Research Review of Solving planning-graph by compiling it into CSP

Introduction:

This paper describes GP-CSP, a system that does planning by automatically converting Graphplan's planning graph into a CSP encoding, and solving the CSP encoding using standard CSP solvers.

Studies show that GP-CSP's dominance over standard Graphplan is in terms of runtime, its advantages over Blackbox's SAT encodings include improvements in both runtime and memory consumption.

CSP encodings also provide several structural advantages over SAT encodings which can be exploited in developing directed partial consistency enforcement algorithms that are suitable for planning encodings.

GP-CSP uses the CSP encoding of the planning graph. The basic idea is to let Graphplan build the planning graph representation, and convert into a CSP encoding.

Implementation details:

1. Remove all irrelevant nodes from the planning graph.
2. Each of the propositions is given a unique CSP variable number, and the actions are given unique CSP value numbers
3. The domains of individual variables are set to the the set of actions plus one distinguished value (upside down T) corresponding to for all propositions in levels other than the goal level. The null value is placed as the first value in the domain of each variable.
4. Setting up the constraints: Three types of constraints called, respectively activity constraint, fact mutex constraint and action mutex constraint.
5. Checking the constraints.

Results:

1. Run time:Graphplan is better for serial and parallel blockworld domains, and worse for the logistic, in which GP-CSP and two SAT solvers are quite close in most of the problems.
2. Memory:GP-CSP taked less memory for most logistics problems except in the parallel blocks world domain taken from the HSP suite.
3. Length of the plans returned by each solver:Solution returned by GP-CSP is strictly better or equal to Blackbox using either SATZ or Relsat for all tested problems. However, for all but one problem, the standard directional backward search of Graphplan returns shorter solutions.

Improvements to CSP solver:

The enhancements investigated: (1) explanation based learning (EBL) (2) level-based variable ordering, (3) random restart search with cutoff limit on backtracks (4) distance based variable and value ordering (5) min-conflict value ordering, and (6) the use of bmutex constraints.

Conclusion and Future directions:

since most AI-based schduling systems use CSP encodings, GP-CSP provides a promising avenue for attempting a principled integration of planning and scheduling phases. that communicate with each other by exchanging failure information in terms of no-goods.

References:

Solving planning-graph by compiling it into CSP by Minh Binh Do & Subbarao Kambhampati
http://rakaposhi.eas.asu.edu/1MDo00.pdf