Odd-Even Transposition Sort

Ned Nedialkov

McMaster University
Canada

CS/SE 4F03
February 2015

Outline

Bubble sort

Algorithm

Input: an array A of size n
Output: sorted array A
Compute:
    for $i = n - 1$ down to 1
        for $j = 1 : i$
            compare-exchange$(A[j], A[j+1])$

How to parallelize it?

Example: odd-even transposition sort

phase															
	2		10		5		3		7		9		4		10
1. odd	2	↔	10		5	↔	3		7	↔	9		4	↔	10
2. even	2		10	↔	3		5	↔	7		9	↔	4		10
3. odd	2	↔	3		10	↔	5		7	↔	4		9	↔	10
4. even	2		3	↔	5		10	↔	4		7	↔	9		10
5. odd	2	↔	3		5	↔	4		10	↔	7		9	↔	10
6. even	2		3	↔	4		5	↔	7		10	↔	9		10
7. odd	2	↔	3		4	↔	5	↔	7	↔	9		10	↔	10
8. even	2		3	↔	4		5	↔	7		9	↔	10		10
	2		3		4		5		7		9		10		10

After n phases the sequence is guaranteed to be sorted

Example

Assume n numbers and p processes

Here $n = 16$, $p = 4$

	P_0	P_1	P_2	P_3
input	$35, 29, 42, 84$	$87, 28, 95, 79$	$54, 92, 64, 34$	$53, 14, 69, 70$
local sort	$29, 35, 42, 84$	$28, 79, 87, 95$	$34, 54, 64, 92$	$14, 53, 69, 70$
odd phase	$29, 35, 42, 84 \leftrightarrow$	$28, 79, 87, 95$	$34, 54, 64, 92 \leftrightarrow$	$14, 53, 69, 70$
even phase	$28, 29, 35, 42$	$79, 84, 87, 95 \leftrightarrow$	$14, 34, 53, 54$	$64, 69, 70, 92$
odd phase	$28, 29, 35, 42 \leftrightarrow$	$14, 34, 53, 54$	$79, 84, 87, 95 \leftrightarrow$	$64, 69, 70, 92$
even phase	$14, 34, 28, 29$	$35, 42, 53, 54 \leftrightarrow$	$64, 69, 70, 79$	$84, 87, 92, 95$
sorted	$14, 28, 29, 34$	$35, 42, 53, 54$	$64, 69, 70, 79$	$84, 87, 92, 95$

After p phases the sequence is guaranteed to be sorted

Algorithm outline

- Each process sorts its n/p numbers
- Perform p passes of odd-even interchanges
- After them, each process has its own sorted sequence

See e.g. http://en.wikipedia.org/wiki/Odd%E2%80%93even_sort#Algorithm

Analysis

Each process can sort in $O(n/p \log n/p)$

At each phase (odd/even)

- the communication is $O(n/p)$
- merging of two n/p sequences is $O(n/p)$

Since we have p phases, the total work for communication and merging is $O(n)$

Hence the parallel time is

$$T_p = O\left(\frac{n}{p} \log \frac{n}{p}\right) + O(n)$$

The serial time is (for the fastest sort)

$$T_s = O(n \log n)$$

$$S = \frac{T_s}{T_p} = \frac{O(n \log n)}{O\left(\frac{n}{p} \log \frac{n}{p}\right) + O(n)} = \frac{1}{O\left(\frac{\frac{n}{p} \log \frac{n}{p}}{n \log n}\right) + O(\frac{n}{n \log n})}$$

$$= \frac{1}{O\left(\frac{\frac{1}{p} \log \frac{n}{p}}{\log n}\right) + O(\frac{1}{\log n})}$$

$$E = \frac{1}{O\left(\frac{\log \frac{n}{p}}{\log n}\right) + O(\frac{p}{\log n})} = \frac{1}{O\left(1 - \frac{\log p}{\log n}\right) + O(\frac{p}{\log n})}$$

$$= \frac{1}{1 - O\left(\frac{\log p}{\log n}\right) + O(\frac{p}{\log n})}$$

How would this algorithm scale?