Information Processing Letters 32 (1989) 271–273
North-Holland

22 September 1989

ON THE SUBTREE ISOMORPHISM PROBLEM FOR ORDERED TREES

Erkki MÄKINEN

University of Tampere, Department of Computer Science, P.O. Box 607, SF-33101 Tampere, Finland

Communicated by M. Paul
Received 16 May 1988
Revised 27 September 1988

Keywords: Ordered tree, tree isomorphism, Zaks sequence, string pattern matching

1. Introduction

This note deals with the *subtree isomorphism problem*: Given two rooted trees T_1 and T_2, decide whether T_1 is isomorphic to any subtree of T_2. A tree is said to be *ordered* if the relative order of its subtrees in each node is fixed. Otherwise, a tree is *unordered*. In [4] the subtree isomorphism problem is studied in the case of unordered trees. It is shown there that if T_1 has m nodes and T_2 has n nodes then there is a $O(mn^{1.5})$ time algorithm for the subtree isomorphism problem. We show that a $O(m+n)$ time algorithm can be obtained when restricting ourselves to ordered trees only.

Our algorithm is based on tree encoding and on string pattern matching. Tree encodings are used for solving the tree isomorphism problem by Jovanović and Danilović [2]. They deal with unrooted and unordered trees and are unable to use string pattern matching.

The subtree isomorphism problem also has connections with the tree pattern matching problem (see, e.g., [1] and the references given there). This problem has the additional features that the nodes are labeled and the trees may contain special symbols which stand for arbitrary trees. There are several algorithms which reduce the tree pattern matching problem to string matching but algorithms with a time complexity less than $O(mn)$ are obtained only in some special cases [1].

2. Zaks sequences

We start by introducing an encoding method for (ordered) binary trees presented by Zaks [5]. This method has the advantage that it uses only $2n+1$ bits when representing a tree with n internal nodes. Label all the internal nodes by 1 and all the leaves by 0. The codeword, called the *Zaks sequence*, is obtained by reading the labels in preorder. (Visit first the root, then recursively traverse the left subtree in preorder, and then the right subtree in preorder.) We have the following characterization for feasible Zaks sequences.

2.1. Lemma. *A bit string is a Zaks sequence if and only if the following three conditions hold:*

(1) *the string begins with* 1,
(2) *the number of* 0*'s is one greater than the number of* 1*'s,*
(3) *no proper prefix of the string has property* (2).

Proof. See [5] □

The codeword of a subtree does not depend on its position. Thus, the use of Zaks sequences reduces the subtree isomorphism problem of ordered binary trees to a string matching problem. If the problem is to check whether T_1 is isomorphic with any of the subtrees of T_2, we can equivalently

check whether there is an occurrence of the codeword of T_1 in the codeword of T_2.

We know [3] that searching a pattern of length p from a string of length s takes time $O(p+s)$. Thus we obtain the following theorem (where the number of nodes in a tree T is denoted by $|T|$).

2.2. Theorem. *Let T_1 and T_2 be ordered binary trees. There is an algorithm which checks whether T_1 is isomorphic with any of the subtrees of T_2 in time $O(|T_1|+|T_2|)$.*

Proof. The Zaks sequence of an ordered binary tree can be produced in linear time. The theorem follows from the above discussion by Lemma 2.1. □

Note that the trivial algorithm which tries to match the root of T_1 to each node of T_2 takes time $O(|T_1| \cdot |T_2|)$.

3. Multiway trees

We use the phrase *multiway tree* for ordered k-ary trees with an arbitrary degree k. From a given multiway tree we can construct the corresponding *child–sibling binary tree representation* as follows. Let x be a node in the multiway tree with children (from left to right) $x_1, x_2, \ldots, x_n$. In the corresponding binary representation the node containing x has only two pointers. The left pointer accesses the first child x_1, and the right pointer accesses x's first sibling, if any. Similarly, the left pointer of x_1 accesses its first child, and the right pointer of x_1 accesses x_2. The child–sibling binary tree representation of the multiway tree of Fig. 1 is shown in Fig. 2. The binary representation of a given multiway tree can be produced in linear time. The binary representation of a multiway tree T is denoted by $b(T)$.

It is clear that this transformation preserves the identity of tree shapes. However, if a multiway tree T_1 is isomorphic to a subtree of a multiway tree T_2, then $b(T_1)$ is isomorphic to an entire subtree of $b(T_2)$ only in the case where the isomorphic subtree is the rightmost sibling in some level. Hence, the reduction to the string matching problem is not as straightforward as in the case of binary trees.

A subtree of a given multiway tree is represented in the corresponding binary tree as a node plus its left subtree (and by a single node if the subtree consists of one node only). On the other hand, the tree $b(T_1)$ to be matched has always an empty right subtree. In the matching problem concerning the Zaks sequences of $b(T_1)$ and $b(T_2)$ the pattern is not a feasible Zaks sequence but it

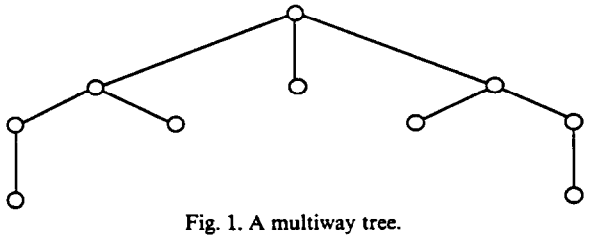

Fig. 1. A multiway tree.

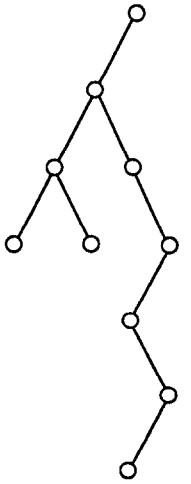

Fig. 2. The child–sibling binary tree representation of the multiway tree in Fig. 1.

has the form $1x$, where x is the Zaks sequence corresponding to the left subtree of $b(T_1)$. Hence, we ignore the empty right subtree of the root of $b(T_1)$ and the right subtree of the node at which the possible isomorphic subtree is rooted in T_2. Consequently, we have the following theorem.

3.1. Theorem. *Let T_1 and T_2 be ordered trees. There is an algorithm which checks whether T_1 is isomorphic with any of the subtrees of T_2 in time* $O(|T_1| + |T_2|)$.

References

[1] C.M. Hoffmann and M.J. O'Donnell, Pattern matching in trees, *J. ACM* **29** (1982) 68–95.

[2] A. Jovanović and D. Danilović, A new algorithm for solving the tree isomorphism problem, *Computing* **32** (1984) 187–198.

[3] D.E. Knuth, J.H. Morris and V.R. Pratt, Fast pattern matching in strings, *SIAM J. Comput.* **6** (1977) 323–350.

[4] S.W. Reyer, An analysis of a good algorithm for the subtree problem, *SIAM J. Comput.* **6** (1977) 730–732.

[5] S. Zaks, Lexicographic generation of ordered trees, *Theoret. Comput. Sci.* **10** (1980) 63–82.