University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
S. Faour, M. Fuchs,
Z. Parsaeian, G. Schmid

Theory of Distributed Systems
Exercise Sheet 12

Due: Wednesday, 26th of July 2023, 12:00 noon

Exercise 1: Aggregation in the MPC Model *(10 Points)*

Assume you are given a number of $M \in O\big(\frac{N}{S}\log_S N\big)$ machines, where N is the number of *aggregation messages* that are collectively stored by the machines M_i, $i \in \{1, \dots, M\}$. Each machine M_i has a memory large enough to store S such messages. By definition of the MPC model every machine can send and receive at most S aggregation messages per round.

Each aggregation message m is a triple consisting of the aggregation value v_m, the target machine t_m and an aggregation group g_m. All messages in the same group go to the same target and each machine is the target of not more than $S/2$ aggregation groups. The *aggregation problem* is solved when every target machine t_m learns an aggregation message m that has *minimal* value among all aggregation messages of its aggregation group g_m.

In the following we give an algorithm that solves the aggregation problem under the assumptions that the intial aggregation messages are stored on $2\lceil\frac{N}{S}\rceil$ machines and none of those machines is a target of an aggregation message. We also assume that there are no more than $2\lceil\frac{N}{S}\rceil$ target machines. Further, assume that we have sufficient long string of "public random bits", which can be used to make random decisions that are the same for all machines, (since all machines utilize the same random bit string). Your task is to show that in the given algorithm no machine sends or receives more than $S/2$ messages per round *in expectation*. After how many rounds will the algorithm terminate? Explain!

Algorithm: We arrange the $O\big(\frac{N}{S}\log_S N\big)$ machines into $\ell := 1 + \lceil\log_{S/2} N\rceil \in O(\log_S N)$ levels $L_1, \dots, L_\ell$ of $2\lceil\frac{N}{S}\rceil$ machines each, which is abstractly shown in Figure 1. We will explain the exact value of ℓ later in the analysis. Furthermore, we arrange the levels such that initially all messages are held only by machines in L_1 ("message sources"), and the targets of the aggregation messages are in level L_ℓ.

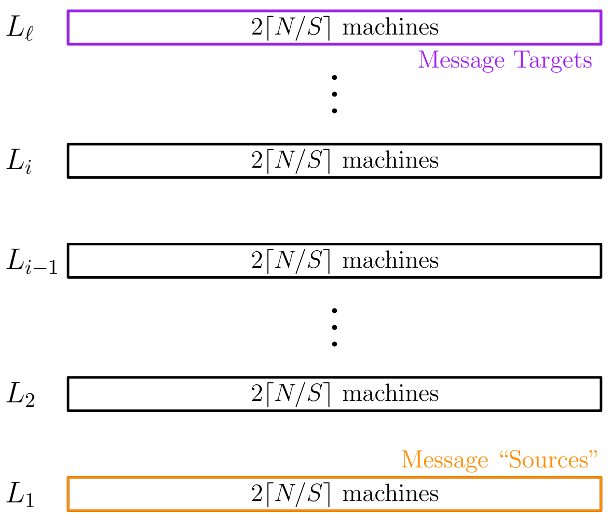
L_ℓ
$2\lceil N/S\rceil$ machines
Message Targets
L_i
$2\lceil N/S\rceil$ machines
L_{i-1}
$2\lceil N/S\rceil$ machines
L_2
$2\lceil N/S\rceil$ machines
Message "Sources"
L_1
$2\lceil N/S\rceil$ machines

Figure 1: Arrangement of machines into levels.

Outline: The idea of solving the above aggregation problem is to establish aggregation trees between machines of successive levels. There will be one aggregation tree for each aggregation group with leaves in L_1 and roots in L_ℓ. The messages are then sent up the trees like in a convergecast. Machines will choose their according tree parents in the next level randomly, which ensures that no machine obtains too many messages *in expectation*.

Aggregation trees on first level: For level 1, we say that a machine in L_1 that has a message m with the aggregation group g_m, participates in the aggregation tree of that group g_m (a given machine can participate in multiple aggregation groups, as it holds multiple messages).

For each aggregation group g the machines in L_1 choose a random subset L_g^2 from the next level L_2 of size $|L_g^2| = N \cdot (\frac{2}{S})^2$. Note that all machines participating in the aggregation tree of group g can all agree on the same random set L_g^2 using the public randomness.

Then each machine that takes part in group g picks a random parent node from L_g^2. Note that this random decision is now independent from the parent choice of other machines! By doing this for all machines in L_1 and for aggregation groups, each machine in L_1 will now have a parent node for each aggregation group it participates in, see Figure 2.

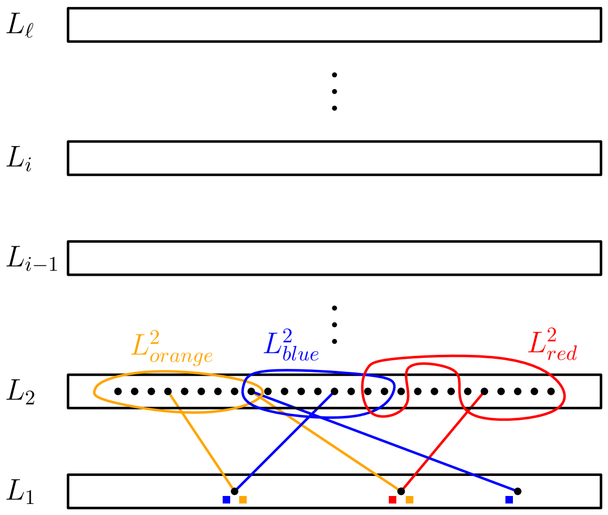

Figure 2: Three example machines in L_1 that have messages (little colored boxes) from three aggregation groups (orange, blue, red). We determine random sets of machines from $L^2_{orange}, L^2_{blue}, L^2_{red}$. Each machine picks a random parent from the according random set for each group it participates in.

Aggregation trees for subsequent levels: The description above forms the base case above and now we describe how to connect L_{i-1} to level L_i for $2 < i \leq \ell - 1$. We say that a machine $\mu \in L_{i-1}$ participates in aggregation group g if it is in the according set $\mu \in L_g^{i-1}$.

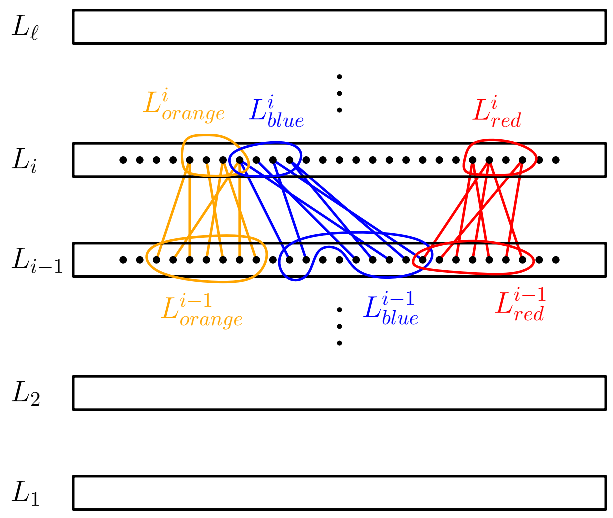

Figure 3: Nodes in L_g^{i-1} choose a parent uniformly at random from L_g^i ($g =$ orange, blue, red).

Similar as before, for each aggregation group g the machines in L_{i-1} choose a random subset L_g^i from the next level L_i of size $|L_g^i| = N \cdot (\frac{2}{S})^i$. Again, all machines agree on the same random sets L_g^i using public randomness. Let μ be a machine that participates in aggregation group g. As before, it chooses a parent in L_g^i uniformly at random (and also independent from other machines), c.f. Figure 3.

We have to make a distinction for the last level ℓ. There we simply connect all nodes in aggregation group $L_g^{\ell-1}$ to the respective target $t \in L_\ell$ (c.f., Figure 4).

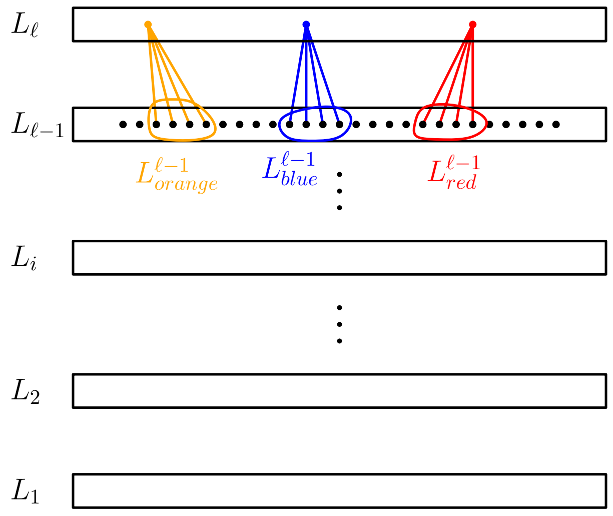

Figure 4: Nodes in $L_g^{\ell-1}$ create an edge to the target machine of group g ($g =$ orange, blue, red).

Aggregation Algorithm: In round i, every machine in level i sends for every aggregation group g_m *one* message m with smallest value v_m among all messages that this machine has from that group g_m to its parent in $L_{g_m}^{i+1}$.

Exercise 2: Implement a Phase of Borůvka's Algorithm *(10 Points)*

In class, we sketched how to implement one phase of Borůvka's MST algorithm in the strongly sublinear regime $S = n^\alpha$ for some constant $0 < \alpha < 1$. Argue in more detail how this can be done in $O(1)$ rounds, given that we can solve the above aggregation problem.