1 Description Heuristics

1. **Group 1**
   - Language: Java
   - Initialization: distance heuristics
   - Next function: sorting the random nodes by their probabilities in the current Markov chain.

2. **Group 2**
   - Language: Java
   - Initialization: distance heuristics.
   - Next function: sorting

3. **Group 4**
   - Language: Java
   - Initialization: value improvement for a bounded number of times.
   - Next function: sorting, swap did not work.

4. **Group 5**
   - Language: Java
   - Initialization: distance-based.
   - Next function: sorting.

5. **Group 6**
   - Language: Java
   - Initialization: distance-based.
   - Next function: reach-based search.

6. **Group 7**
   - Language: Python
   - Initialization: don’t care.
   - Next function: sorting + lexicographic order.
2 Conclusions

Goal: Find an example of a stopping SSG and random permutation $p$ such that the algorithm in Figure 1 does not terminate, or prove that the algorithm always terminates.

Data: permutation $p$
while $p$ is not self consistent do
  $p \leftarrow \text{sort}(p)$;
end
return $p$;

Figure 1: The structure of the sorting-based algorithm.

Ideas for counterexamples: Use at least 3 random nodes.