Massively Parallel Computation

- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree

Philip Bille

Massively Parallel Computation

- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree

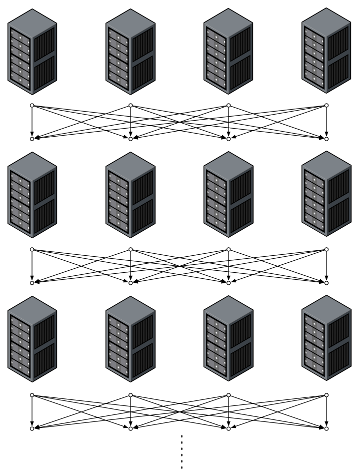
Computational Model

- Massively Parallel Computation (MPC) model.
  - P processors each with space S.
  - Typically $S = N^{\varepsilon}$ and $P = N^{1-\varepsilon}$.
  - Synchronous computation in rounds.
  - Round = local computation + communication.
  - Communication into a processor is < S.
- Complexity model.
  - Rounds and space ($\Longrightarrow$ communication)
  - Computation is free (!)
- Implementations.
  - Map Reduce
  - Bulk-Synchronous parallel

N = problem size

P = number of processors

S = space on each processor



Massively Parallel Computation

- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree

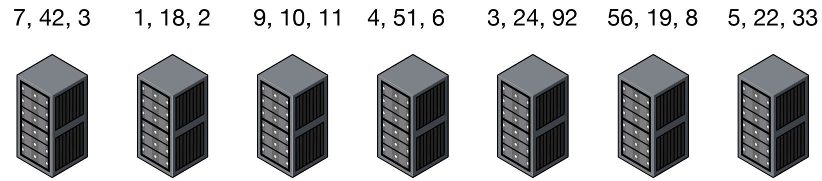
Summing

7, 42, 3,1, 18, 2, 9, 10, 11, 4, 51, 6, 3, 24, 92, 56, 19, 8, 5, 22, 33

426

- Sum. Given a list of N integers $A_0, A_1, \dots, A_{N-1}$ compute their sum.
- Input distributed arbitrarily among processors.

Summing



- Assume $S = \Theta\left(\sqrt{N}\right)$ and $P = \Theta\left(\sqrt{N}\right)$
- Sum.
  - Each processor computes local sum and sends to processor 0.
  - Compute global sum at processor 0.
- Rounds. 2.

Massively Parallel Computation

- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree

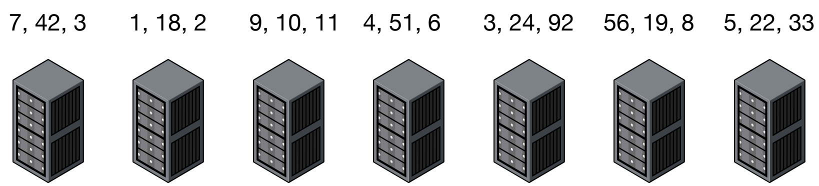
Sorting

7, 42, 3,1, 18, 2, 9, 10, 11, 4, 51, 6, 3, 24, 92, 56, 19, 8, 5, 22, 33

(7,8), (42,18), (3,3), (1,1), (18,13),(2,2), (9,10), (10, 11), (11,12),(4,5),(51,19), (6,7), (3,4), (24,16), (92,21), (56, 20), (19,14), (8,9), (5, 6), (22,15), (33, 17)

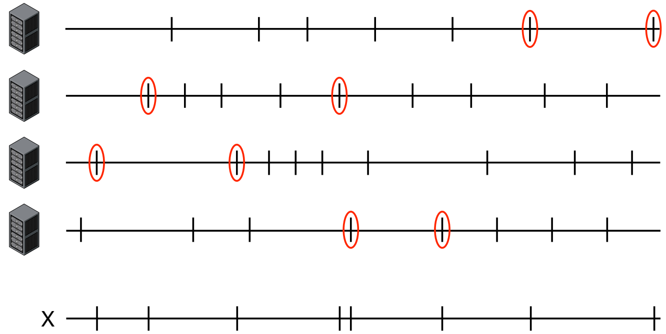
- Sorting. Given a list of N integers $A_0, A_1, \dots, A_{N-1}$ compute list $(A_0, \text{rank}(A_0)), (A_1, \text{rank}(A_1)), \dots, (A_{N-1}, \text{rank}(A_{N-1}))$
- Input and output distributed arbitrarily among processors.

Sorting



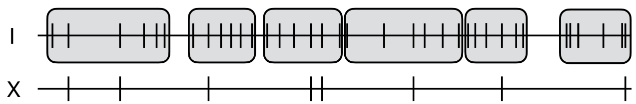
- Goal. Sorting in O(1) rounds whp. with $S = \tilde{\Theta}\left(\sqrt{N}\right)$ and $P = \tilde{\Theta}\left(\sqrt{N}\right)$.
- Idea.
  - Sample $\tilde{\Theta}(\sqrt{N})$ items and use sample to partition items into $\tilde{\Theta}(\sqrt{N})$ ranges.
  - Distribute items according to ranges and sort each range locally.

Sorting



- Sample.
  - Each processor samples its items with probability $2P \ln N / N$ and sends these to processor 0.
  - Processor 0 broadcasts the set of samples to all processors.
  - Let X be the set of samples. $|X| \leq 4P \ln N$ whp.

Sorting



- Lemma. Let I be the sorted input. Consider a partition of I into P ranges of N/P consecutive items. Then, all ranges contain at least one item from X whp.
- Proof.

Pr we don't sample item

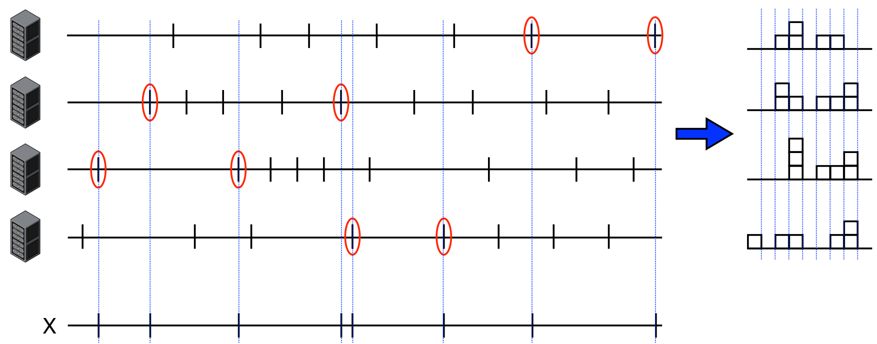
$$\Pr(\text{range contains no items}) = \left(1 - \frac{2P\ln N}{N}\right)^{\frac{N}{P}} \leq e^{-2\ln N} = \frac{1}{N^2}$$

Pr we don't sample item in range

$(1+x)^r \leq e^{rx}$

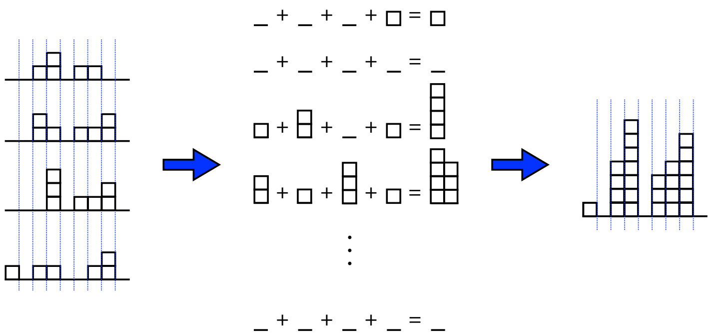
- $\implies \Pr(\text{some range contain no items}) = P \cdot \frac{1}{N^2} < \frac{1}{N}$
- $\implies \Pr(\text{all ranges contain at least 1 item}) > 1 - \frac{1}{N}$

Sorting



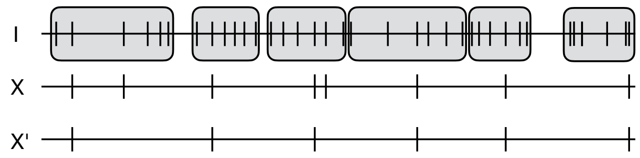
- Compute local histogram.
  - Each processor counts number of items in ranges defined by X.
  - Each histogram uses O(|X|) space.

Sorting



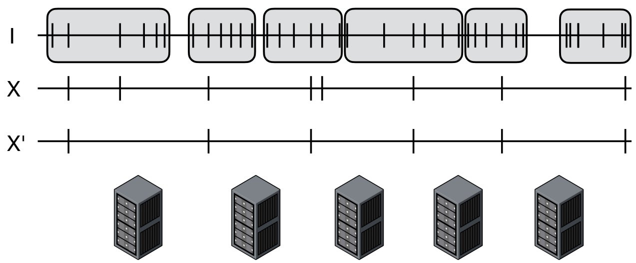
- Compute global histogram.
  - Each processor sends count for range i to processor i mod P.
  - Processor i sums counts for range i mod P and sends sum to processor 0.
  - Processor 0 constructs global count.
  - Each processor is responsible for counting $|X|/P = O(\log N)$ ranges and receives $O(P \log N)$ integers.

Sorting



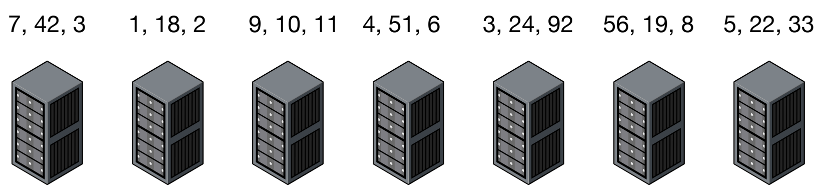
- Select.
  - Processor 0 selects $X' \subseteq X$ such that each range defined by X' contains O(N/P) items from I and |X'| = O(P).
  - Processor 0 broadcasts X' to all machines.
  - Sampling lemma $\Longrightarrow$ X' exists whp.

Sorting



- Exchange.
  - Assign each range defined by X' to a processor.
  - Each processor sends each of its items to processor assigned to corresponding range.
  - Each processor locally sorts its items.
  - Output sorted sequence.

Sorting

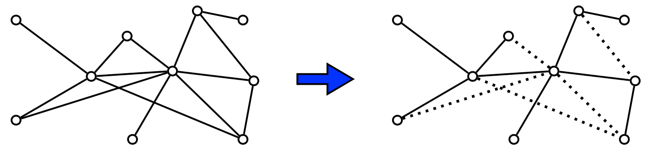


- Theorem. Sorting in O(1) rounds whp. with $S = \tilde{\Theta}\left(\sqrt{N}\right)$ and $P = \tilde{\Theta}\left(\sqrt{N}\right)$.

Massively Parallel Computation

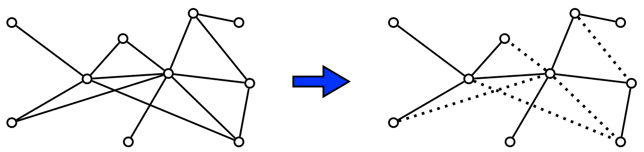
- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree

Minimum Spanning Tree



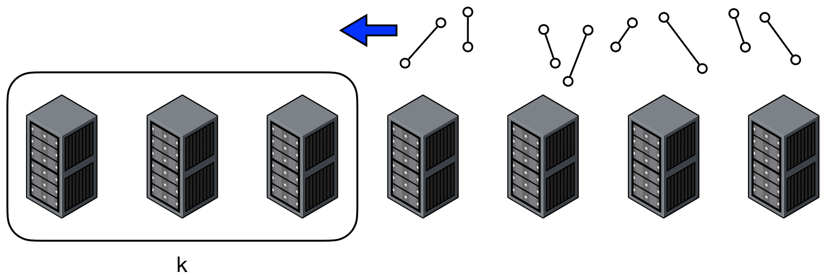
- Minimum spanning tree. Given a connected, weighted, undirected graph compute the minimum spanning tree (MST).
- Input given as list of edges with weights. Output edges in MST.
- Input and output distributed arbitrarily among processors.

Minimum Spanning Tree



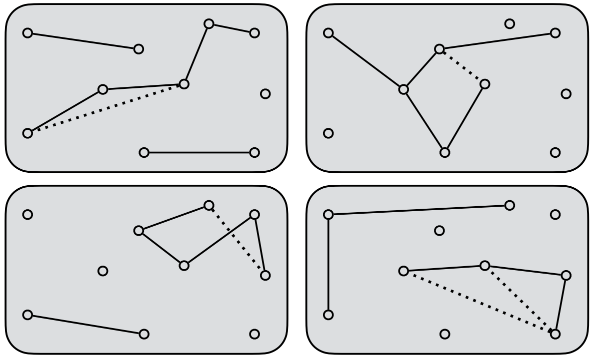
- Let G be graph with n nodes and m edges.
- Goal. MST in $O(1/\varepsilon)$ rounds whp. for $S = \Theta(n^{1+\varepsilon})$ and $P = \Theta(m/S) = \Theta(m/n^{1+\varepsilon})$
- Idea.
  - Repeatly filter edges not part of MST in rounds.
  - When all edges fit on one processor compute the MST directly.

Minimum Spanning Tree



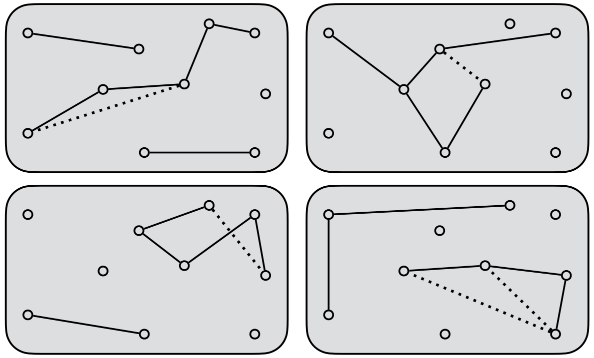
- Shuffle.
  - Let m' be the current edges. Initially, m' = m.
  - Choose $k = 2m'/n^{1+\varepsilon}$ active processors.
  - Distribute edges among active processors randomly.
  - Let E_i be the edges at processor i. $|E_i| = n^{1+\varepsilon}$ whp.

Minimum Spanning Tree



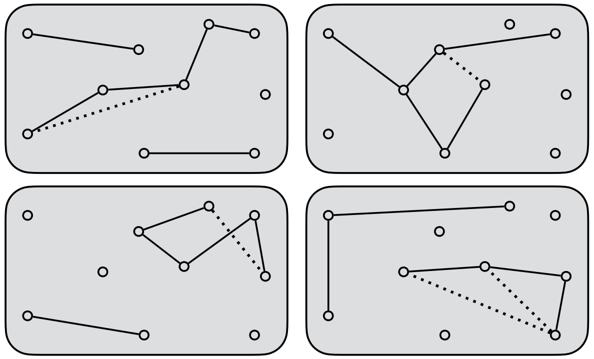
- Filter. Active processor i:
  - computes a local minimum spanning forest of $G = (V, E_i)$.
  - discards all other edges in E_i

Minimum Spanning Tree



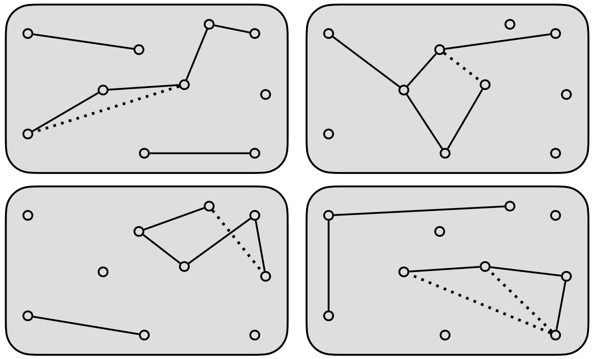
- Repeat.
  - Repeat shuffle and filter step until remaining edges fit on a single machine.
  - Then compute MST.

Minimum Spanning Tree



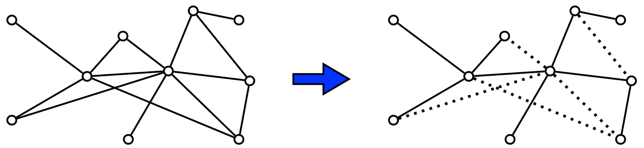
- Correctness.
  - Edges in E_i that are not in the local minimum spanning forest are not in the MST.

Minimum Spanning Tree



- Rounds.
  - Total edges remaining after a round is $\leq k(n-1) = \frac{2m'}{n^{1+\varepsilon}}(n-1) < \frac{2m'}{n^{\varepsilon}}$
  - $\implies$ A round reduces edges by factor n^{ε}
  - $\implies$ After $O(1/\varepsilon)$ rounds the remaining edges is $< n^{1+\varepsilon}$.

Minimum Spanning Tree



- Theorem. MST in $O(1/\varepsilon)$ rounds whp. for $S = \Theta(n^{1+\varepsilon})$ and $P = \Theta(m/n^{1+\varepsilon})$

Massively Parallel Computation

- Computational Model
- Summing
- Sorting
- Minimum Spanning Tree