Annals of Discrete Mathematics 2 (1978) 91–106.

SUBTREE ISOMORPHISM IN $O(n^{5/2})$

David W. MATULA
Department of Computer Science, Southern Methodist University, Dallas, TX 75275, U.S.A.

The problem of determining if the tree S (unrooted) on n_s vertices is isomorphic to any subtree of the tree T on $n_t \geq n_s$ vertices is shown to be solvable in $O(n_t^{3/2} n_s)$ steps. The method involves the solution of an $(n_s - 1)$ by $2(n_t - 1)$ array of maximum bipartite matching problems where some of these subproblems are solved in groups. Recognition of isomorphic subproblems yields a compacted data structure reducing practical storage requirements with no increase in the order of time complexity.

1. Introduction and summary

Given a tree T on n_t vertices and a tree S on n_s vertices, the *subtree isomorphism problem* for trees asks if T possesses a subtree isomorphic to S. If $n = n_t = n_s$, this is simply the tree isomorphism problem which is equivalent to the determination of a canonical naming procedure for trees. The inherent recursive structure of rooted trees can be employed to generate a naming procedure for rooted trees as illustrated in the method of Edmonds [2, pp. 196–199]. Along with a canonical root placement procedure (e.g. at a center vertex) a solution of the (unrooted) tree isomorphism problem is achieved. An algorithm employing these procedures to resolve tree isomorphism in time complexity $O(n)$ has been described by Hopcroft and Tarjan [6, pp. 140–142].

As noted by Edmonds [2, p. 199], this solution to the tree isomorphism problem yields no obvious efficient extension for solving the subtree isomorphism problem, particularly since the maximum number of subtrees of an n vertex tree grows exponentially with n.

A polynomial bounded procedure for solving the subtree isomorphism problem was independently discovered by Edmonds [3] and Matula [7] in 1968 and has been noted by several other researchers since then. The Edmonds–Matula procedure translates the initial subtree isomorphism problem into a polynomially bounded collection of recursively smaller subtree isomorphism problems. Each of these subproblems is shown to be solvable in appropriate order as a maximum bipartite matching problem (equivalently: system of distinct representatives problem) for which polynomial bounded algorithms are known. Edmond's [3] solution involved use of the weighted maximum bipartite matching problem (equivalently: assignment problem) and resolved the more general question of determining the largest subtree of T isomorphic to a subtree of S. These early solutions stressed the polynomial bounded nature of the procedures rather than detailed complexity

analysis. The main purpose of this paper is to pursue further efficiencies achievable in subproblem organization and solution and improve and document a worst case complexity bound. In particular, a method for efficiently solving groups of subproblems by an extension of the maximum bipartite matching algorithm is introduced and shown to yield a subtree isomorphism algorithm having time complexity $O(n_t^{3/2} n_s)$.

In contrast to this complexity result for determining particular subtrees of a tree, it should be noted that the subtree isomorphism problem formulated for a general graph is reducible to the Hamiltonian cycle problem and therefore is NP-complete. Furthermore, Garey, Johnson, and Tarjan [4] have shown that the Hamiltonian cycle problem for a planar graph is NP-complete, so that the determination of whether a planar graph P possesses a subtree isomorphic to a particular tree S is also an NP-complete problem.

2. The subtree isomorphism problem

A graph $G = (V, E)$ is composed of a finite non-void vertex set V and an edge set E where each edge $e \in E$ is a distinct unordered pair $e = v_i v_j$ of vertices of V. The graph $G_1 = (V_1, E_1)$ is isomorphic to the graph $G_2 = (V_2, E_2)$, denoted $\phi G_1 = G_2$, if there is a one-to-one correspondence of the vertices $\phi: V_1 \rightarrow V_2$ that preserves edges, i.e. $\phi(v_i)\phi(v_j) \in E_2$ if and only if $v_i v_j \in E_1$. A tree $T = (V, E)$ is a connected graph without cycles, and a subtree of T is any subgraph of T that is a tree.

The problem of determining if the tree S is isomorphic to any subtree of the tree T is termed the *subtree isomorphism problem for trees.* If S is isomorphic to the subtree T' of T, then the isomorphism $\phi S = T'$ is termed an *imbedding* of S in T. Fig. 1 shows diagrams of a tree S and a particular imbedding of S in the tree T.

A *rooted tree* $T[r]$ is a tree with a particular vertex $r \in V(T)$ designated as the root. If v and u are adjacent vertices of the tree T, then the *limb* $T[v, u]$ denotes the maximal subtree of T containing the edge vu where v is an end vertex of the subtree and is designated as the root of the limb. Intuitively, the limb $T[v, u]$ is the portion of the tree severed at v including all vertices in the direction from v through u. Each edge vu of the tree $T = (V, E)$ thus corresponds to two limbs of T, $T[v, u]$ and $T[u, v]$, so T has a total of $2|E|$ limbs. In Fig. 1 the limb $T[a, b]$ of T is the whole tree T rooted at a, and $T[b, a]$ is the limb consisting of the single edge ba with root vertex b. For a rooted tree $T[v]$, the limbs of $T[v]$ are the limbs $T[u, w]$ in which w is farther than u from the root. Thus all limbs of a rooted tree are oriented away from the root, and there are simply $|E|$ limbs of the rooted tree $T[v]$. Since $|E| = |V| - 1$ for any tree, the following result is obtained.

Lemma 2.1. *Let $l(T[v])$ be the set of limbs of the tree $T = (V, E)$ rooted at $v \in V$, and let $l(T)$ be the set of limbs of the (unrooted) tree T. Then with $n_t = |V|$,*

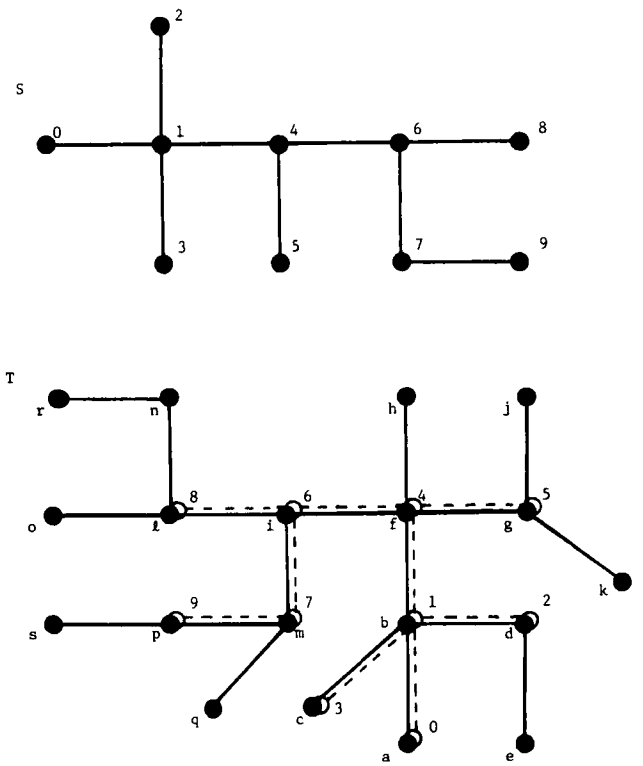


Fig. 1. Diagrams of a tree S on 10 vertices and a tree T on 19 vertices illustrating a particular imbedding of S in T.

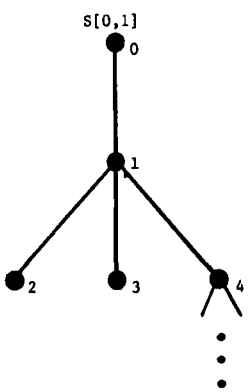
(i) $|l(T[v])| = n_t - 1$ *for any* $v \in V$,
(ii) $|l(T)| = 2(n_t - 1)$,
(iii) $l(T) = \bigcup_{v \in V} l(T[v])$.

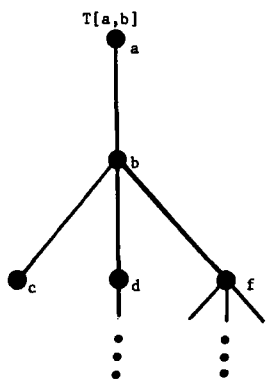
Although each of the individual n_t placements of the root in T yields $(n_t - 1)$ distinct limbs for that rooted tree, the simple fact of Lemma 2.1 that the collection of all limbs derived from all placements of the root results in only $2(n_t - 1)$ distinct limbs underlies an important simplification in our solution of the subtree isomorphism problem.

The rooted tree $S[x]$ is taken to be isomorphic to the rooted tree $T[v]$ only if there is an isomorphism of S to T which takes x into v. An isomorphism of the limb $S[x, y]$ to a rooted subtree of the limb $T[v, u]$ then must take x to v and y to u, and is termed a *limb imbedding* of $S[x, y]$ in $T[v, u]$. The original subtree isomorphism problem asking if the tree S can be imbedded in the tree T can be recast as a limb imbedding problem as follows. Choose a limb $S[x, y]$ of S where x is an end vertex of S, so that $S[x, y]$ contains all vertices of S. Then S can be imbedded in T if and only if the particular limb $S[x, y]$ can be imbedded in some limb of T. For the example of Fig. 1, note that S can be imbedded in T if and only if $S[0, 1]$ can be

imbedded in some limb of T, in particular the limb $T[a, b]$ suffices to establish the imbedding.

The general question of whether a rooted tree is isomorphic to a subtree of another rooted tree was noted independently by Edmonds [3] and Matula [7] to be reducible recursively to the solution of a polynomial bounded collection of maximum bipartite matching problems. The essential recursion of the Edmonds–Matula procedure applied to the determination of whether the limb $S[x, y]$ can be imbedded in the limb $T[v, u]$ proceeds as follows. Let $a_1, a_2, \ldots, a_p$ be the vertices other than x adjacent to y in $S[x, y]$, and $b_1, b_2, \ldots, b_q$ the vertices other than v adjacent to u in $T[v, u]$. The *limb imbedding submatrix* associated with $S[x, y]$ and $T[v, u]$ has rows corresponding to the limbs $S[y, a_i]$, $1 \leq i \leq p$, and columns corresponding to the limbs $T[u, b_j]$, $1 \leq j \leq q$, with the $S[y, a_i]$, $T[u, b_j]$ position of value unity if $S[y, a_i]$ can be imbedded in $T[u, b_j]$ and zero otherwise. Fig. 2 illustrates the limb imbedding submatrix associated with the limbs $S[0, 1]$ and $T[a, b]$ of the trees S and T of Fig. 1.





Limb imbedding submatrix of $S[0, 1]$ and $T[a, b]$:

	$T[b, c]$	$T[b, d]$	$T[b, f]$
$S[1, 2]$	1	(1)	1
$S[1, 3]$	(1)	1	1
$S[1, 4]$	0	0	(1)

Fig. 2.

In general, for a $p \times q$ 0, 1-matrix with $p \leq q$, a *bipartite matching* is a set of unit entry positions no two from the same column or from the same row. A maximum bipartite matching is *complete for the rows* if there are p unit entry positions in the bipartite matching, one for each row, in which case these unit positions are said [8, Ch. 5] to identify a *system of distinct representatives* of columns for rows. The circled entries in the limb imbedding submatrix of Fig. 2 denote a particular maximum bipartite matching that corresponds to the imbedding of the limb $S[0, 1]$ in the limb $T[a, b]$ illustrated in Fig. 1.

Theorem 2.2 [3, 7]. *The limb* $S[x, y]$ *is isomorphic to a rooted subtree of the limb* $T[v, u]$ *if and only if the associated limb imbedding submatrix has a maximum bipartite matching which is complete for the rows, i.e. a system of distinct representatives of columns for rows.*

Proof. Let $a_1, a_2, \ldots, a_p$ be the vertices other than x adjacent to y in $S[x, y]$, and $b_1, b_2, \ldots, b_q$ the vertices other than v adjacent to u in $T[v, u]$. Let $S[x, y]$ be isomorphic to a subtree of $T[v, u]$ under the mapping ϕ. Thus $\phi(x) = v$, $\phi(y) = u$, and $\phi(a_i) = b_{j_i}$ for $1 \leq i \leq p$, where the indices $j_1, j_2, \ldots, j_p$ are distinct integers of $\{1, 2, \ldots, q\}$. The mapping ϕ identifies an isomorphism of the limb $S[y, a_i]$ with a rooted subtree of the limb $T[u, b_{j_i}]$, so the $S[y, a_i]$, $T[u, b_{j_i}]$ position of the associated limb imbedding submatrix is unity for $1 \leq i \leq p$, thus determining a maximum bipartite matching that is complete for the rows.

For the converse let the associated limb imbedding submatrix for $S[x, y]$ and $T[v, u]$, have a bipartite matching which is complete for the rows and is identified by $S[y, a_i]$, $T[u, b_{j_i}]$ for $1 \leq i \leq p$. Then there exist imbeddings ϕ_i taking $S[y, a_i]$ into $T[u, b_{j_i}]$ for each i. The ϕ_i can be consistently combined to yield an imbedding ϕ of $S[x, y]$ into $T[v, u]$ by defining $\phi(x) = v$, and $\phi(w) = \phi_i(w)$ for all $w \in V(S[y, a_i])$ for $1 \leq i \leq p$, completing the theorem.

The *height* of a limb $T[v, u]$ denotes the largest distance of any vertex of $T[v, u]$ from the root vertex v. Thus Theorem 2.2 implies that the limb imbedding problem is reducible via a maximum bipartite matching problem to the solution of a collection of limb imbedding problems on limbs of smaller height. To see that the total number of limb imbedding problems that must be solved by recursion is manageable, the following 0, 1-matrix is introduced.

Let $S[x, y]$ be a limb on n_s vertices and T a tree on n_t vertices. The *limb imbedding matrix* $L(S[x, y], T)$ has for rows the $n_s - 1$ limbs of the rooted tree $S[x, y]$ and for columns the $2(n_t - 1)$ limbs of the unrooted tree T. For all limb pairs $S[a, b]$ of $S[x, y]$ and $T[v, u]$ of T, the entry for the $S[a, b]$, $T[v, u]$ position is unity if $S[a, b]$ can be imbedded in $T[v, u]$ and zero otherwise. A *standard form* for the limb imbedding matrix $L(S[x, y], T)$ has the rows ordered by non-decreasing height and the columns grouped so that all limbs $T[v, u]$ with the same second vertex u are in sequence. The limb imbedding matrix $L(S[0, 1], T)$ for the trees of Fig. 1 in such a standard form is shown in Fig. 3.

	$S[7,9]$	$S[6,8]$	$S[4,5]$	$S[1,3]$	$S[1,2]$	$S[6,7]$	$S[4,6]$	$S[1,4]$	$S[0,1]$
$T[p,s]$	1	1	1	1	1	0	0	0	0
$T[n,r]$	1	1	1	1	1	0	0	0	0
$T[m,q]$	1	1	1	1	1	0	0	0	0
$T[s,p]$	1	1	1	1	1	1	1	0	0
$T[m,p]$	1	1	1	1	1	1	0	0	0
$T[l,o]$	1	1	1	1	1	0	0	0	0
$T[r,n]$	1	1	1	1	1	1	0	0	0
$T[l,n]$	1	1	1	1	1	1	0	0	0
$T[q,m]$	1	1	1	1	1	1	1	1	0
$T[p,m]$	1	1	1	1	1	1	1	1	0
$T[i,m]$	1	1	1	1	1	1	1	0	0
$T[o,l]$	1	1	1	1	1	1	1	1	0
$T[n,l]$	1	1	1	1	1	1	1	1	0
$T[i,l]$	1	1	1	1	1	1	1	0	0
$T[g,k]$	1	1	1	1	1	0	0	0	0
$T[g,j]$	1	1	1	1	1	0	0	0	0
$T[m,i]$	1	1	1	1	1	1	1	1	0
$T[l,i]$	1	1	1	1	1	1	1	1	0
$T[f,i]$	1	1	1	1	1	1	1	1	0
$T[f,h]$	1	1	1	1	1	0	0	0	0
$T[k,g]$	1	1	1	1	1	1	1	1	0
$T[j,g]$	1	1	1	1	1	1	1	1	0
$T[f,g]$	1	1	1	1	1	1	0	0	0
$T[i,f]$	1	1	1	1	1	1	1	1	0
$T[h,f]$	1	1	1	1	1	1	1	1	1
$T[g,f]$	1	1	1	1	1	1	1	1	1
$T[b,f]$	1	1	1	1	1	1	1	1	1
$T[d,e]$	1	1	1	1	1	0	0	0	0
$T[e,d]$	1	1	1	1	1	1	0	0	0
$T[b,d]$	1	1	1	1	1	1	0	0	0
$T[b,c]$	1	1	1	1	1	0	0	0	0
$T[f,b]$	1	1	1	1	1	1	1	0	0
$T[d,b]$	1	1	1	1	1	1	1	1	0
$T[c,b]$	1	1	1	1	1	1	1	1	1
$T[a,b]$	1	1	1	1	1	1	1	1	1
$T[b,a]$	1	1	1	1	1	0	0	0	0

Fig. 3. Limb imbedding matrix $L(S[0,1], T)$ for the trees S and T of Fig. 1.

As previously noted the determination of whether limb $S[x, y]$ could be imbedded in limb $T[v, u]$ required as input to a maximum bipartite matching problem the solution of a collection of other limb imbedding problems involving limbs of $T[v, u]$ and limbs of $S[x, y]$ of smaller height. Inspection of the limb imbedding matrix $L(S[x, y], T)$ reveals that the solution of each particular limb imbedding problem corresponding to an entry in the matrix requires the solution of a collection of other limb imbedding problems which *are all in the matrix in preceding rows* in any standard form. Thus the total number of subproblems is manageable and they can be computed in an appropriate order as now formalized in Algorithm A.

Algorithm A — Subtree isomorphism

Given a tree T on n_t vertices and a tree S on n_s vertices, this algorithm determines if there is a subtree of T isomorphic to S.

A1. Root S at an end vertex x determining the limb $S[x, y]$, and order the $(n_s - 1)$ limbs of $S[x, y]$ by non-decreasing height.

A2. For each limb of $S[x, y]$ of height 1, set all entries in the corresponding row of the limb imbedding matrix $L(S[x, y], T)$ to unity.

A3. Let h step by 1 from 2 through *height*$(S[x, y])$. For each limb $S[a, b]$ of $S[x, y]$ of height h and each limb $T[v, u]$ of T, determine the maximum bipartite matching for the limb imbedding submatrix associated with $S[a, b]$ and $T[v, u]$. If this maximum bipartite matching is complete for the rows of the limb imbedding submatrix, set the value of the $S[a, b]$, $T[v, u]$ position of $L(S[x, y], T)$ to unity, otherwise set that position to zero.

A4. If there is a unit entry in the final row of $L(S[x, y], T)$, then there is a subtree of T isomorphic to S, otherwise no such subtree exists.

Theorem 2.3. *Given trees T and S, Algorithm A determines if there is a subtree of T isomorphic to S.*

Proof. The question of whether T has a subtree isomorphic to S is equivalent to whether the limb $S[x, y]$ rooted at an end vertex x of S as specified in step A1 can be imbedded in some limb of T. It will now be shown by induction on the height of the limbs of $S[x, y]$ that steps A2 and A3 compute the limb imbedding matrix $L(S[x, y], T)$. To initiate the induction note that a limb $S[a, b]$ of $S[x, y]$ of radius 1 can be imbedded in any limb of T, so step A2 properly computes the entries of $L(S[x, y], T)$ corresponding to limbs of $S[x, y]$ of height 1.

Now assume the entries of $L(S[x, y], T)$ are properly computed for all entries in all rows corresponding to limbs of $S[x, y]$ of height less than i for some $i \geq 2$, and let the limb $S[a, b]$ of $S[x, y]$ have height i. Consider the position $S[a, b]$, $T[v, u]$ of $L(S[x, y], T)$ for any limb $T[v, u]$ of T. At the time this position is considered for assignment of a value by step A3, the associated limb imbedding submatrix for

$S[a, b]$ and $T[v, u]$ is required. Note that the limbs $S[b, c_i]$, $i = 1, 2, \ldots,$ degree $(b) - 1$, $c_i \neq a$, of $S[a, b]$ are also limbs of $S[x, y]$ having height at most $i - 1$, and the limbs $T[u, d_j]$, $j = 1, 2, \ldots,$ degree $(u) - 1$, $d_j \neq v$, of $T[v, u]$ are also limbs of T, so the entries in the limb imbedding submatrix for $S[a, b]$ and $T[v, u]$ are all available in the previously computed and assigned portion of the limb imbedding matrix $L(S[x, y], T)$. Step A3 assigns the position $S[a, b]$, $T[v, u]$ of $L(S[x, y], T)$ the value unity if the limb imbedding submatrix for $S[a, b]$ and $T[v, u]$ has a maximum bipartite matching that is complete for the rows and the value zero otherwise, and by Theorem 2.2 this is equivalent to assigning unity if $S[a, b]$ can be imbedded in $T[v, u]$ and zero otherwise. Thus the entry for the position $S[a, b]$, $T[v, u]$ of the limb imbedding matrix $L(S[x, y], T)$ is properly computed for all entries in all rows corresponding to limbs of $S[x, y]$ of radius i, and by the induction assumption the whole limb imbedding matrix $L(S[x, y], T)$ is then properly computed by steps A2 and A3 of Algorithm A. The final step A4 then resolves from the limb imbedding matrix the question of whether T has a subtree isomorphic to S, and the algorithm is verified.

Algorithm A as stated simply determines if there exists a subtree of T isomorphic to S. It is straightforward by the following computation to actually determine a specific subtree of T isomorphic to S and to identify the isomorphism. If the solution of all maximum bipartite matching problems for limb imbedding submatrices that yield unit entries in $L(S[x, y], T)$ are preserved, then one may simply retrace the appropriate bipartite matchings starting from a unit in the $S[x, y]$ row of $L(S[x, y], T)$ to determine the subtree of T and its isomorphic mapping to S. If it is desirable to utilize available storage to provide for the solution of larger problems, then it is sufficient to preserve just the matrix $L(S[x, y], T)$, since the appropriate $(n_s - 1)$ maximum bipartite matching problems may be recomputed efficiently in a top-down order corresponding to a breadth first search of $S[x, y]$ commencing from the root x and utilizing the known values of the matrix $L(S[x, y], T)$.

To realize further storage efficiency a more compact data structure for representing $L(S[x, y], T)$ is now described. Note that the determination of whether or not the limb $S[a, b]$ can be imbedded in the limb $T[v, u]$ is dependent only on the isomorphism types of $S[a, b]$ and $T[v, u]$, thus rows and/or columns of the limb imbedding matrix corresponding to isomorphic limbs have identical entries as was seen in Fig. 3. Compacting the matrix to allow only a single row and/or column for each isomorphism type and establishing pointer lists for the limbs of $S[x, y]$ and T results in the *limb imbedding data structure* illustrated in Fig. 4 for the limb imbedding matrix of Fig. 3.

The isomorphism type determination of the limbs of $S[x, y]$ and T can be handled efficiently by well known procedures [1, pp. 84–85]. Thus step 1 of Algorithm A may be expanded to include determination of the isomorphism types of the limbs of $S[x, y]$ and T and establishing the pointer lists for the limb imbedding data structure. This modification will be termed Algorithm A′ for subtree isomorphism.

Limb	Column Pointer
$T[b, a]$	1.1
$T[a, b]$	6.3
$T[c, b]$	6.3
$T[d, b]$	6.2
$T[f, b]$	3.2
$T[b, c]$	1.1
$T[b, d]$	2.1
$T[e, d]$	7.2
$T[d, e]$	1.1
$T[b, f]$	5.2
$T[g, f]$	5.3
$T[h, f]$	5.4

Limb	Column Pointer
$T[i, f]$	4.2
$T[f, g]$	2.2
$T[j, g]$	6.5
$T[k, g]$	6.5
$T[f, h]$	1.1
$T[f, i]$	4.1
$T[l, i]$	5.1
$T[m, i]$	5.1
$T[g, j]$	1.1
$T[g, k]$	1.1
$T[i, l]$	3.1
$T[n, l]$	6.1

Limb	Column Pointer
$T[o, l]$	6.4
$T[i, m]$	3.1
$T[p, m]$	6.1
$T[q, m]$	6.4
$T[l, n]$	2.1
$T[r, n]$	7.1
$T[l, o]$	1.1
$T[m, p]$	2.1
$T[s, p]$	7.1
$T[m, q]$	1.1
$T[n, r]$	1.1
$T[p, s]$	1.1

Limb	Row Pointer
$S[7, 9]$	1
$S[6, 8]$	1
$S[4, 5]$	1
$S[1, 3]$	1
$S[1, 2]$	1
$S[6, 7]$	2
$S[4, 6]$	3
$S[1, 4]$	4
$S[0, 1]$	5

	1.1	2.1	2.2	3.1	3.2	4.1	4.2	5.1	5.2	5.3	5.4	6.1	6.2	6.3	6.4	6.5	7.1	7.2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
4	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0
5	0	0	0	0	0	0	0	0	1	1	1	0	1	1	0	0	0	0

Fig. 4. The isomorphically compacted limb imbedding data structure for the limb imbedding matrix $L(S[0, 1], T)$ of Fig. 3.

The primary sources of significant storage reduction in the process of isomorphically compacting the limb imbedding matrix $L(S[x, y], T)$ are (1) limbs of small height and (2) limbs of T rooted at end vertices that are adjacent to a common vertex of T. Although the average number of non-isomorphic limbs of a tree on n vertices is not known for general n, reduction by 30% to 60% in each dimension of the limb imbedding matrix seems reasonably attainable for a large portion of computationally feasible subtree isomorphism problems. Noting that the limb imbedding matrix is 0, 1 and thus may be stored in bits, an implementation of Algorithm A′ allowing a solution of subtree isomorphism problems for trees with thousands of vertices is storagewise feasible. The question of time complexity of subtree isomorphism is now considered.

3. Complexity of subtree isomorphism

The time complexity of an algorithm shall denote the worst case behavior of the algorithm measured in terms of the number of instruction executions on an abstract random access machine model of typical computers, such as the RAM [1, pp. 5–11]. A cursory examination of Algorithm A indicates that steps A1, A2, and A4 will require at most $O(n_t n_s)$ time, so the total time will be dominated by the solution time for the $2(n_s - 1)(n_t - 1)$ maximum bipartite matching problems encountered in step A3.

An algorithm for determining the maximum bipartite matching in a $p \times q$ 0, 1-matrix has been described by Hopcroft and Karp [5] with the following time complexity result.

Theorem 3.1 [5]. *A maximum bipartite matching in a $p \times q$ 0, 1-matrix with $p \leq q$ can be determined with time complexity $O(q^{3/2}p)$.*

Utilizing the maximum bipartite matching algorithm of Hopcroft and Karp in step A3 of Algorithm A and noting that every limb imbedding submatrix has dimensions less than $n \times n$ for $n = \max\{n_s, n_t\}$, it is immediate that Algorithm A has the time complexity bound $O(n^{9/2})$. A closer inspection of the limb imbedding submatrix sizes yields the following.

Theorem 3.2. *Given the tree T on n_t vertices and the tree S on $n_s \leq n_t$ vertices, the determination of the existence of a subtree of T isomorphic to S is resolved by Algorithm A and can be implemented in time complexity $O(n_t^{5/2} n_s)$.*

Proof. As previously noted, the time complexity for Algorithm A is dominated by the time complexity of the $2(n_s - 1)(n_t - 1)$ maximum bipartite matching problems whose solution is required to determine the limb imbedding matrix $L(S[x, y], T)$. Consider the time complexity for the subproblems necessary to determine the row

corresponding to $S[a, b]$ in $L(S[x, y], T)$. Each of the $2(n_t - 1)$ limb imbedding subproblems has dimensions bounded by $\deg_S(b) \times n_t$ where $\deg_S(b)$ denotes the degree of vertex b in tree S. Thus, from Theorem 3.1, the time complexity for determining this row of $L(S[x, y], T)$ is bounded by $O(n_t^{5/2} \deg_S(b))$. Note that each limb $S[a, b]$ of the $(n_s - 1)$ limbs of $S[x, y]$ corresponds to a distinct vertex b of S. Each maximum bipartite matching subproblem is bounded by the same time complexity function, so the total time complexity to resolve all subproblems for all rows of $L(S[x, y], T)$ is bounded by $O(\sum_{b \in V(S)} n_t^{5/2} \deg_S(b))$. Noting that

$$\sum_{b \in V(S)} \deg_S(b) = 2|E(S)| = 2(n_s - 1) \tag{1}$$

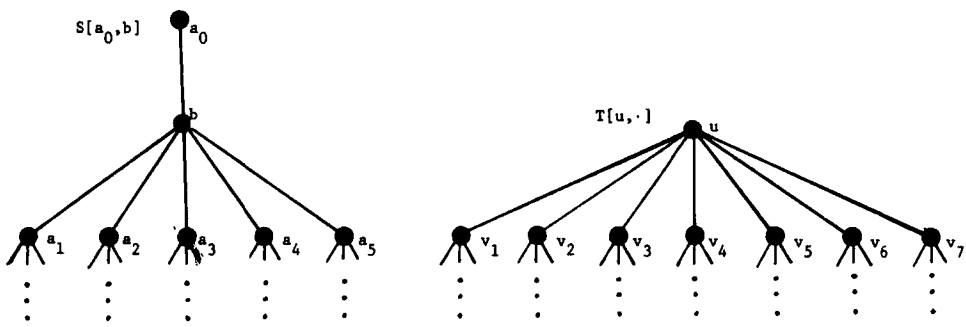
it follows that Algorithm A has the time complexity bound $O(n_t^{5/2} n_s)$.

For each non-root vertex b of $S[x, y]$ and each vertex $u \in V(T)$, there are $\deg_T(u)$ entries in the limb imbedding matrix $L(S[x, y], T)$ corresponding to positions $S[a, b]$, $T[v_i, u]$, $1 \leq i \leq \deg_T(u)$. The questions of whether limb $S[a, b]$ can be imbedded in limb $T[v_i, u]$ are answered separately for each $1 \leq i \leq \deg_T(u)$ in Algorithm A, yielding a necessary time complexity bound $O(\deg_T^{5/2}(u) \deg_S(b))$ for the contribution of solving these subproblems in Algorithm A. It will now be shown that these subproblems are sufficiently related so that this full set of $\deg_T(u)$ limb imbedding subproblems can be solved in $O(\deg_T^{3/2}(u) \deg_S(b))$ by an extension of the solution procedure for the maximum bipartite matching problem.

Let $a_1, a_2, \ldots, a_p$ be the vertices other than a_0 adjacent to b in $S[a_0, b]$, and $v_1, v_2, \ldots, v_q$ the vertices of T adjacent to u. The $(p + 1) \times q$ *composite limb imbedding matrix for* $S[a_0, b]$ *and* $T[u, \cdot]$ has rows corresponding to a_0 and $S[b, a_i]$, $1 \leq i \leq p$, and columns corresponding to $T[u, v_j]$, $1 \leq j \leq q$. The row corresponding to a_0 is termed the root row and all entries in that row are unity. The $S[b, a_i]$, $T[u, v_j]$ position is unity if the limb $S[b, a_i]$ can be imbedded in the limb $T[u, v_j]$ and zero otherwise for $1 \leq i \leq p$, $1 \leq j \leq q$, as illustrated in the example of Fig. 5.

Let the general $(p + 1) \times q$ 0, 1-matrix M have rows $r_0, r_1, \ldots, r_p$, where r_0 is the *root row*, and columns $c_1, c_2, \ldots, c_q$. The *rooted bipartite matching problem* for M denotes the problem of determining those columns composing the *associated root set* $R(M)$, where $c_i \in R(M)$ if and only if there is a maximum matching of M that is complete for the rows in which column c_i is associated with the root row r_0 (i.e. the matching contains the $r_0 c_i$ position of M). For example, the matching denoted by the circled elements in Fig. 5 shows that $T[u, v_3]$ is a member of the associated root set R for the rooted bipartite matching problem for that composite limb imbedding matrix, and further computation yields the full solution $R = \{T[u, v_1], T[u, v_3], T[u, v_4], T[u, v_6], T[u, v_7]\}$.

Lemma 3.3. *The limb* $S[a_0, b]$ *of* $S[x, y]$ *can be imbedded in the limb* $T[v_j, u]$, $1 \leq j \leq \deg_T(u)$, *of the tree* T *if and only if* $T[u, v_j]$ *is a member of the associated root set* $R = \{T[u, v_{j_1}], T[u, v_{j_2}], \ldots, T[u, v_{j_k}]\}$ *of the rooted bipartite matching problem for the composite limb imbedding matrix for* $S[a_0, b]$ *and* $T[u, \cdot]$.



	$T[u, v_1]$	$T[u, v_2]$	$T[u, v_3]$	$T[u, v_4]$	$T[u, v_5]$	$T[u, v_6]$	$T[u, v_7]$
root row $\rightarrow a_0$	1	1	(1)	1	1	1	1
$S[b, a_1]$	(1)	1	0	1	1	0	0
$S[b, a_2]$	0	(1)	0	0	1	0	0
$S[b, a_3]$	0	1	0	(1)	0	1	0
$S[b, a_4]$	0	1	0	0	(1)	0	0
$S[b, a_5]$	0	0	1	0	0	1	(1)

Fig. 5. An example composite limb imbedding matrix for $S[a_0, b]$ and $T[u, \cdot]$.

Proof. For any $1 \leq j \leq \deg_T(u)$, deleting the row corresponding to a_0 and the column corresponding to $T[u, v_j]$ from the composite limb imbedding matrix for $S[a_0, b]$ and $T[u, \cdot]$ yields the limb imbedding submatrix for $S[a_0, b]$ and $T[v_j, u]$. A maximum bipartite matching for the latter matrix that is complete for the rows is then readily identified with the bipartite matching in the composite limb imbedding matrix for $S[a_0, b]$ and $T[u, \cdot]$ having the same unit position choices along with the position for a_0, $T[u, v_j]$. This correspondence along with Theorem 2.2 establishes the Lemma.

Algorithm RM — *Rooted bipartite matching*

Given a $(p+1) \times q$ 0, 1-matrix M with rows $r_0, r_1, \ldots, r_p$ and columns $c_1, c_2, \ldots, c_q$, this algorithm determines the associated root set $R(M)$ of the rooted bipartite matching problem for M.

RM1. If $p < q$, determine a maximum bipartite matching for M. If this matching

is not complete for the rows or if $p \geq q$ then $R(M) = \emptyset$ and the algorithm terminates, otherwise continue.

RM2. Form a bipartite graph $G(M)$ with vertices $\{r_1, r_2, \ldots, r_p\} \cup \{c_1, c_2, \ldots, c_q\}$ and an edge $r_i c_j$ in $G(M)$ for $1 \leq i \leq p$, $1 \leq j \leq q$, if the r_i, c_j position of M is unity. Let the edges corresponding to the matching determined in RM1 (excluding the $r_0 c_k$ pair) be heavy edges of $G(M)$, the others being light edges.

(Fig. 6 illustrates the graph corresponding to the composite limb imbedding matrix and specified matching of Fig. 5.)

RM3. Let R_0 be the set of vertices of $G(M)$ which are not incident to any heavy edges and note that $R_0 \subset \{c_1, c_2, \ldots, c_q\}$. Determine the set R^* of vertices of $\{c_1, c_2, \ldots, c_q\}$ reachable from R_0 by alternating light-heavy chains in $G(M)$ (including chains of length zero). (Note that for the graph illustrated in Fig. 6, $R_0 = \{T[u, v_3], T[u, v_6]\}$ and $R^* = \{T[u, v_1], T[u, v_3], T[u, v_4], T[u, v_6], T[u, v_7]\}$.)

RM4. Then $R(M)$ is the subset of R^* containing all entries $c_j \in R^*$ for which r_0, c_j is a unit entry of M. Thus if the r_0 row of M has all unit entries, $R(M) = R^*$.

Theorem 3.4. *For a* $(p+1) \times q$ 0,1-*matrix* M, *Algorithm RM determines the associated root set* $R(M)$ *for the rooted bipartite matching problem for* M *and can be implemented with time complexity* $O(q^{3/2} p)$.

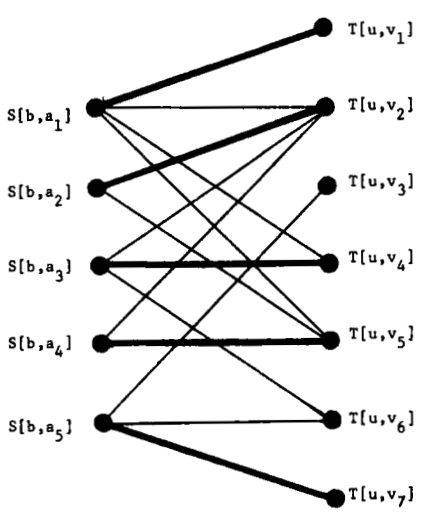


Fig. 6. The graph $G(M)$ determined by step RM2 of Algorithm RM for the matrix and specified matching of Fig. 5.

Proof. We first establish that the set $R(M)$ determined by Algorithm RM is indeed the associated root set of the matrix M. Then the complexity of implementation of each step of Algorithm RM is investigated yielding the result that the full algorithm can be implemented in $O(q^{3/2}p)$.

If M has no maximum bipartite matching that is complete for the rows, then $R(M) = \emptyset$ as noted in step RM1. Otherwise let r_i, c_{j_i}, $i = 0, 1, 2, \ldots, p$ denote the maximum bipartite matching determined in step RM1 that is complete for the rows of M. Then the matching r_i, c_{j_i}, $i = 1, 2, \ldots, p$ is complete for all rows except r_0 of M and does not utilize any element of the set R_0 established in step RM3. Furthermore if c_m can be reached from R_0 by an alternating light-heavy chain of the bipartite graph $G(M)$ defined in step RM2, then reversing the light and heavy edges of the alternating chain yields another matching r_i, c_{k_i}, $i = 1, 2, \ldots, p$, that is complete for all rows except r_0 of M and does not utilize c_m. Thus for each $c_m \in R^*$ as constructed in step RM3, a unit value in the r_0, c_m position of M assures the existence of a maximum bipartite matching complete for the rows of M using the r_0, c_m entry, which satisfies the criteria for membership in the associated root set for M.

Let r_i, c_{m_i}, $i = 0, 1, 2, \ldots, p$ be any maximum bipartite matching that is complete for the rows of M. Form the bipartite graph $G'(M)$ on $\{r_1, r_2, \ldots, r_p\} \cup \{c_1, c_2, \ldots, c_q\}$ having as heavy edges the pairings of the original matching r_i, c_{j_i}, $i = 1, 2, \ldots, p$ that are not in r_i, c_{m_i}, $i = 1, 2, \ldots, p$, and as light edges the pairings of r_i, c_{m_i}, $i = 1, 2, \ldots, p$, that are not in r_i, c_{j_i}, $i = 1, 2, \ldots, p$. An alternating heavy-light chain (possibly of length zero) must exist in $G'(M)$ from c_{m_0} to some c_{m_i} incident to no heavy edge where then $c_{m_i} \in R_0$. Thus there is an alternating light-heavy chain of $G(M)$ from $c_{m_i} \in R_0$ to c_{m_0}, establishing that $c_{m_0} \in R^*$. Thus $R(M)$ as determined in step RM4 is the associated root set for the rooted bipartite matching problem for M, proving the correctness of Algorithm RM.

To establish the time complexity result note that step RM1 can be implemented in $O(q^{3/2}p)$ by Theorem 3.1. The bipartite graph formed in step RM2 has at most qp edges and can be constructed in $O(qp)$. By breadth-first search the set of vertices reachable from a given set of vertices by alternating chains in a bipartite graph with heavy edges corresponding to some matching and other edges light can be found in time proportional to the number of edges, hence $O(qp)$. Step RM4 has a complexity bound $O(q)$, so Algorithm RM determines the associated root set $R(M)$ for the rooted bipartite matching problem for M and can be implemented in time complexity $O(q^{3/2}p)$.

Algorithm B — Subtree isomorphism

Given a tree T on n_t vertices and a tree S on n_s vertices, this algorithm determines if there is a subtree of T isomorphic to S.

B1. Root S at an end vertex x determining the limb $S[x, y]$, and order the $(n_s - 1)$ limbs of $S[x, y]$ by non-decreasing height.

B2. For each limb of $S[x, y]$ of height 1, set all entries in the corresponding row of the limb imbedding matrix $L(S[x, y], T)$ to unity.

B3. Let h step by 1 from 2 through *height*$(S[x, y])$. For each limb $S[a, b]$ of $S[x, y]$ of height h and each vertex u of the tree T, determine, by Algorithm RM, the associated root set R_u of the rooted bipartite matching problem for the composite limb imbedding matrix for $S[a, b]$ and $T[u, \cdot]$. Set the value of the $S[a, b], T[v, u]$ position of $L(S[x, y], T)$ to unity if $T[u, v] \in R_u$, otherwise set that value to zero, for each vertex v adjacent to u in T.

B4. If there is a unity entry in the final row of $L(S[x, y], T)$, then there is a subtree of T isomorphic to S, otherwise no such subtree exists.

Theorem 3.5. *For the tree T on n_t vertices and the tree S on $n_s \leq n_t$ vertices, Algorithm B determines if there is a subtree of T isomorphic to S and can be implemented with time complexity* $O(n_t^{3/2} n_s)$.

Proof. Steps B1, B2 and B4 of Algorithm B are identical to steps A1, A2 and A4 of Algorithm A. The recursive computation of step B3 proceeds in the same order as step A3, and by Lemma 3.3 the resulting assignment of values to positions in the limb imbedding matrix is identical for B3 and A3. Thus the correctness of Algorithm B follows from the correctness of Algorithm A established in Theorem 2.3.

Steps B1, B2 and B4 require only $O(n_t n_s)$ time, so the time complexity of step B3 is dominant. As in the analysis of Algorithm A consider the time necessary to compute the row corresponding to $S[a, b]$ in $L(S[x, y], T)$. For this row it is necessary to solve a rooted bipartite matching problem for the limb imbedding submatrix for $S[a, b]$ and $T[u, \cdot]$ for each vertex $u \in V(T)$ which by Algorithm RM uses $O(\deg_T^{3/2}(u) \deg_S(b))$ time. Since

$$\sum_{u \in V(T)} \deg_T^{3/2}(u) \leq (2|E(T)|)^{3/2} = [2(n_t - 1)]^{3/2}, \tag{2}$$

the row computation complexity bound is $O(n_t^{3/2} \deg_S(b))$. Since there is precisely one row of the limb imbedding matrix for each vertex $b \in V(S)$, $b \neq x$ (root of $S[x, y]$), the total time of step B3 is bounded by $O(n_t^{3/2} \sum_{b \in V(S)} \deg_S(b))$ or, equivalently, $O(n_t^{3/2} n_s)$, completing the theorem.

It should be noted from the proofs of Theorem 3.5 and Theorem 3.4 that the time complexity bound for subtree isomorphism is dominated by the time necessary in step RM1 of Algorithm RM to solve the maximum bipartite matching problem to the following extent. If a maximum bipartite matching algorithm for a $p \times q$ 0, 1-matrix, $p \leq q$, of complexity $O(q^\alpha p^\beta)$, $\alpha \geq 1$, $\beta \geq 1$, were available, then the subtree isomorphism problem would be solvable in $O(n_t^\alpha n_s^\beta)$. Thus improvements in maximum bipartite matching algorithms yielding reductions in $\alpha + \beta$ below 5/2, with $\alpha \geq 1$, $\beta \geq 1$, would provide similar improvements in the complexity bound for subtree isomorphism.

Acknowledgment

I would like to acknowledge David W. Walkup for some helpful discussions leading to the efficient solution of the rooted bipartite matching problem.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms (Addison-Wesley, Reading, MA, 1974).
[2] R.G. Busacker and T.L. Saaty, Finite Graphs and Networks (McGraw-Hill, New York, 1965).
[3] J. Edmonds, personal communication.
[4] M.R. Garey, D.S. Johnson and R.E. Tarjan, The planar Hamiltonian circuit problem is NP-complete, SIAM J. Comp. 5 (1976) 704–714.
[5] J.E. Hopcroft and R.M. Karp, An $n^{5/2}$ Algorithm for maximum matchings in bipartite graphs, SIAM J. Comp. 2 (1973) 225–231.
[6] J.E. Hopcroft and R.E. Tarjan, Isomorphism of planar graphs, in: R.E. Miller and J.W. Thatcher, eds., Complexity of Computer Computations (Plenum, New York, 1972) 131–152.
[7] D.W. Matula, An algorithm for subtree identification (abstract) SIAM Rev. 10 (1968) 273–274.
[8] H.J. Ryser, Combinatorial Mathematics, Carus Math Monograph Fourteen (MAA, Rahway, 1963).