A Distributed Model and Algorithm For Binary Search

Mike Girou Theodora L. Girou
MT Systems Company MT Systems Company
2538 Big Horn 2538 Big Horn
Richardson TX 75080 Richardson TX 75080

I. Hal Sudborough Ioannis G. Tollis
Computer Science Computer Science
Univ. Texas at Dallas Univ. Texas at Dallas
Richardson TX 75083 Richardson TX 75083

Synopsis

As part of the design of a roamer validation system for the cellular telecommunications industry, we consider parallel binary search techniques. Binary search on a distributed memory system, such as the hypercube or mesh, is no faster than it is on a single processor. The CRCW PRAM model offers significant speedup, but is not practical. As an alternative, we construct a hybrid parallel processing model by adding several dozen bytes of CRCW memory to a distributed memory system. A modified binary search algorithm for this model yields performance equal to that obtainable on a CRCW PRAM.

Our environment consists of a static sorted search list of $2^n$ elements, and a distributed memory system with $2^p$ computers, where we assume for simplicity that $p$ divides $n$. We use a modified divide and conquer approach, where the search list is distributed such that, for any current search interval, the left endpoint of each of the $2^p$ sub-intervals is contained in a separate computer. These endpoints are compared against the search key, causing the incremental generation of the subscript corresponding to the search key. We build this subscript from the left, one base $2^p$ digit per cycle.

Distributed Model. Add enough CRCW memory to any distributed memory parallel processing system to contain a search key, an address vector $V$, a stop bit $S$, an error bit $E$, a working variable $W$, and an index variable $I$. $V$ contains $n/p$ digits, each a base $2^p$ number. $V$ will contain the subscript, built from the left, one digit at a time, corresponding to our search key. We build this subscript from the left, one base $2^p$ digit per cycle.

Basic Cycle For Processor $k$. Turn $E$ on. Compare the search key against the element whose subscript is given by a local variable which is equal to the CRCW variable $V$, except that $V_I$ is changed locally such that $|W + V_I|_{2^p} = k$. If the search key is higher, turn $E$ off, and attempt to update $V_I$, subject to the contention protocol. If the search key is lower, do nothing. If the search key is equal, turn $S$ on and $E$ off.

It can be shown that the algorithm is correct, and either computes the subscript corresponding to the search key or sets the error bit. Since the subscript contains $n/p$ digits, and one digit is calculated during each time increment, we conclude that the time performance is $O(n/p)$, the best obtainable with the CRCW PRAM model. However, we achieve this performance on a slightly modified distributed memory system.

We continue to investigate several related questions. Are there parallel data structures and algorithms, similar to those associated with balanced binary trees, that allow $O(n/p)$ searching as the search list changes dynamically? Can other parallel algorithms achieve CRCW PRAM performance on distributed memory models by adding a small amount of CRCW memory? Or is the binary search problem an anomaly?