Optimal Tree Contraction on the Hypercube and Related Networks

Ernst W. Mayr¹ and Ralph Werchner²

¹ Institut für Informatik, Technische Universität München, mayr@informatik.tu-muenchen.de
² Fachbereich Informatik, J.W. Goethe-Universität, Frankfurt am Main, werchner@informatik.uni-frankfurt.de

Abstract. An optimal tree contraction algorithm for the boolean hypercube and the constant degree hypercubic networks, such as the shuffle exchange or the butterfly network, is presented. The algorithm is based on novel routing techniques and, for certain small subtrees, simulates optimal PRAM algorithms. For trees of size \( n \), stored on a \( p \) processor hypercube in order, the running time of the algorithm is \( O\left(\left\lfloor \frac{n}{p} \right\rfloor \log p \right) \). The resulting speed-up of \( O(p/\log p) \) is optimal due to logarithmic communication overhead, as shown by a corresponding lower bound.

1 Introduction

Tree contraction is a fundamental technique for solving problems on trees. A given tree is reduced to a single node by repeatedly contracting edges, resp., merging adjacent nodes. This operation can be used for problems like top-down or bottom-up algebraic tree computations [ADKP89], which themselves can be applied to solve the membership problem for certain subclasses of languages in DCFL [GR86] or to evaluate expressions consisting of rational operands and the operators \(+, -, \cdot, /\).

In the context of parallel processing the objective is to contract the tree using a small number of stages, each consisting of independent edge contractions. Brent [B74] was the first to show that a logarithmic number of such stages is sufficient, and he applied this result to the restructuring of algebraic expression trees, producing logarithmic depth. Subsequent work [MR85] [GR86] [GMT88] [KD88] [ADKP89] concentrated on the efficient parallel computation of contraction sequences, and eventually resulted in work optimal logarithmic time tree contraction algorithms on the EREW PRAM.

We consider the tree contraction problem on the binary hypercube and on similar networks. In this model, computing a suitable contraction sequence is more difficult, and the additional problem of routing pairs of nodes to be contracted to common processors arises. For the contraction of paths, both problems become trivial and can be solved in logarithmic time by a parallel prefix operation [S80]. For arbitrary trees, we combine two known (PRAM) contraction techniques with a recursive approach. To achieve a logarithmic running time we separate the local communication operations from the few long distance communication steps and perform them in appropriately sized subcubes. The necessary routing steps are performed by new logarithmic time algorithms designed for special classes of routings. Our algorithm
contracts a tree of size $n$ on a $p$ processor binary hypercube in $O\left(\frac{n}{p} \log p\right)$ steps, which we also show is asymptotically optimal by a matching lower bound.

2 Fundamental Concepts and Notation

We first give a short description of our model of computation. A network of processors is a set of processors, interconnected by bidirectional communication links. Each processor has the capabilities of a RAM, a unique processor-id and additional instructions to send or receive one machine word to respectively from a direct neighbor. We assume the word length of the processors to be $\Theta(\log p)$ bits where $p$ is the number of processors. The topological structure of the communication links can be described by an undirected graph $(V, E)$.

A $d$-dimensional hypercube is a network of processors represented by the graph $G = (V, E)$ with

$$V = \{0, 1\}^d,$$
$$E = \{(u, v) \mid u \text{ and } v \text{ differ in exactly one bit}\}.$$

A network with bounded degree and a structure very similar to the hypercube is the $d$-dimensional shuffle-exchange. The processors are numbered as they are in the hypercube. Two processors are connected by a link if their ids differ only in the last bit (exchange edges) or if one id is a cyclic shift by one position of the other id (shuffle edges). A compendium of results concerning hypercubes and shuffle-exchange networks can be found in [L92].

A tree contraction is the reduction of a given binary tree to a single node, proceeding in stages. In a stage, a set of disjoint pairs of adjacent nodes in the current tree is selected, and the edges connecting these pairs are contracted, i.e., each pair of nodes is merged into a single node. The contractions are not allowed to produce a node with more than two children.

For the tree contraction problem on networks of processors we assume that the data comprising each node can be stored in a constant number of processor words and that two adjacent nodes stored by the same processor can be merged in constant time.

On networks the complexity of the tree contraction problem depends on the representation of the given tree. We assume the in-order sequence of the nodes to be evenly distributed over the sequence of processors ordered by processor-id. To uniquely describe the structure of the tree in this way, each nontrivial subtree is assumed to be enclosed by a pair of parentheses. To contract a tree given in pre- or post-order, it can be transformed to in-order without asymptotically increasing the time complexity using the routing algorithm of [MW92].

If the tree is given by an arbitrary, but balanced distribution of the nodes linked by pointers, a list ranking on the Euler tour of the tree and a corresponding routing operation can be used to transform the tree into in-order representation. On a $p$ processor hypercube, this transformation can be carried out in $O\left(\frac{n}{p} \log^2 p \log \log \log p \log^* p\right)$ steps for trees of size $n$. The dominating part is the time required for the list-ranking.