Stable Matching Report

Bjarke Brodin (bjal), Bjørnar Haugstad Jåtten (bjja), Helle Friis (hefr) and Simon Boye Jørgensen (sboj)

September 2, 2022

Results

Our implementation produces the expected results on all input-output file pairs, however we have a different ordering of the outputted matches. This could either be due to the data structure we use to keep track of the proposers, or simply the ordering of the pairs when output. We use a stack for this purpose, putting proposers back on the stack whenever they are rejected or discarded (in favor of a better proposer).

Implementation details

The proposers preferences are stored in a HashMap, where the key is the ID of the specific proposer and the value is a queue composed of their preferences. The rejecters preferences are stored in a HashMap where the key is the ID of the specific rejecter. The value is another HashMap where the key is the ID of the proposer and the value is the rank of given by the rejecter to the proposer.

We can find a free proposer who has not proposed to every rejecter in time $O(1)$, because we store free proposers on a Stack with $O(1)$ pop operations.

With these data structures, our implementation runs in worst-case time $O(n^2)$ on inputs with n proposers and n rejecters, we can safely set this bound because all operations are $O(1)$, expect reading the input which is $O(n)$, and considering each possible of the $O(n^2)$ pairs at most once.