

CS137: Electronic Design Automation

Day 12: February 6, 2006
Sorting



CALTECH CS137 Winter2006 -- DeHon

Today

- Sequential Sorting
- Building on Parallel Prefix
- Systolic
 - Sort
 - Priority Queue
- Streaming Sort
- Mesh Sort (Shear Sort)
- Sorting Networks
- Parallel Merge Sort

CALTECH CS137 Winter2006 -- DeHon

2

Sequential Sort

- What's your favorite sequential sort?
- Runtime?

CALTECH CS137 Winter2006 -- DeHon

3

Sequential Merge Sort

- **Observe:** can merge two sorted list of length N in $O(N)$ time
- Start with N lists of length 1
- Merge to form $N/2$ lists of length 2
- Merge to form $N/4$ lists of length 4
- ...how many times?
- Each merge?

CALTECH CS137 Winter2006 -- DeHon

4

Sequential Merge Sort

- **Observe:** can merge two sorted list of length N in $O(N)$ time
- Merge successively longer lists
- $\log(N)$ merges
- Each takes time $O(N)$
- Sort in: $O(N \log(N))$

CALTECH CS137 Winter2006 -- DeHon

5

Parallel Sorting

prefix

CALTECH CS137 Winter2006 -- DeHon

6

Rank Finding

- Looking for l 'th ordered element
- Do a prefix-sum on high-bit only
 - Know m =number of things $> 01111111\dots$
- High-low search on result
 - *i.e.* if number $> l$, recurse on half with leading zero
 - If number $< l$, search for $(l-m)$ 'th element in half with high-bit true
- Find l 'th element in $\log^2(N)$ time

Rank-based Sort

- In $O(\log^2(N))$ time on N processors can find the l 'th element
- Use separate groups of N processors to find the 1st, 2nd, 3rd, ... element in parallel
- Also count the number of such elements in $O(\log(N))$ time using parallel prefix
 - Give each unique offset
- Send each element to its correct position
- $\rightarrow O(\log^2(N))$ sorting algorithm with $O(N^2)$ processors

Rank Sort Analysis

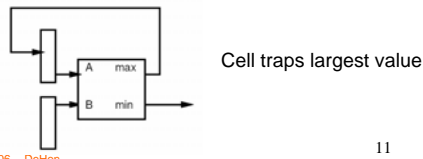
- Area N^2
- Time $\log^2(N)$
- Work: $(N \log(N))^2$
 - \rightarrow square of sequential work

Systolic

One Dimensional Array

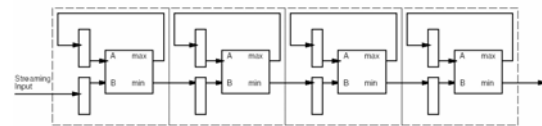
Sort as Data Arrives

- Often receive data as a sequential stream
- Can I sort the data as it arrive?
- Build a systolic solution?
 - Use only local interconnect



Linear Systolic Sort

- Often receive data as a sequential stream
- Can I sort the data as it arrive?
- Build a systolic solution?
 - Use only local interconnect



Linear Systolic Sort Analysis

- Area N
- Time N
- Work: N^2

CALTECH CS137 Winter2006 -- DeHon

13

Priority Queue

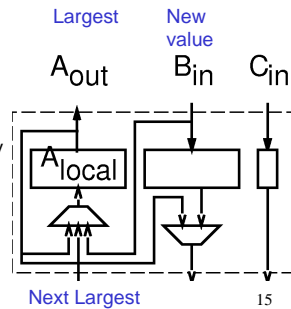
- Insert top
- Extract Largest
- With $O(N)$ cells
- $O(1)$ Extract
- Allows interleave insert/delete

CALTECH CS137 Winter2006 -- DeHon

14

Priority Queue Idea

- Trap Largest
 - Like Linear Sort
- Largest always at front
 - Always immediately available
- On extract
 - Shift up

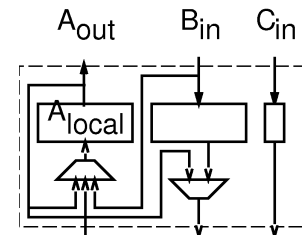


CALTECH CS137 Winter2006 -- DeHon

15

Priority Queue Cell

- If (C_{in} =insert)
 - $A_{local} \leftarrow \text{largest}$
 - $B_{out} \leftarrow \text{smallest}$
- If (C_{in} =extract)
 - $A_{local} \leftarrow A_{in}$
 - $B_{out} \leftarrow B_{in}$
- $C_{out} \leftarrow C_{in}$



CALTECH CS137 Winter2006 -- DeHon

16

Streaming Merge Sort

CALTECH CS137 Winter2006 -- DeHon

17

Streaming Sort

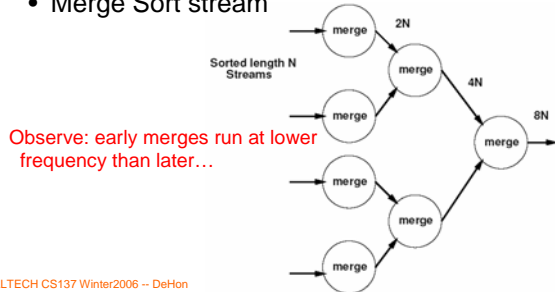
- Can we sort streaming data with $O(\log(N))$ hardware?
- How do you sort efficiently in SCORE?
 - Pipe-and-filter System Architecture?

CALTECH CS137 Winter2006 -- DeHon

18

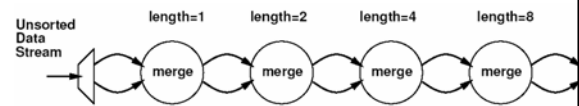
Build Merge Tree

- Merge Sort stream



CALTECH CS137 Winter2006 -- DeHon

Streaming Sort

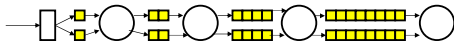


After $\log(N)$ merges, output stream is sorted.

CALTECH CS137 Winter2006 -- DeHon

20

Streaming Sort



- Buffer lengths grow by $2\times$ each stage.
- Total memory: $2\times(N/2) + 2\times(N/4) + 2\times(N/8) + \dots \leq 2N$

CALTECH CS137 Winter2006 -- DeHon

21

Streaming Sort Analysis

- Area $\log(N)$ compare/switch
 - $O(N)$ memory
 - [also true of sequential case]
- Time $O(N)$
- Work: $O(N \log(N))$
 - Work efficient

CALTECH CS137 Winter2006 -- DeHon

22

Mesh Sort

CALTECH CS137 Winter2006 -- DeHon

23

Mesh Sort

- Start with N items in $\sqrt{N} \times \sqrt{N}$ mesh
- Sort into specified order
- Nearest-neighbor communication only

CALTECH CS137 Winter2006 -- DeHon

24

Observation 1

- Can sort m things on linear array in $O(m)$ time
 - Perform Parallel Bubble sort in m steps
 - *i.e.* alternate odd/even swap pairings

Shearsort

- **Algorithm:** alternate sorting rows and columns for $\log(N)+1$ steps
 - *i.e.* sort rows on odd steps; columns on even steps
 - Sort odd rows ascending, even rows descending
 - Can use even/odd swapping for row/column sorts
- $O(\sqrt{N} \log(N))$

Simplifying Lemma

- **0-1 Sorting Lemma:** If an **oblivious comparison-exchange** algorithm sorts all input sets consisting of solely 0's and 1's, then it sorts all input sets with arbitrary values
 - proof in Leighton
- Odd/even swapping is an **oblivious comparison-exchange**

Shearsort Works?

- General form after column sort:
 - 0 rows
 - Mixed (dirty) rows
 - 1 rows
- Consider all row pairs:
 - 3 cases
 - More zeros, more ones, equal number
 - Row sort puts all zeros on one side, ones on other
 - Column sort \rightarrow one of the pair ends up all ones/zeros
 - Therefore, each row/column sort cuts the number of "dirty" rows in half

Shearsort Works?

- Consider all row pairs:
 - 3 cases
 - More zeros, more ones, equal number
 - Row sort puts all zeros on one side, ones on other
 - Column sort \rightarrow one of the pair ends up all ones/zeros
 - Therefore, each row/column sort cuts the number of "dirty" rows in half

```

10001000  row  00000011  column  00000000
10101001      11110000      11110011
    
```

Dirty Rows after column sort

Rounding up Steps

- Each sort $m=\sqrt{N}$ steps
- $\log(\sqrt{N})$ row/column sorts to remove dirty rows
- $2 \log(\sqrt{N}) = \log(N)$
- Total steps: $\sqrt{N} \log(N)$

Shear Sort Analysis

- Area N
- Time $\sqrt{N} \log(N)$
 - Best could hope to do is \sqrt{N} w/ nearest-neighbor connections in 2D world
 - Asymptotically in any 2D world
- Work: $N^{1.5} \log(N)$

CALTECH CS137 Winter2006 -- DeHon

31

Mesh Sort

- Can do Mesh sort in $O(\sqrt{N})$ steps
 - Best could hope to do
- More complicated...see Leighton

CALTECH CS137 Winter2006 -- DeHon

32

Extend to 3D Array

- Can sort N numbers on $N^{1/3} \times N^{1/3} \times N^{1/3}$ array in $O(N^{1/3})$ steps
 - Sort xz into zx order
 - Sort yz into zy -order
 - Sort xy into yx -order (reversing order on every-other plane)
 - Two-steps of odd/even merging within each z -line
 - Sort xy into yx -order

CALTECH CS137 Winter2006 -- DeHon

33

Sorting \propto Movement

- If you believe
 - We only have 3 dimensions
 - Signal transport is bounded by speed of light
- This is asymptotically tight
 - Cannot do any better.
 - Will take $O(N^{1/3})$ time just to transport an item from start location to destination

CALTECH CS137 Winter2006 -- DeHon

34

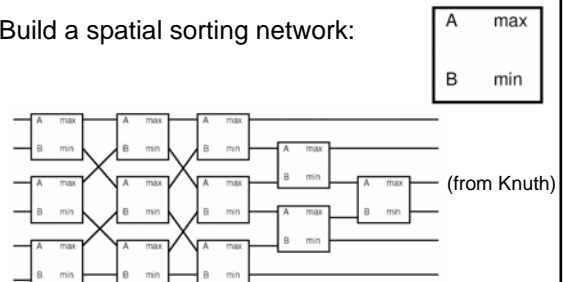
Sorting Networks

CALTECH CS137 Winter2006 -- DeHon

35

Sorting Network

- Build a spatial sorting network:

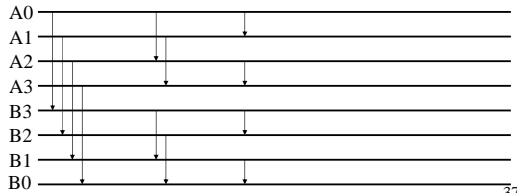


Too big, too fast? \rightarrow bit serial datapath elements?

CALTECH CS137 Winter2006 -- DeHon

Systematic Construction: Step 1: Merge Network

- Recursively swap large/small elements from halves of network
 - Merge in $\log(N)$ steps



CALTECH CS137 Winter2006 -- DeHon

37

Systematic Construction: Sorting Network

- Perform recursive merging
 - $\log(N)$ merge networks
 - Of depth $\log(N)$, $\log(N)-1$...
 - Depth: $O(\log^2(N))$
 - Area: $O(N \log^2(N))$
 - Can be used in pipelined fashion
 - Only using $O(N)$ hardware exclusively per step

CALTECH CS137 Winter2006 -- DeHon

38

Parallel Merge Sort

CALTECH CS137 Winter2006 -- DeHon

39

Parallel Merge Sort

- With $O(N)$ processors
- Sort in $O(\log^2(N))$ steps
- Sequentially executing the $O(\log^2(N))$ pairwise swaps of the sorting network
- Randomized algorithm
 - Works in $O(\log(N))$ steps
 - With high probability
 - ...see Leighton

CALTECH CS137 Winter2006 -- DeHon

40

Admin

- Wednesday, Friday: NC
- Project: two things due in two weeks
 - Sequential baseline
 - Proposed plan of attack

CALTECH CS137 Winter2006 -- DeHon

41