Synthesis

State Of The Art And ICAPS Reviews

Antoine GRÉA

1 ICAPS Review Analysis

The review of my article in the context of ICAPS is pretty interesting : The reaction is strongly oriented toward two axis : The redaction and the (lack of) placement / motivations.

1.1 Claims And Judgments

Here is a list of claims done by the reviewers :

- The paper is poorly written and verbose, This is making a *draft* impression on the reviewers.
- Some mistakes have been brought to attention.
- The "solution" to unsolvable problems has hit a point with them and need heavy rewriting.
- Papers has been suggested along with leads on POCL and an interesting question about how to soft solve in specific domains
- Motivations for soft solving are needed as its own part like on the "good excuses" paper
- The state of the art is shown as not done or not done well enough
- Closed world assumption should be respected, along with more classical way of doing (like STRIPS, etc)
- Heuristics are said to solve most of these issues
- Lifted model is asked for quite strongly.
- The experiment seems to not make "a convincing point".

1.2 Response

My idea of a response is a total *rewriting* of the article focused on the new version of SODA. Here are the improved points it needs to have :

- Strongly checked language and style
- Clear and efficient Plan
- A SOTA part that states the exact placement of this new paper relative to other

works.
- A motivations part that states precisely why we are doing all this.
- The new SODA should be : Lifted with a clear language STRIP compatible, using the closed world assumption, Heuristic driven and well optimized
- Additional points will be made on Online planning and the such.
- Experiments should use precise metrics and must show clearly the performance profile of each way of doing on at least a standard domain. Plan quality metrics should be convincing and widely used if possible.

2 Studied Papers

> *All the following is mostly citations of important parts of the referenced article along with comments on said parts.*

2.1 A Good Excuse

2.1.1 The Closed World Assumption

The initial state is specified by providing a set s 0 of ground atoms, e.g., (holding block1) and ground fluent assignments, e.g., (= (loc obj1) loc2) . As usual, the description of the initial state is interpreted under the closed world assumption, i.e., any logical ground atom not mentioned in s 0 is assumed to be false and any fluent not mentioned is assumed to have an undefined value.

2.1.2 Motivations !!!!

a household robot was developed that uses a domain independent planning system. More often than not a sensor did not work the way it was supposed to, e.g., the vision component failed to detect an object on a table. [...] Obviously, thinking ahead of everything that might go wrong in a real-life environment is almost impossible [...] but it should also be able to tell the user what went wrong and why it couldn't execute a given command.

2.1.3 Good Excuse

Given an unsolvable planning task Π = Δ, C Π , s 0 , s ∗ , an excuse is a tuple χ = C χ , s χ that implies the solvable excuse task Π = Δ, C χ , s χ , s such that C Π ⊆ C χ and if (f = x) ∈ s 0 s χ (where denotes the symmetric set difference) then f must not contribute to s ∗ . [...] Given two acceptable excuses, it might nevertheless be the case that one of them subsumes the other if the changes in one excuse can be explained by the other one.

2.1.4 Causal Graph

In order to analyze the relations between fluent symbols, we apply the notion of causal graphs and domain transition graphs (Helmert 2006) to the abstract domain description. The causal graph CG Δ of a planning domain Δ = T , C Δ , S, O is a directed graph (S, A) with an arc (u, v) ∈ A if there exists an operator o ∈ O so that u ∈ pre(o) and v ∈ eff(o) or both u and v occur in eff (o). If u = v then (u, v) is in A iff the fluents in the precondition and effect can refer to distinct instances of the fluent. The causal graph captures the dependencies of fluents on each other

We can even put variables information on edges !

operator o ∈ O so that f = u is contained in pre(o) and f = v ∈ eff(o). The label consists of the preconditions of o minus the precondition f = u.

For example, the domain transition graph of the robot pos fluent has one vertex consisting of a variable of type room and one edge (room , room) with the label of connected (room 1 , room 2 , door) ∧ open (door).

The relevant domain, dom rel (f), of a fluent f is defined by the following two conditions and can be calculated using a fixpoint iteration: If f = v contributes to the goal, then v ∈ dom rel (f). Furthermore, for each fluent f on which f depends, dom rel (f) contains the subset of dom(f) which is (potentially) required to reach any element of dom rel (f).

2.2 Refinement Planning (State Based)

> *The complexity of plan synthesis depends on a variety of properties of the environment and the agent, including whether (1) the environment evolves only in response to the agent's actions or also independently, (2) the state of the environment is observable or partially hidden, (3) the sensors of the agent are powerful enough to perceive the state of the environment, and (4) the agent's actions have deterministic or stochastic effects*

2.2.1 Closed World Again

The initial state is assumed to be specified completely; so, negated conditions (that is, state-variables with false values) need not be shown.

2.2.2 How Important Is Least Commitment?

least commitment refers to the idea of constraining the partial plans as little as possible during individual refinements, with the intuition that over committing can eventually make the partial plan inconsistent, necessitating backtracking. [...] the first thing to understand about least commitment is that it has no special exclusive connection to ordering constraints [...]

2.3 Design Tradeoffs

> *The special step to is always mapped to the dummy operator start, and similarly t, is always mapped to finish [...] A ground linearization (aka completion) of a partial plan P : (T, O, H, ST, £) is a fully instantiated total ordering of the steps of ~ that is consistent with O*

2.3.1 Factors Of Performance

- branching factor !

- average cost per invocation
- effective depth of the search
- goal selection cost
- consistency check cost

2.4 Encoding Planning Constraints Into POP !

> *the user specifies additional constraints on the problem solution in order to: give a limit to the length of the solution plan, use or avoid specific action instances [...] Often solution plans that simply achieve end goals are unsatisfactory since real user have more complex goals. Like goal of having temporal precedences among activities or attainment goals.*

2.4.1 The Ancestor Of Ramirez !

In order to force the search space to produce plans which verifies the extended constraints, some dummy elements are introduced into the domain (i.e. dummy operators, initial facts and goals) and since the equivalent problem and its solutions will contain these dummy elements, dummy operators are to be removed in order to produce a solution to the original constrained problem through a solution decoding phase.

OMG this is the ONE !

2.4.2 Constraints

C is a PCDL constraint. A solution for a constrained problem P is a partial plan such that solves the unconstrained version of P and satisfies C .

2.4.3 Dummy, Dummy! DUMMY !

These techniques add some dummy facts to the initial states and/or the goals and/or the preconditions or effects of some existing operators. They can sometimes add some new dummy operators to the domain.
The presence of dummy items causes the solution plan to be sometimes expressed in a language richer than the original planning language.

I love this guy !

Hence a post compilation phase is needed to translate the solution in the old language, i.e. by deleting dummy operators.

2.4.4 Ramirez Before Ramirez

The presence constraint AT-LEAST-ONCE (a) is translated by inserting a new dummy goal u a into the user goals G and adding it to the effects of a