Parallelism

1 Purpose of this document

In the article it is mentioned that using an appropriate sorting strategy it is possible to execute up to $N/2$ comparators in parallel with a depth of $O(1)$ which leads to a total $O(N)$ depth. The strategy proposed there is a practical greedy algorithm; here instead the goal is to simply show that, for any graph, it is indeed possible to execute all of the comparators in $O(N)$ depth.

2 Premise

A complete graph of N nodes is considered, since a solution for it would also work for any graph with less arcs. A comparator operates on two nodes and has a depth of $O(1)$; also, any number of comparators can be executed with the same cost as just one comparator if they all operate on different nodes.

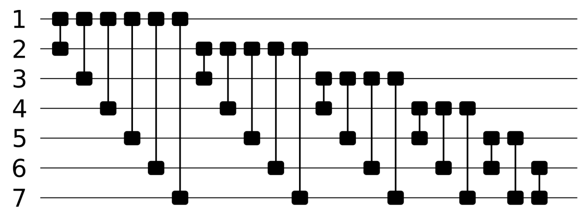
3 $O(N)$ order

In Figure 1 an example is shown for a complete graph of seven nodes. Comparators are represented by two boxes, indicating the nodes used, joined by a line. In 1a the comparators are grouped by node.

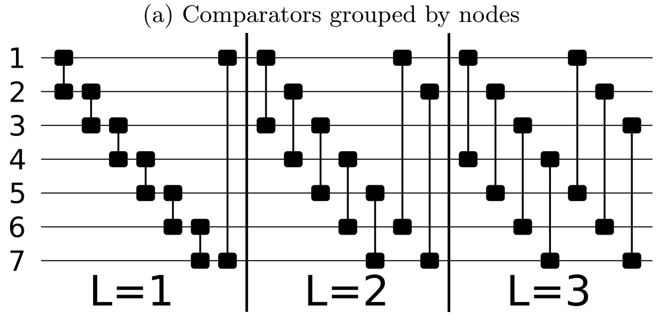
Let's number the nodes of the graph from 1 to N. We can divide the arcs into groups the following way: with L going from 1 to $N/2$, for each node i, take the arc that joins it with node $(i + L) \mod N$. This yields us $N/2$ groups of N arcs, for example, see Figure 1b. We can further subdivide each group based on whether, given i the number assigned to the starting node, $\lfloor i/L \rfloor$ is even or odd. Finally we can subdivide the arcs again based on whether $i + L > N$. The resulting $N/2 \cdot 4 = O(N)$ groups are fully parallelizable (and so can be executed in $O(1)$), meaning that the overall depth is $O(N)$. For our example, Figure 1c shows the final order, with the three L groups further subdivided in four columns.

Why are the final groups fully parallelizable? The L-grouping guarantees that no two comparators have the same "starting" node. The subgrouping guarantees that no comparator within a group has an "ending" node that is some other comparator's "starting" node. That's because if, given i the starting node, $\lfloor i/L \rfloor$ is even, then for the ending node $i + L$ the opposite holds: $\lfloor (i + L)/L \rfloor$

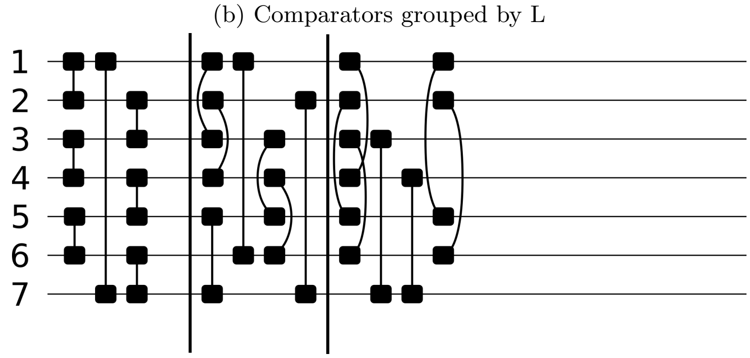
is odd (and viceversa) and so the comparator which has $i + L$ as the starting node belongs to a different subgroup than the one who has it as an ending node. This however does not account for comparators with a starting node i such that $i+L > N$ and the ending node is $(i+L) \mod N$. Those comparators can cause conflicts, so another split is needed, which brings us to the final subgrouping.



(a) Comparators grouped by nodes



(b) Comparators grouped by L



(c) Final comparator order/grouping

Figure 1: Comparators orders