By Jordan L. Carson

For my Research Review, I choose the Research Paper by Haslum, Helmert, and Hoffman titled 'Explicit-State Abstraction: A New Method for Generating Heuristic Functions'.

A lot of research in the realm of computer science has been dedicated to planning search. That is finding the optimal path (shortest or least cost) in a discrete search space. This can be related to the uniform cost search heuristic as this method attempts to find an optimal path by selecting the least cost when moving from one node to the next. In these specific applications, the state spaces are explored through implicit defined directed graphs, whose nodes are assignments to a set of state variables and whose edges correspond to state transitions.

In this paper, Hoffman et al. introduce a new method for generating heuristic functions. That is, to reduce the size of the state space by aggregating several states into one called an abstraction. The cost of reaching a solution state in the abstract state space is a lower bound on the same cost in the original state space. The best way to represent the mapping in a way that is compact and permits efficient indexing is to consider a restricted class of abstractions. In particular, pattern databases (PDBs) have been explored and shown to be very useful in planning search. The abstraction underlying a PDB ignores completely all but a subset of the state variables, known as the pattern: states that do not differ on the chosen variables are aggregated in the abstract space. The distinguishing feature of these abstractions is that they explicitly select particular pairs of states to aggregate. "The state space of the problem is viewed as the synchronized product of its projections onto the single state variables. Each such projection is an abstract space and the product is computed by iteratively composing two abstract spaces, replacing them with their product." The resulting heuristic accuracy is dependent on the order in which composites are formed and the choice of abstract states to aggregate.

Hoffman et al. also introduce a concept called explicit-state abstractions, in a generic form based on labelled transaction systems. In essence, an explicit-state abstraction is created through an iterative construction of steps replacing two transaction systems with their synchronized product, and in between these steps they abstract by aggregating pairs of states. Both time and space are kept under control by enforcing a size limit, specified as an input parameter *N*. The output layer and each computed transaction system consists of at most *N* states. The quality of the resulting heuristic is determined by an abstraction strategy, a combination of a particular composition and shrinking strategy that is dependent on the application domain. Indifference to pattern databases, explicit-state abstractions can concisely represent exponentially many equivalent options. Explicit-state abstractions have been put to use in cost-optimal planning and model checking.

There are many applications to using explicit-state abstractions in cost optimal planning. Such as planning tasks and transaction systems, model checking, and any problem of finding an optimal path in a discrete state space. Hoffman et al. mention that if one were to plug the explicit-state abstraction heuristic into A*, one obtains a planner that is highly competitive and sometimes superior to a state-of-the-art in optimal sequence planning.

These explicit-state abstractions generalize pattern databases by explicitly selecting pairs of states to aggregate, rather than aggregating all states that agree on the state variables of a given pattern. Other areas of cost-optimal search have also had much success in this area such as topological sorting. It would be interesting to compare the latter approach to A*, and explicit-state abstraction heuristics, but it still may be dependent on the problem domain and the order of which one can successfully order / sort the state variables.