Some Scalable Parallel Algorithms for Geometric Problems

L. Boxer, R. Miller, and A. Rau-Chaplin

miller@cs.buffalo.edu
Outline

1. Motivation
2. Models of Computation
3. Previous Results
4. Fundamental Operations
5. Sample Algorithm
6. Additional Results
7. Summary
Motivation

- Wealth of algorithms: fine-grained models.
- Commercial machines: coarse-grained.
- Fine-grained algorithms do not port well.
- Fine-grained algorithms often tied to interconnection network.
- Consider
  - scalable algorithms
  - interconnection-independent environment
  - geometric problems
Portable Models

- **BSP [Valiant90]**: supersteps consist of
  a) local computation,
  b) global communication, and then
  c) barrier synchronization.
  Input and output pool for each PE.

- **LogP [Culler93]**: PE is either in operational or stalling mode at each step. Operational: either a) local computation, b) receive message, or c) submit a message.

- **$C^3$ [Hambrusch95]**: considers the complexity of computation, the pattern of communication, and the potential congestion that arises during communication.
Coarse Grained Multicomputer (CGM)

- $CGM(n, p)$ consists of $p$ processors, each with $\Omega(\frac{n}{p})$ local memory, where $\Omega(\frac{n}{p})$ is “considerably larger” than $\Theta(1)$.
- Arbitrary interconnection network.
- Examples: Cray T3D, IBM SP2, Intel Paragon, TMC CM-5
- For determining time complexities, consider both local computation time and interprocessor communication time.
Previous Results on CGM

- Area of union of rectangles [Dehne93]
- 3D-maxima [Dehne93]
- 2D-nearest neighbors of a point set [Dehne93]
- Lower envelope of non-intersecting line segments in plane [Dehne93]
- 2D-weighted dominance counting [Dehne93]
- Randomized 3D convex hull [Dehne95]
Fundamental Algorithms: Previous Results  
(Sort-based)

\[ T_{\text{sort}}(n, p): \text{the time required to sort } \Theta(n) \text{ data on a } CGM(n, p). \]

\( T_{\text{sort}}(n, p) \) time on a \( CGM(n, p) \) [Dehne]:

- **Segmented broadcast:** For indices \( 1 \leq j_1 < j_2 < \ldots < j_q \leq p \), each PE \( P_{j_i} \) broadcasts a list of \( \frac{n}{p} \) data items to PEs \( P_{j_i+1}, \ldots, P_{j_i+1} \).

- **Multinode broadcast:** Every PE sends the same \( \Theta(1) \) data to every other PE.

- **Total exchange:** Every PE sends \( \Theta(1) \) data (not necessarily the same) to every other PE.
Fundamental Algorithms: New Results
(Sort-based)

$T_{\text{sort}}(n, p)$ time on a $CGM(n, p)$:

- **Permutation exchange:** Given a permutation $\sigma$, every PE $P_i$ sends a list of $\frac{n}{p}$ data items to PE $P_{\sigma(i)}$.

- **Semigroup operation:** Let $X = \{x_n\}$ be distributed evenly among the PEs. Let $\circ$ be a unit-time, associative, binary operation on $X$. Compute $x_1 \circ x_2 \circ \ldots \circ x_n$. 
Fundamental Algorithms: New Results
(Sort-based)

$T_{\text{sort}}(n, p)$ time on a $C\!G\!M(n, p)$:

- **Parallel prefix:** Let $X = \{x_n\}$ be distributed evenly among the PEs. Let $\circ$ be a unit-time, associative, binary operation on $X$. Compute all $n$ members of $\{x_1, x_1 \circ x_2, \ldots, x_1 \circ x_2 \circ \ldots \circ x_n\}$.

- **Merge:** Let $X$ and $Y$ be lists of ordered data, each evenly distributed among the PEs, with $|X| + |Y| = \Theta(n)$. Combine these lists so that $X \cup Y$ is ordered and evenly distributed among the PEs.
Parallel search: Let $X = \{x_m\}$ and $Y = \{y_n\}$ be lists, each distributed evenly among the PEs. Each $x_i \in X$ searches $Y$ for a value.

**Time:** $T_{\text{sort}}(m + n, p)$ on a $CGM(m + n, p)$.

Formation of combinations: Let $X = \{x_n\}$ and let $k > 1$ be a fixed positive integer. Form the set of $\Theta(n^k)$ combinations of members of $X$ that have exactly $k$ members.

**Time:** $O(T_{\text{sort}}(n^k, p))$ on a $CGM(n^k, p)$. 
Fundamental Algorithms: New Results
(Sort-based)

- Formation of pairs from lists: Let $X = \{x_m\}$ and let $Y = \{y_n\}$. Form all pairs $(x_i, y_j)$, where $x_i \in X$, $y_j \in Y$.

Time: $T_{\text{sort}}(mn, p)$ on a $CGM(mn, p)$. 
All Rectangles Problem

**Defn.** A polygon \( P \) is *from* \( S \subset R^2 \) if all vertices of \( P \) belong to \( S \). The *AR problem* is to find all rectangles from \( S \).

**Proposition [V KD91]:** Let \( S \subset R^2 \), \( |S| = n \). Then a solution to the AR problem has \( \Theta(n^2 \log n) \) output in the worst case.
Scalable Algorithm for All Rectangles Problem

**Theorem:** Let $S = \{v_0, v_1, \ldots, v_{n-1}\}$ be given as input. Then the AR problem can be solved in $T_{\text{sort}}(n^2 \log n, p)$ time on a $CGM(n^2 \log n, p)$.

**Note:** A rectangle may be determined by a pair of opposite sides with nonnegative slope.
Algorithm: All Rectangles Problem

1. Form the set $L$ of all line segments with endpoints in $S$ and with nonnegative slopes.

2. Sort the members of $L$ so that if $\ell_0 < \ell_1 < \ell_2$ and $(\ell_0, \ell_2)$ is a pair of opposite sides of a rectangle, then $(\ell_0, \ell_1)$ and $(\ell_1, \ell_2)$ are pairs of opposite sides of rectangles.
Algorithm (cont’d)

3. Using parallel prefix:
   - For each $\ell \in L$, determine the first and last edge in its group.
   - For each $\ell \in L$, determine $First(\ell)$, the number of rectangles for which $\ell$ is the first edge.
   - For each $\ell \in L$, determine $Prec(\ell)$, the number of rectangles that precede it.
Algorithm (cont’d)

4. Using parallel search operations:

- Determine the first side of every rectangle based on the $Prec(\ell)$ and $First(\ell)$ values.
- Determine the second side of every rectangle based on the $Prec(\ell)$ and $First(\ell)$ values.
Lower Envelope

**Defn.:** Let $S$ be a set of polynomial functions $\{f_n\}$. The lower envelope of $S$ is the function

$$LE(x) = \min \{f_i(x) \mid i = 1, \ldots, n\}.$$  

**Theorem:** Let $k$ be a fixed positive integer and let $S$ be a set of polynomial functions, each of degree at most $k$. Assume that the members of $S$ are distributed evenly among the processors. Then the lower envelope of $S$ may be determined in slightly worse than linear time and space.
Envelope-Related Problems

**Theorem:** Let $S$ be a set of vertically convex polygons in $\mathbb{R}^2$ whose boundaries have a total of $n$ line segments. Then the Common Intersection Problem for $S$ can be solved in slightly worse than linear time and space.

**Theorem:** Let $S$ be a system of point-objects, each of which is in $k$–motion in $\mathbb{R}^d$. Then, as a function of $t$, a nearest member of $S \setminus \{s_0\}$ to $s_0$ may be described in slightly worse than linear time and space.

**Theorem:** Let $S = \{P_0, \ldots, P_{n-1}\}$ be a set of points in the plane with $k$-motion. Then the ordered intervals of time during which a given point $P_i$ is an extreme point of $\text{hull}(S)$ can be determined in slightly worse than linear time and space.
Maximal Collinear Sets

Defn.: Given a set $S$ of $n$ points in a Euclidean space, find all maximal equally-spaced collinear subsets of $S$ determined by segments of any length $\ell$. (The algorithm of [Kahng] runs in optimal $\Theta(n^2)$ serial time.)

Theorem: Let $d$ be a fixed positive integer. Let $S \subset \mathbb{R}^d$, $|S| = n$. Then the AMESCS Problem can be solved for $S$ in $T_{\text{sort}}(n^2, p)$ time on a $CGM(n^2, p)$. 
Point Set Pattern Matching

**Defn.:** Given a set $S$ of points in a Euclidean space $\mathbb{R}^d$ and a pattern $P \subseteq \mathbb{R}^d$, find all instances of subsets $P' \subseteq S$ such that $P$ and $P'$ are congruent.

**Theorem:** The Point Set Pattern Matching Problem in $\mathbb{R}^1$ can be solved on a $CGM(k(n-k), p)$ in optimal $T_{sort}(k(n-k), p)$ time.
Summary

1. Scalable algorithms on *Coarse Grained Multicomputer*

2. Fundamental Operations

3. Geometric Problems