LPG: Local search for Planning Graphs

Seung H. Chung

16.412J
Cognitive Robotics

Outline

• Temporal Action Graph
• Walksat: Stochastic Local Search
• Better Neighbor
• Relaxed Plan

Overview

- Uses Strips-like operator but adds time and metric resources to the planning description: Planning Domain Definition Language (PDDL) 2.1

- Features:
  - Uses Temporal Action graphs (TA-graphs): Similar to a plan graph, but adds temporal representation
  - Uses stochastic local search: Similar to Walksat
  - Uses relaxed plan for heuristic to guide the search: similar to FF

- LPG outperformed all general purpose planners in the time and metric resource domains (3rd IPC)
Temporal Action Graph (TA-Graph)

Ordering Constraint

Causal Constraint
- \( a_1 < a_4 \)
- \( a_2 < a_3 \)

Exclusive Constraints
- \( a_1 < a_2 \)
- \( a_2 < a_3 \)
- \( a_2 < a_4 \)
Walkplan: Stochastic Local Search

- **Walkplan**($\Pi$, max_steps, max_restarts, p)
  - Input
    - $\Pi$: Planning problem description
    - max_steps: Maximum number of search
    - max_restarts: Maximum number of restart
    - p: Noise factor
  - Output
    - Solution TA-graph

- **Idea:**
  - With probability p use stochastic local search to find a plan
  - Search the plan space max_steps number of times
  - If no plan is found, try restarting the search from the beginning up to max_restarts number of times

Walkplan Algorithm

```plaintext
Walkplan(\Pi, \text{max\_steps, max\_restarts, p})
for i = 1 to max_restarts do
    A = an initial TA-graph derived from \Pi
    for j = 1 to max_steps do
        if A is solution then return A
        $\sigma$ = an inconsistency in A
        $N(\sigma, A)$ = neighborhood of A for $\sigma$
        if $\exists A' \in N(\sigma, A)$ such that $A'$ is no worse than A then
            A = A'
        else if random < p then
            A = A' \in N(\sigma, A)
        else
            A = best A' \in N(\sigma, A)
    return fail
```

Set of TA-graphs in which an action was inserted or removed to resolve the inconsistency

What is a better neighbor?
Better Neighbor?

- A neighbor $A' \in N(\sigma, A)$ resolves the inconsistency $\sigma$ by inserting or deleting an action.

- Use evaluation function $E(a)$

$$E(a) = \begin{cases} 
E(a)^i = \alpha \cdot \text{Execution\_cost}(a)^i \\
+ \beta \cdot \text{Temporal\_cost}(a)^i \\
+ \gamma \cdot \text{Search\_cost}(a)^i 
\end{cases}$$

$$E(a)^r = \alpha \cdot \text{Execution\_cost}(a)^r \\
+ \beta \cdot \text{Temporal\_cost}(a)^r \\
+ \gamma \cdot \text{Search\_cost}(a)^r$$

Relaxed Plan

- Idea: Don’t consider the mutex relation and perform reachability analysis.

- Insert action $a$
  - Find all actions that is required to support the preconditions of $a$
  - Compute the maximum time duration required for all actions
  - Return:
    - Set of actions added: $\text{Aset(EvalAdd}(a))$
    - Max time duration of the actions: $\text{End\_time(EvalAdd}(a))$

- Remove action $a$
  - Find all actions that is required to support all preconditions that were unsupported due to removal of $a$
  - Compute the maximum time duration required for all newly inserted actions
  - Return:
    - Set of actions added: $\text{Aset(EvalDel}(a))$
    - Max time duration of the actions: $\text{End\_time(EvalDel}(a))$
Better Neighbor

\[ \text{Execution\_cost\ (a)}^i = \sum_{a' \in \text{Aset(EvalAdd(a))}} \text{Cost}(a') \]
\[ \text{Temporal\_cost}(a)^i = \text{End\_time(EvalAdd(a))} \]
\[ \text{Search\_cost}(a)^i = |\text{Aset(EvalAdd(a))}| + \sum_{a' \in \text{Aset(EvalAdd(a))}} \text{Threats}(a') \]

\[ \text{Execution\_cost\ (a)}^r = \sum_{a' \in \text{Aset(EvalDel(a))}} \text{Cost}(a') \]
\[ \text{Temporal\_cost}(a)^r = \text{End\_time(EvalAdd(a))} \]
\[ \text{Search\_cost}(a)^r = |\text{Aset(EvalAdd(a))}| + \sum_{a' \in \text{Aset(EvalDel(a))}} \text{Threats}(a') \]

Advantages and Disadvantages

• Pros
  – One of the fastest domain-independent planners
  – Relatively expressive domain description languages
  – Can easily be extended to be anytime algorithm

• Cons
  – Algorithm is not guaranteed to be complete
  – No guarantee on the quality of the plan
  – Does not allow flexible time bounds