15-492: Parallel Algorithms **Lecturer:** Guy Blelloch
Topic: Sets and Sequences II **Date:** September 13, 2007
Scribe: Michael Sanphy

6.1 Merging

Contraction Method

- Choose even-indexed elements
- Recursively merge these
- Reinsert "left out" elements
- Fix up inversions

Input: $S_1 = [3, 5, 7, 9]$, $S_2 = [1, 2, 8, 1]$

Even elements, recursively merged: $[1, 3, 7, 8]$

You insert the missing elements as if they were paired up with the element preceding them in the original array. $R = [1, 2, 3, 5, 7, 9, 8, 11]$

To sort out the inversions that this creates, we can use a Scan with Max() as our binary associative operator. This works because we know that the array is 'mostly sorted' and will be sorted from one inversion to the next. After Scan is finished, we can run over the array performing the following (given in pseudocode) to put the array into sorted order. Here, 'current' is the current value of the element, 'new' is the value of Scan at this element, and 'next' is the value of the next element in the array.

```
if new < next then new
else if new < current then current
else next;
```

This gives the following recurrences:

$$W(n) = W(n/2) + O(n) = O(n)$$
$$D(n) = D(n/2) + \log n = O(\log^2 n)$$

6.2 Median / Order Statistics

Given int k and set S, find the k-th smallest element.

Approach: select(k, S)

- find some $e \in S$
- let $n = |\{s \in S : s < e\}|$
- if $n < k$
  select $(k - n, \{s \in S : s \geq e\})$
  else
  select $(k, \{s \in S : s < e\})$

6.3 Deterministic e-Selection

- Break array into blocks of 5
- Find the median of each block, put into set M
- $e =$ the median of M

Using this algorithm, select(k, S) has good work and depth because it 'throws away' $\frac{3}{10}$ of the array at each step. Without loss of generality, if $e < k$ then we can throw away all the elements in set M that were below e, and since those were themselves medians of a 5-element block, 2 more can be thrown away as well. That makes 3 out of every 5 for the lower half of the array, for 3 out of 10 in total.

$W(n) = W(\frac{n}{5}) + W(\frac{7}{10}n) + O(n) = O(n)$
$D(n) = D(\frac{n}{5}) + D(\frac{7}{10}n) + O(\log n) = O(n^{\alpha})$
$0 < \alpha < 1$ In fact $\alpha \approx 0.82$

6.3.1 Variant: breaking into blocks of $|\log n|$

$W(n) = W(\frac{n}{\log n}) + W(\frac{3}{4}n) + O(n) = O(n)$
$D(n) = D(\frac{n}{\log n}) + D(\frac{3}{4}n) + O(\log n) = O(n^{\beta})$
$0 < \beta < \alpha < 1$

However, because it is in blocks of log n, we can go below polynomial depth.
Find the median of $(\frac{n}{\log n})$ blocks by sorting each rather than recursively merging:

$W(n) = O(n) + W(\frac{3}{4}n) + O(n) = O(n)$
$D(n) = O(\log^2 n) + D(\frac{3}{4}n) + O(\log n) = O(\log^3 n)$

6.4 Sorting

- Quicksort $\rightarrow D = O(\log^2 n)$
- Mergesort $\rightarrow D = O(\log^2 n)$
- Radix Sort $\rightarrow D =$? (for a later lecture)
- Shell Sort $\rightarrow D =$? (too painful to think about)

- Bubble Sort $\rightarrow W = O(n^2), D = O(n)$
- Selection Sort $\rightarrow W = O(n^2), D = O(\log n)$
- Insertion Sort $\rightarrow W = O(n^2), D = O(n)$
- Heap Sort $\rightarrow$ not parallel at all

6.5 Sample Sort

Given P processors,

1. Select a set of $P - 1$ splitters
2. Based on splitters, send keys to P buckets
3. Sort buckets
4. Append buckets

Continued next time.