A VLSI Algorithm for Calculating the Tree to Tree Distance

Xu Meirui and Liu Xiaolin
Beijing Computer Institute, Beijing 100044
Received June 15, 1991; revised August 29, 1991.

Abstract

Given two ordered, labeled trees and to find the distance from tree to tree is an important problem in many fields, for example, the pattern recognition field. In this paper, a VLSI algorithm for calculating the tree-to-tree distance is presented. The computation structure of the algorithm is a 2-D Mesh with the size and the time is , where are the numbers of nodes of the tree and tree respectively.

Keywords: VLSI algorithm, tree-to-tree distance, mesh, pattern recognition.

1. Introduction

Concerning the tree-to-tree distance, there are many applications, especially in the field of pattern recognition, since there are many objects which can be represented by trees. For example, any English letter can be represented by a labeled, ordered tree. With the help of the tree-to-tree distance, English letters can be recognized by a computer system.

In 1979, Tai presented a sequential algorithm for computing the distance between two trees with the time complexity , where are the numbers of nodes in the trees and respectively, and and are the maximum depth of the trees and respectively. There is also a sequential algorithm for this problem in [2], the time of which is , where is the maximum of and and is for the tree . In 1989, Zhang and Shasha improved the time and the space of Tai's algorithm, and proposed a sequential algorithm. In the same paper they also proposed a parallel algorithm for tree-to-tree distance. The computation model of the parallel algorithm is CRCW PRAM. The time of the parallel algorithm is and the number of processors is where leaves is O(m).

In this paper, a VLSI parallel algorithm for calculating the tree-to-tree distance is presented, and the computation structure of the algorithm is a 2-dimensional mesh with the size and the time is . Obviously, our parallel algorithm has the same time complexity as that in [3]. But there are two differences. One is that CRCW PRAM is only an abstract computation model, while in our parallel algorithm we have given a practical computation structure, i.e. a VLSI structure "2-D Mesh". The other is that the number of processors in our algorithm is less than that in [3], since leaves (β) = O(m).

2. Definitions and Notations

Before presenting our algorithm, some definitions and notations are needed.

In this paper, we shall discuss ordered, labeled trees, i.e. for each node , there is a label and the left to right order among siblings is significant. Each node will have a name. The name of the root is always 0. For a node, if its father's name is and it is the k-th son of its father (from left to right), then its name is . Each node a in a tree β will be given an index denoted by which is the index of the node a in the postorder traversal of the tree β. It is very often to use the notation in this paper, which is the index of the
Algorithm (tree-to-tree distance)

Input : tree β and tree α
Output : d(β, α), the distance from β to α.

Method :

(1) \(D(0, 0) = 0;\)
(2) for \(j = 1\) to \(n\) do
\[D(0, j) = N(\alpha/a)q;\] /* \(a = h^{-1}_{\alpha}(j)\) */
(3) for \(i = 1\) to \(m\) do
\[D(i, 0) = N(\beta/b)p;\] /* \(b = h^{-1}_{\beta}(i)\) */
(4) for \(j = 1\) to \(n\) do

\[D(i, j) = \min \left\{ \begin{array}{l}
E(k-1, i) + N(\beta/b)q \quad /* b = h^{-1}_{\beta}(i), i_k = h^{-1}_{\beta}(b, k) */
E(k-1, i-1) + D(i, j) \quad /* a = h^{-1}_{\alpha}(j), j_i = h^{-1}_{a}(a, l) */
\end{array} \right.\]

(5) \(d(\beta, \alpha) = D(m, n)\)

Fig. 1. A sequential algorithm for tree-to-tree distance.

For a tree \(\beta\), there are three kinds of editing operations: delete, insert and substitute.

- delete operation: \(b \rightarrow \land (b \neq \land), p\). (the node \(b\) is deleted from \(\beta\) with cost \(p\).)
- insert operation: \(\land \rightarrow a (a \neq \land), q\). (the node \(a\) is inserted into \(\beta\) with cost \(q\).)
- substitute operation: \(b \rightarrow a (b \neq \land \land a \neq \land), r\). (the node \(b\) is replaced by the node \(a\) with cost \(r\).)

**Definition.** Suppose \(S\) is a sequences \(s_1, s_2, \ldots, s_n\) of editing operations. \(S\) is called a transformation from a tree \(\beta\) to a tree \(\alpha\), if there exists a sequence \(T_0, T_1, \ldots, T_n\) of trees such that \(T_0 = \beta, T_n = \alpha\) and \(T_i\) is resulted from \(T_{i-1}\) by the editing operation \(s_i\) for all \(i\)'s, \(1 \leq i \leq n\). The cost of the transformation is

\[\text{cost}(S) = \sum_{i=1}^{n} \text{cost}(s_i).\]

**Definition.** The distance from a tree \(\beta\) to a tree \(\alpha\) is defined as follows: