CHAPTER 7

CONCLUSION

This dissertation presents a look at two parallel algorithm design strategies – linked list ranking and parentheses matching (PPM) – and their application to various problems. The study of linked list ranking demonstrates that it is a well-established strategy, used by a variety of algorithms, particularly those related to tree and graph problems. It also serves to demonstrate how strategies become established, leading to the proposal to establish parentheses matching as a general algorithm design strategy. In support of this proposal, algorithms and approaches which apply parentheses matching to a variety of tree problems have been developed.

This dissertation provides two new linked list ranking algorithms for the EREW PRAM model. The deterministic algorithm performs in $O(n \log(n/p))$ time utilizing $p$ processors, where $n \geq p \log p$. Although not work-optimal, the algorithm is simple when compared to previously proposed work-optimal algorithms and has the added advantage of having small constant factors in terms of space and time requirements. The randomized algorithm follows the same approach, but the use of randomization in one step decreases the time complexity. With high probability, it requires $O(n/p + \log p)$ time and, thus, achieves work-optimal $O(\log n)$ performance when $p = O(\frac{n}{\log n})$ processors. Furthermore, it uses less space than the deterministic algorithm.

As part of the effort to establish parallel parentheses matching as a viable strategy for algorithm design, the relationships between parentheses matching and several other known techniques have been established. The direct relationship between parentheses matching and the Euler tour technique has been developed. In addi-
tion, the relationship between tree contraction and parentheses matching has been demonstrated through the development of an algorithm to accomplish tree contraction through parentheses matching [40].

In addition, several algorithms have been developed which apply parentheses matching to tree related computations. These include solutions to the problems of computing the heights of all nodes in a tree, computing the extreme (maximum or minimum) values in the subtrees of a tree, and determining the lowest common ancestor (LCA) of two nodes of a tree. The height of the nodes in a tree and the extreme values in the subtrees are found in $O(\log n)$ time using $O\left(\frac{n}{\log n}\right)$ processors on the CREW PRAM model. Determining the LCA for a single pair of nodes in a tree is computed in $O(\log n)$ time using $O\left(\frac{n}{\log n}\right)$ processors on the EREW PRAM model. A variation of the algorithm computes a table of the least common ancestors of all node-pairs in $O(\log n)$ time using $O\left(\frac{n^2}{\log n}\right)$. Retrieval from the table requires the CREW model for concurrent access. An algorithm for computing the nearest enclosing parentheses for an arbitrary pair of parentheses is also presented as one of the tools necessary for successful application of parentheses matching. It attains $O(\log n)$ time utilizing $O\left(\frac{n}{\log n}\right)$ processors on the EREW PRAM model.

As a practical application of the PPM strategy, two algorithms have been developed for globally balancing binary trees. Each accepts as input the unbalanced binary tree and creates either the perfect or complete binary tree that maintains the inorder numbering. The balanced tree is created in work-optimal $O(\log n)$ time utilizing $O\left(\frac{n}{\log n}\right)$ processors on the EREW PRAM model.

In order for the parentheses matching strategy to be extended to other models of computation, in particular the well-known hypercube model, a PPM algorithm has been developed for the hypercube. The development of a load balancing strategy for the problem allows the simple divide-and-conquer technique to run in $O(\log^2 p + \frac{n}{p})$
using \( p \) processors. Although, not work-optimal, it compares favorably with two other existing algorithms in terms of simplicity and flexibility with respect to the number of processors used. A hypercube implementation for the nearest enclosing parentheses problem has also been developed. It achieves \( O(\log n) \) time when using \( O(\frac{n}{\log n}) \) processors.

This work has shown that parallel parentheses matching is a viable technique for the solutions of numerous tree related problems. The recognition of the relationship between parentheses matching to Euler tour techniques and to tree contraction [40] opens the door to the application of this technique to many other problems to which these techniques have been previously applied. In particular, the application of PPM to special classes of graphs, such as interval graphs, cographs, permutation graphs, etc., appears very promising. Another possibility is the application of PPM to problems which are solved sequentially using stacks and queues, as this is also characteristic of several of the problems for which parentheses matching solutions have been provided. The relationship of PPM to these and other data structures is a possible area for future research.

It is believed that further study will provide solutions to other classes of problems as well and will allow for a more concrete characterization of the problems for which parallel parentheses matching is a viable strategy. In addition, it is likely that some of the algorithms presented here can be improved with respect to the time complexity or with transition to a weaker model.

Finally, the implementation of PPM (and NEPA) on the hypercube allows for the application of this technique to many of the same problems for which PRAM solutions have been provided.