching.

ALGORITHM A (Maximum matching algorithm).

Step 0. $M \leftarrow \varnothing$.

Step 1. Let $l(M)$ be the length of a shortest augmenting path relative to M. Find a maximal[2] set of paths $\{Q_1, Q_2, \cdots, Q_t\}$ with the properties that

(a) for each i, Q_i is an augmenting path relative to M and $|Q_i| = l(M)$;

(b) the Q_i are vertex-disjoint.

Halt if no such paths exist.

Step 2. $M \leftarrow M \oplus Q_1 \oplus Q_2 \oplus \cdots \oplus Q_t$; go to 1.

COROLLARY 5. *If the cardinality of a maximum matching is s, then Algorithm* A *constructs a maximum matching within* $2\lfloor\sqrt{s}\rfloor + 2$ *executions of Step* 1.

This way of describing the construction of a maximum matching suggests that we should not regard successive augmentation steps as independent computations, but should concentrate instead on the efficient implementation of an entire phase (i.e., the execution of Step 1 in Algorithm A). The next section shows the advantage of this approach in the case where G is a bipartite graph.

[2] A set is *maximal* with a given property if it has the property and is not properly contained in any set that has the property.

3. The bipartite case. The graph $G = (V, E)$ is *bipartite* if the set of vertices V can be partitioned into two sets, X and Y, such that each edge of G joins a vertex in X with a vertex in Y. An element of X will be called a *boy*, and an element of Y, a *girl*. Let M be a matching in a bipartite graph G. We discuss the implementation of Step 1 of Algorithm A, in which a maximal vertex-disjoint set of shortest augmenting paths relative to M is found. First we assign directions to the edges of G in such a way that augmenting paths relative to M become directed paths. This is done by directing each edge in $E - M$ so that it runs from a girl to a boy, and each edge in M so that it runs from a boy to a girl. The resulting directed graph is $\bar{G} = (V, \bar{E})$, where

$$\bar{E} = \{(y, x) | \{x, y\} \in E - M, x \in X, y \in Y\} \cup \{(x, y) | \{x, y\} \in M, x \in X, y \in Y\}.$$

Next we extract a subgraph $\hat{G}$ of $\bar{G}$, with the property that the directed paths of $\hat{G}$ running from a free girl to a free boy correspond one-to-one to the shortest augmenting paths in G relative to M. This is done as follows.

Let L_0 be the set of free boys, and let

$$E_i = \{(u, v) | (u, v) \in \bar{E}, v \in L_i, u \notin L_0 \cup L_1 \cup \cdots \cup L_i\}, \qquad i = 0, 1, 2, \cdots,$$

$$L_{i+1} = \{u | \text{ for some } v, (u, v) \in E_i\}, \qquad i = 0, 1, 2, \cdots.$$

Let $i^* = \min \{i | L_i \cap \{\text{free girls}\} \neq \varnothing\}$.

Then $\hat{G} = (\hat{V}, \hat{E})$, where

$$\hat{V} = L_0 \cup L_1 \cup \cdots \cup L_{i^*-1} \cup (L_{i^*} \cap \{\text{free girls}\}),$$

$$\hat{E} = E_0 \cup E_1 \cup \cdots \cup E_{i^*-2} \cup \{(u, v) | v \in L_{i^*-1} \text{ and } u \in \{\text{free girls}\}\}.$$

An example of a graph $\hat{G}$ is given in Fig. 3.

The following properties of $\hat{G}$ are immediate.

(i) The vertices at even levels $(L_0, L_2, \cdots)$ are boys, and those at odd levels are girls.

(ii) If $(u, v) \in \hat{E}$, then for some i, $u \in L_{i+1}$ and $v \in L_i$.

(iii) $\hat{G}$ is acyclic.

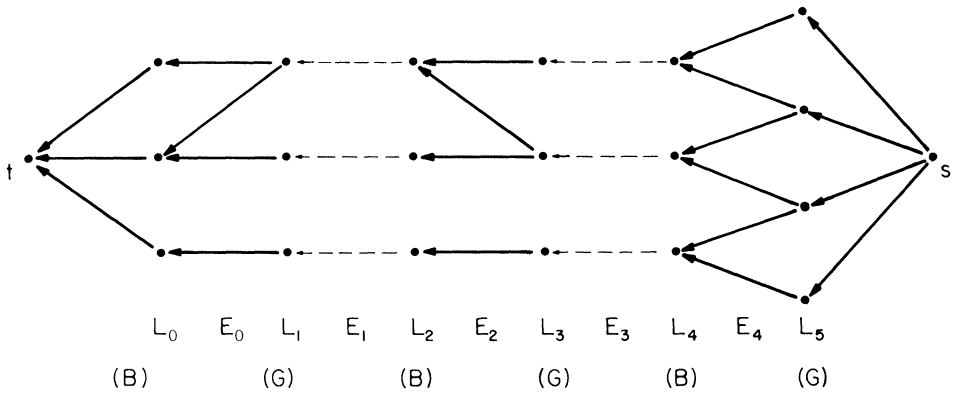

FIG. 3. *The graph $\hat{G}$ used by Algorithm* B. *Dotted arrows indicate edges in current matching.*

(iv) The shortest augmenting paths relative to M are in one-to-one correspondence with the paths of $\hat{G}$ which begin at a free girl and end at a free boy. These paths are all of length i^*.

For convenience, adjoin to $\hat{G}$ two new vertices, s (the source) and t (the sink); adjoin an edge from s to every free girl in $\hat{V}$, and an edge to t from every free boy in $\hat{V}$. Then we seek a maximal set of paths from s to t, subject to the restriction that the paths are vertex-disjoint except for their endpoints.

We give an algorithm to find a maximal vertex-disjoint (except for endpoints) set of paths from s to t in an arbitrary acyclic directed graph H. The mechanism for finding a maximal set of paths is a straightforward depth-first search. Each edge processed either becomes part of the path being constructed from s to t, or else there is no s-t path using that edge. In either case, the edge need never be examined again and so is deleted.

We assume that the graph is represented as follows: for each vertex u, a read-only linear list LIST(u) is given containing, in an arbitrary order, the vertices v such that (u, v) is an edge. The algorithm also uses an auxiliary last-in first-out list called STACK, which is initially empty, and a set B of vertices which is initially the empty set. The following primitives occur in the algorithm.

Variables

TOP	top element of STACK
FIRST	

Operations

PUSH x	push element x onto STACK
POP	pop an element from STACK
DELETE	delete the first element from LIST(TOP)
PRINT	POP until STACK is empty and print the successive elements

Predicates

EMPTY	STACK is empty
NULL	LIST(TOP) is empty

ALGORITHM B (maximal set of s-t paths).

```
PUSH s
while STACK ¬ EMPTY do
begin
* while LIST(TOP) ¬ NULL do
  begin
    FIRST = first element of LIST(TOP)
    if FIRST ∉ B then
    begin
      PUSH FIRST
      if TOP ¬ = t then B ← B ∪ {TOP}
        else PRINT, PUSH s
    end
  end
  POP
end
```

We make the following inductive assertions: every time instruction * is executed,

(i) STACK contains the vertex sequence of a path from s to TOP;

(ii) for every vertex $u \neq s, t$, one of the following holds:

(a) $u \in B$ and u is on a s-t path already printed;

(b) $u \in B$ and u is on the stack;

(c) $u \in B$ and u does not occur in any s-t path which is disjoint from the s-t paths already printed;

(d) $u \notin B$, u does not occur on the stack or in any s-t path previously printed, and for every v, $u \in \text{LIST}(v)$ if and only if (v, u) is an edge of the graph H (i.e., u has not been DELETEd from any LIST).

The algorithm terminates when s is POPed from STACK after it is found that LIST(s) is EMPTY. The inductive assertions given above then imply that upon termination, no s-t path exists disjoint from those already PRINTed.

Each while block in the algorithm contains either a POP or a DELETE operation. Since no vertex is POPed more than once, or DELETEd from any LIST more than once, the running time of the algorithm is bounded by a constant times (number of vertices + number of edges).

Hence the execution of Step 1 of Algorithm A requires at most $O(m + n)$ operations, and the execution of the entire maximum matching algorithm requires at most $O((m + n)\sqrt{s}) = O(n^{5/2})$ steps.

REFERENCES

[1] C. BERGE, *Two theorems in graph theory*, Proc. Nat. Acad. Sci. U.S.A., 43 (1957), pp. 842–844.

[2] J. EDMONDS, *Paths, trees and flowers*, Canad. J. Math., 17 (1965), pp. 449–467.

[3] L. R. FORD AND D. R. FULKERSON, *Flows in Networks*, Princeton University Press, Princeton, N.J., 1965.

[4] M. HALL, *Distinct representatives of subsets*, Bull. Amer. Math. Soc., 54 (1948), pp. 922–926.

[5] H. W. KUHN, *The Hungarian method for the assignment problem*, Naval Res. Logist. Quart., 2 (1955), pp. 83–97.