1 Structural Patterns Heuristics via Fork Decomposition

1.1 Summary

One of the best heuristics developed over the last twenty-five years is to measure the amount of work needed to solve a simpler version of the original problem. There is a fundamental tension in this scheme between the effort it takes to solve the simpler version and the informativeness of measuring that effort, with respect to the original problem. Typically, the simpler problems are generated from the original by ignoring some of the variables in the original problem. As we ignore more variables, the problem becomes easier to solve but less informative about the original problem. The paper proposes a method of choosing variables to ignore which would permit larger (and therefore more informative) sets of variables while still guaranteeing that the generated problem is solvable. The paper uses the $SAS+$ problem representation and makes the claims that, if variables can be found with small domains, then substructures (particularly, a node and all of the children nodes fanning out of it, or a node and all of its predecessors) of the causal graph can be extracted that are solvable in polynomial time. The paper concludes with a theoretical analysis of tightness of the lower bound which I didn't really follow.

1.2 Significance

This paper extends the previous papers we discussed about additive heuristics and pattern databases. In that previous work, the actions had to be completely disjoint so that, when we summed the heuristic cost given by any database, we could be assured that no action cost was being double counted. In this paper, that constraint is relaxed that so that the cost for an action is split across the various databases. When we sum up the action costs across all the databases, we still don't exceed the original action cost. That is an interesting generalization and would seem to allow a lot of alternative additive formulations.

The authors also make the following interesting observation on bounds of the size of simplified domains. Since we wish to compute heuristics in polynomial time, and computing a planning problems is exponential in either the number of variable values, this means that the size of suitable simplified problems is logarithmic in the size of the original problem (unless we exploit problem structure). It is easy to believe that logarithmic heuristic functions would be weak predictors of the difficulty of the full problem, so the search for problem structure to exploit and how to exploit it seems very important.