
Part 3: Parallelism in Algorithms

Edgar Solomonik

University of Illinois at Urbana-Champaign
Circuits and PRAM

- Circuits were the first parallel algorithms

- The PRAM model tries to stay consistent with this view
Inner Product in the PRAM Model

- Inner product with $n$ processors

- Inner product with $n/\log_2(n)$ processors
Basic Linear Algebra Subroutines (BLAS) in the PRAM Model

- Vector scaling (BLAS 1)

- Matrix-vector multiplication and outer product (BLAS 2)
Work-Depth Model

- The work-depth (or work-time) model keeps track only of total work and algorithm depth/time

- It's possible to schedule a work-optimal PRAM algorithm so that it uses an asymptotically optimal number of processors
Numerical Linear Algebra in PRAM

- Standard algorithms for triangular solve and matrix factorizations have polynomial depth

- Polylogarithmic depth algorithms exist for solving linear systems
Recursive Matrix Factorization Depth

- Recursive Cholesky $A = LL^T$ has polynomial depth

- Recursive triangular inversion $S = L^{-1}$ has logarithmic depth
Parallel sorting within a single shared-memory

Most sorting algorithms can be classified as *merge-based* or *distribution-based*
Bitonic Sort
Bitonic Merge
Bitonic sequence as a circle
Matching opposite pairs in the circle
Swapping opposite pairs in the circle
Collecting the min/max into different subsequences
Any partition subdivides smaller/greater halves
Arranging the two halves into new circles
Swapping opposites again
Continuing with bitonic merge recursively
Bitonic merge

- A *bitonic sequence* is any cyclic shift of the sequence
  \[ \{i_0 \leq \cdots \leq i_k \geq \cdots \geq i_{n-1}\} \]

- There exists \( l \leq k \), such that the largest \( n/2 \) elements of (unshifted bitonic sequence) \( S \) are the subsequence \( \{i_l, \ldots, i_{l+n/2-1}\} \)
BFS with Sparse Linear Algebra

- For undirected graph $G = (V, E)$ Breadth First Search (BFS) takes as input a source vertex $s$ and outputs an assignment of vertices to frontiers.

- With adjacency matrix $A$ of $G$, can compute BFS via matrix-vector products.
Sparse Linear Algebra in PRAM

- Sparse-matrix-vector product (SpMV) with $m$ nonzeros (edges) in matrix

- Sparse-matrix-sparse-vector product (SpMSpV) with $k$ nonzeros (frontier vertices) in vector
BFS on a PRAM

- Each BFS iteration requires an SpMSpV with an output filter:
  \[ f^{(i+1)} = u^{(i)} \odot (Af^{(i)}) \]

- Different choices of BFS algorithm yield different work/depth
BFS on a PRAM

- Each BFS iteration requires an SpMSpV with an output filter
  \[ f^{(i+1)} = u^{(i)} \odot (Af^{(i)}) \]

- Different choices of BFS algorithm yield different work/depth
Connectivity in Graphs

- Connectivity seeks to label vertices with a unique label for each connected component

- Shiloach and Vishkin (1980) CRCW PRAM algorithm for connectivity
Shiloach-Vishkin Connectivity Algorithm

Let each node \( i \) store ‘parent’ \( p(i) \) and perform below steps until convergence

- conditional star hooking

- unconditional star hooking

- Shortcutting (pointer chasing)
A graph with two connected components
First iteration

1. conditional star hooking
First iteration

2. unconditional star hooking

star

star
First iteration

3. shortcutting
Second iteration

1. conditional star hooking
Second iteration

2. unconditional star hooking
Analysis of parallel tree connectivity

Algorithm converges after $O(\log(n))$ iterations

- Sum of tree heights (starts at $n$) decreases by a factor of at least $\frac{3}{2}$ every iteration

- Requires $O(n + m)$ work per iteration
Shortest Paths

- Given a positive weight function $w : E \rightarrow \mathbb{R}^+$, compute shortest distances from a source vertex $s$ to all other vertices.

- Bellman-Ford can be expressed as matix-vector products on the tropical (min–plus) semiring, using SpMV/SpMSpV.
All-Pairs Shortest-Paths

- Given a positive weight function \( w : E \rightarrow \mathbb{R}^+ \), compute shortest distances matrix \( D \) containing minimum distances between all pairs of vertices.

- Floyd-Warshall algorithm computes achieves \( O(n^3) \) work.
Floyd Warshall Algorithm

- \( D^{(i)} \) contains the distances of all shortest paths \( S^{(i)} \) with at most \( i \) edges going through some subset of vertices \( \{1, \ldots, i - 1\} \)

- A recursive alternative to Floyd-Warshall is given by Gauss-Jordan elimination (Kleene’s APSP algorithm)
Parallel All-Pairs Shortest-Paths

- Path doubling can be used to obtain polylogarithmic depth

- Tiskin (2001) proposed an improvement to achieve $O(n^3)$ cost
Parallel (Approximate) Matrix Inversion

- Gauss-Jordan can be used to invert matrix, recursive Cholesky is similar

- Can theoretically invert with polylogarithmic depth via Newton’s method