Accelerating Partial-Order-Planners: Some Techniques for Effective Search Control and Pruning

Paper Author: Gerevini and Schubert

Presenter: John (Trey) Boggess

Setup

- This Covers pages 95-112 (1-18 of PDF)
- Sections 1-3 of the paper
- Main Question: “What is a zero-commitment choice and how are they helpful?

Introduction

- The writers of this paper saw ample opportunity to bring UCPOP into practicality using their Zero-commitment Last-In First-Out method (ZLIFO)
- Their method includes a (Steps) + (Open Conditions) as a heuristic measure for UCPOP’s A* search of the plan space
  - Term for Unsafe Conditions Is also useful sometimes
- Flaw selection is ZLIFO
  - Prefer zero commitment plan refinements to others
- Improvement factor with UCPOP and ZLIFO ranges from 5x to 1000x

UCPOP Review

STRIPS-Like Operators

Starts with a true list, all else is false

Allows conditional effects

- If y not table, y not clear at end
- If z not table, z clear at end

Universally quantified conditions and effects are permitted as well

- Permissible to have precondition for PICKUP(x) that says for all y, not (on (y,x))

Parts

- Steps : Actions
- Causal Links: “Step produces goal condition for other step to run (consume)”
- Set of Binding Constraints: unifier for action effects to achieve goal
- Set of Ordering constraints

UCPOP cont.

- Threats are either definite or potential:
  - Definite: the unification that confirmed involved no new binding of variables (not a theoretical collision)
  - Potential: The unification confirmed involved new bindings of variables (theoretical collision)
- UCPOP has *d-sep* which says only definite threats are dealt with

Trouble with Counting Unsafe Conditions

- Choice of next plan in UCPOP is an A* best-first search
    - A* uses a heuristic estimate f(p) of overall solution cost with parts:
        - g(p) = cost of current partial solution (plan) p
        - h(p) = estimate of additional cost of the best complete solution that extends p.
    - f(p) is a measure of complexity
- We need to know two things:
    - For A* to guarantee discovery of optimal plan, h(p) should not overestimate remaining solution cost
    - If goal isn't optimum solution but find solution quickly then f(p) can be augmented to include a term that estimates the remaining cost to find a solution (base solution of speed of computing rather than efficiency)

What does A* use?

S is Steps: actions

OC is open conditions: unsatisfied goals and preconditions

CL is causal links

UC is unsafe conditions

Default is S+OC+UC^4. This becomes S+OC+UC+F (facts) but fact are ignored, only mentioned because they followed UCPOP strategy of inclusion where relevant

How good are these parts?

- S is a good generic complexity measure
- OC is a good estimator for residual plan complexity
- CL appears to be an alternative to step counting. But is generally larger than S. if CL is used in addition to / instead of S, it makes achieving multiple subgoals with a single step undesirable because the larger CL is in comparison to S indicates number of subgoals achieved by action re-use. It is not included in f(p)
- UC is not a g-measure. Threats are not part of a plan complexity. The paper also decides it should not be in the h measure either due to the O(n) nature of threats appearing and expiring where n is steps. It also isn’t safe to use in remaining complexity due to the fact threats appear and disappear constantly
- Paper decides on using S+OC

Goal Selection Strategy

- An important opportunity for improving planning performance independently of the domain lies in identifying forced refinements, or refinements that can be made deterministically.
- Specifically in considering possible refinements to a given partial plan, it makes sense to give top priority to open conditions that cannot be achieved; and then open conditions that can only be achieved one way

Why?

- The argument for giving top priorities to unachievable goals is just that a plan with this goal can be eliminated.
- The argument for open conditions achievable one way is since every open condition must be established by some action, it follows that if this action is unique, it must be a part of every possible plan. This is what we call a “zero-commitment action” involving no choice or guesswork.

Refinement Optimization

- At the same time, adding any refinement in general narrows down the search space by adding binding constraints, ordering constraints, and threats, which constrain existing and not-yet existing steps.
- For unique refinements this is monotonic, never needing revocation. (AKA it won't be constrained multiple times)
  - For example,suppose some refinement happens to add constraints that eliminate a certain action instance A as a possible way of achieving a certain open condition C. If the refinement is unique, then we are assured that no completion of the plan contains A as a way of establishing C.
- In short, zero commitment strategy cuts down search space without loss to viable solutions.

Other Methods (for comparison)

Least Commitment (LC)

- Least commitment always selects an open condition which generates the fewest refined plans. So it entails the priorities for unachievable and uniquely achievable goals while also entailing a certain prioritization of nonuniquely achievable goals.

Least Cost Flaw Repair (LCFR)

- uniformly apply LC strategy to both threats and open conditions
- Combining this with UCPOP’s plan selection, they obtained significant search reductions, but not time due to overhead used for repair

These won’t be discussed further, but were compared to ZLIFO for tests

Zero-Commitment Last-In First-Out (ZLIFO)

- Chooses next flaw with this criteria:
  - A definite threat (*d-sep* is turned on) using LIFO to pick among them
  - An open condition that cannot be established in any way
  - An open condition that can be resolved in only one way, preferring open conditions that can be established by introducing a new action to those that can be established by using start
  - An open condition, using LIFO to pick
- The overhead used by ZLIFO is limited to the open conditions and is lower than least cost flaw repair and least commitment.

Results

- This method performed better than either LC or LCFR except for certain easier situations.
- For example, the Train problem (Shown later). For the easiest problem, it performed worst. As the difficulty increased ZLIFO outperformed all competitiors

Example

5 cities (ABCD)

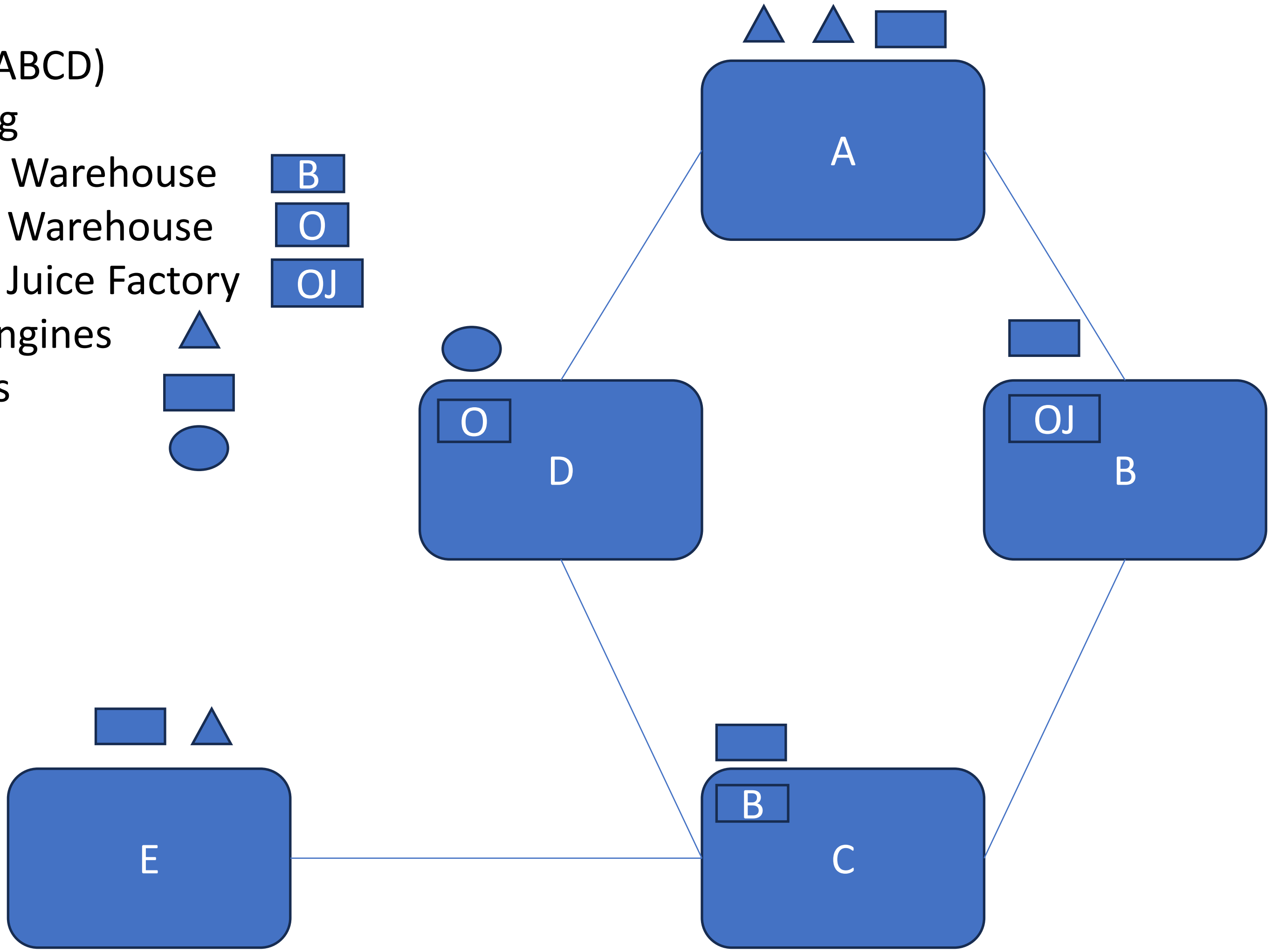
3 building

- Banana Warehouse B
- Orange Warehouse O
- Orange Juice Factory OJ

3 Train Engines ▲

4 Boxcars ▬

1 Tanker ⬬



Goals

1- Bananas in City A

2- OJ in city E (No OJ is made yet)

3- Oranges in city C

4- OJ in City D