CSE-40533

Introduction to Parallel Processing

Chapter 2

A Taste of Parallel Algorithms

• Consider five basic building-block parallel operations
• Implement them on four simple parallel architectures
• Learn about the nature of parallel computations and interplay between algorithm and architecture
2.1 Some Simple Computations

Semigroup computation on a uniprocessor.

Semigroup computation viewed as a tree or fan-in computation.
Prefix computation on a uniprocessor.

3. Packet routing
   one processor sending a packet of data to another

4. Broadcasting
   one processor sending a packet of data to all others

5. Sorting
   processors cooperating in rearranging their data into desired order
2.2 Some Simple Architectures

Diameter of linear array: $D = p - 1$
(Max) Node degree: $d = 2$

Diameter of balanced binary tree: $D = 2\lceil \log_2 p \rceil$; or one less
(Max) Node degree: $d = 3$

We almost always deal with complete binary trees:

$p$ one less than a power of 2 \quad D = 2 \log_2 (p + 1) - 2
Fig. 2.4. A 2D mesh of nine processors and its torus variant.

Diameter of \( r \times (p/r) \) mesh: \( D = r + p/r - 2 \)
(Max) Node degree: \( d = 4 \)
Square meshes preferred; they minimize \( D \ (= 2\sqrt{p} - 2) \)

Fig. 2.5. A shared-variable architecture modeled as a complete graph.

Diameter of complete graph: \( D = 1 \)
(Max) Node degree: \( d = p - 1 \)
2.3 Algorithms for a Linear Array

Fig. 2.6. Maximum-finding on a linear array of nine processors.

Fig. 2.7. Computing prefix sums on a linear array of nine processors.

Diminished prefix computation: the ith result excludes the ith element (e.g., sum of the first i – 1 elements)
Packet routing or broadcasting:
right- and left-moving packets have no conflict
Fig. 2.9. Sorting on a linear array with the keys input sequentially from the left.
In odd steps, 1, 3, 5, etc., odd-numbered processors exchange values with their right neighbors.

For odd-even transposition sort:

Speed-up $= O(p \log p) / p = O(\log p)$

Efficiency $= O((\log p) / p)$

Redundancy $= O(p / (\log p))$

Utilization $= 1/2$

Fig. 2.10. Odd-even transposition sort on a linear array.
2.4 Algorithms for a Binary Tree

Fig. 2.11. Parallel prefix computation on a binary tree of processors.
Some applications of the parallel prefix computation

Finding the rank of each 1 in a list of 0s and 1s:

Data : 0 0 1 0 1 0 0 1 1 1 0
Prefix sums : 0 0 1 1 2 2 2 3 4 5 5
Ranks of 1s : 1 2 3 4 5

Priority circuit:

Data : 0 0 1 0 1 0 0 1 1 1 0
Diminished prefix ORs : 0 0 0 1 1 1 1 1 1 1 1
Complement : 1 1 1 0 0 0 0 0 0 0 0
AND with data : 0 0 1 0 0 0 0 0 0 0 0

Carry computation in fast adders

Let “g”, “p”, and “a” denote the event that a particular digit position in the adder generates, propagates, or annihilates a carry. The input data for the carry circuit consists of a vector of three-valued elements such as:

\[ p \, g \, a \, g \, g \, p \, p \, p \, g \, a \, c_{in} \]

\[ g \text{ or } a \]

direction of indexing

Parallel prefix computation using the carry operator “¢”

\[ p \, c \, x = x \quad \text{x propagates over p, for all } x \in \{g, p, a\} \]

\[ a \, c \, x = a \quad \text{x is annihilated or absorbed by a} \]

\[ g \, c \, x = g \quad \text{x is immaterial; a carry is generated} \]
Packet routing on a tree

\[
\begin{align*}
\text{maxl} (\text{maxr}) &= \text{largest node number in the left (right) subtree} \\
\text{if} & \quad \text{dest} = \text{self} \\
\text{then} & \quad \text{remove the packet \{done\}} \\
\text{else if} & \quad \text{dest} < \text{self} \quad \text{or} \quad \text{dest} > \text{maxr} \\
& \quad \text{then route upward} \\
\text{else if} & \quad \text{dest} \leq \text{maxl} \\
& \quad \text{then route leftward} \\
& \quad \text{else route rightward} \\
\text{endif} \\
\text{endif} \\
\end{align*}
\]
Other indexing schemes might lead to simpler routing algorithms

Broadcasting is done via the root node
Sorting: let the root "see" all data in nondescending order

Fig. 2.12. The first few steps of the sorting algorithm on a binary tree.

Fig. 2.13. The bisection width of a binary tree architecture.
2.5 Algorithms for a 2D Mesh

Finding the max value on a 2D mesh

Computing prefix sums on a 2D mesh

Row-major order required if the operator is not commutative
Routing and broadcasting done via row/column operations

Fig. 2.14. The shearsort algorithm on a 3 × 3 mesh.
2.6 Algorithms with Shared Variables

Semigroup computation: each processor read all values in turn and combine

Parallel prefix: processor $i$ read/combine values $0$ to $i - 1$

Both of the above are quite inefficient, given the high cost

Packet routing and broadcasting: one step, assuming all-port communication

Sorting: rank each element by comparing it to all others, then permute according to ranks

Figure for Problem 2.13.