Directed Search for Generalized Plans Using Classical Planners

Siddharth Srivastava
University of Massachusetts
Amherst

(Joint work with Neil Immerman, Shlomo Zilberstein, and Tianjiao Zhang)
What is Generalized Planning?

- Given: a *domain, set of initial states* from potentially different state spaces, goal formula
- Find: A *generalized plan that solves* all of the initial states
Existing Approaches

- **Plan reuse** [Fikes et al. ’72, Hammond ’86]
- **Plans with loops**
  - Recognize repetitive patterns in plans [Hu and Levesque ’10, Winner and Veloso ’07, Levesque ’05]
  - Search for cyclic controllers for a partially observable instance [Bonet et al., ’09]
- **Our objectives:**
  - Guarantees of termination & correctness; directed generalization
  - Not addressed by these approaches
Framework
States

States = logical structures in a domain’s FO(TC) vocabulary

\[ V = \{ \text{on, onTable, clear, on}^+ \} \]

Define:
Role of an element = set of unary predicates it satisfies

\[ a = \{ \text{clear} \} \]
\[ b = \{} \]
\[ c = \{ \text{onTable} \} \]

- Transitive closure preserves some relationships
- Integrity constraints clarify the sets of states represented by abstract states
Action Application

- Branch depends on #elements (role-count) in the middle
- May need to draw out representative elements prior to action application
  - Such operations + changing the drawn element’s role = TM expressiveness (abacus programs)

[ICAPS 2010; AI Journal 2011]
Generalized Plans: Definition

- Connected, directed graph
- Nodes = actions
- Edges = conditions/ “abstract” states
  - Will use the dual representation in algorithms
- Start/Terminal nodes

A special class of finite state transducers
Recall: Generalized Planning Problem

• Given: a domain (predicate vocabulary + actions + integrity constraints), set of initial states (abstract state $S_o$), goal formula (FO(TC))

• Find: A generalized plan that solves every initial state represented by $S_o$

• “Solves” an initial state: Every possible execution reaches the goal in a finite number of steps

[AI Journal, 2011]
Generalized Plan Synthesis
Our Approach

- Get an example *unsolved* problem instance
- Get a classical plan for solving this instance
- Generalize and merge this plan
- Repeat
Hybrid Search: Idea

Objects $b_1$ and $b_2$ don’t exist in the general problem, or in the abstract state.

Start node (Initial abstract state)

Dual Representation:

Nodes = Abstract States
Edges = Actions
Hybrid Search: Idea

Goal State!

- *Some* states represented by the initial abstract state reach the goal along this path.
- Others end up at the “open” node

Classical Planner

Add choice actions

\[
\text{choose } x : \text{role}(x) = \text{role}(b_1) \\
\text{mvToTable}(x) \]

\[
\text{choose } x : \text{role}(x) = \text{role}(b_2) \\
\text{mvToTable}(x) \]

• Some states represented by the initial abstract state reach the goal along this path.  
• Others end up at the “open” node
Hybrid Search: Idea

Overall Algorithm:

- Repeat this process with new open nodes
Hybrid Search: Idea

Classical Planner

Add choice actions

<mvToTable(b1), mvToTable(b2)>

choose x: role(x) = role(b1)
mvToTable(x)
choose x: role(x) = role(b2)
ToTable(x)

Goal State!
Hybrid Search: Idea

Overall Algorithm:

- Repeat this process with new open nodes
- Merge an abstract state with an existing one that subsumes it:
  - Loops must be guaranteed to terminate: undecidable problem
- Use methods from [ICAPS 2010]
Generalized Plan Synthesis: Hybrid Approach

1. Get open node
2. Gen Plan w/ Branches and Loops
3. Merge
4. Trace
5. Linear Gen Plan
6. Unsolved Abstract State
7. Generate Instance
8. Classical Problem Instance
9. Classical Planner
10. Classical Plan
11. Add Choice Axns
12. a1(o1), a2(o2), a3...
13. ch(x:...), a1(x), ch(x:...), a2(x),...
14. New open node
Components of Hybrid Search: Instance Generation

Describe the abstract state using FO(TC) formulas capturing:
1. Each role present: e.g. $\exists x \text{ clear}(x) \&...$
2. Each singleton present: e.g. $\forall x, y \text{ clear}(x) \& \text{clear}(y) \rightarrow x=y \&...$
3. Each true predicate tuple: e.g. $\text{clear}(x) \& \text{onTable}(y) \rightarrow \text{on}^+(x,y) \&...$
4. Each false predicate tuple: e.g. $\text{clear}(x) \& \text{onTable}(y) \rightarrow \neg \text{on}(x,y) \&...$
5. Integrity constraints: e.g. $\forall x, y, z \text{ on}(x,z) \& \text{on}(y,z) \rightarrow x=y\&...$

$\frac{1}{2}$’s remain unspecified
Instance Generation

- First-order model generators are available, but our formulas use transitive closure (TC)
- TC cannot be expressed in first-order logic
- Simulate it using:

\[ \forall x, y \ p^{tc}(x, y) \leftrightarrow p(x, y) \lor \exists z \ (p(x, z) \land p^{tc}(z, y)) \]

This is accurate in finite, acyclic models

[Lev Ami et al., 2005, 2009]
Formal Guarantees

- Action application: sound
  - $\text{States-represented-by}[\ axn(\text{abstract state}) \ ] \supseteq \ axn(\text{states-represented-by}[\text{abstract state}])$

- If a plan’s execution terminates, it must do so at a state represented by an open node

- If plan terminates & all open nodes satisfy the goal condition: plan solves all instances of init. abs. state
Results
Results: Instance Generation

[Graph showing the relationship between the number of elements and instance generation time for different scenarios: Delivery, Blocks, Grid 3x, Grid 5x, Grid 7x.]
## Results

| Problem            | $N_{calls}$ | $||S||_{max}$ | $||\pi||_{max}$ | Applicability | T(s) | $N_{calls}$ | $||S||_{max}$ | $||\pi||_{max}$ | Applicability | T(s) |
|--------------------|-------------|---------------|-----------------|---------------|------|-------------|---------------|-----------------|---------------|------|
| Delivery           | 7           | 9             | 9               | T, C, P       | 297  | 7           | 14            | 25              | T, C, P       | 246 |
| Grid3x             | 6           | 15            | 12              | T, C, P       | 166  | 3           | 21            | 20              | T, C, P       | 82  |
| Grid5x             | 8           | 25            | 22              | T, C, P       | 631  | 4           | 35            | 34              | T, C, P       | 172 |
| Grid6x*            | 10          | 30            | 27              | T, C, P       | 1791 | 9           | 30            | 29              | T, C, P       | 1902 |
| Hall-A             | 7           | 10            | 6               | T, C, P       | 180  | 3           | 24            | 18              | T, C, P       | 70  |
| Reverse            | 7           | 8             | 6               | T, C, P       | 119  | 6           | 8             | 15              | T, C, P       | 128 |
| Sorting            | 7           | 7             | 6               | T, C, P       | 78   | 7           | 9             | 6               | T, C, P       | 82  |
| Striped Tower      | 10          | 10            | 11              | T, C, P       | 831  | 5           | 14            | 24              | T, C, P       | 290 |
| Y-Transport        | 9           | 13            | 53              | T, C, P       | 943  | 7           | 13            | 63              | T, C, P       | 550 |

- At most 10 planning problems required to generate generalized plans for unbounded instances
- Problem instances to be solved by classical planners are “small” and deterministic
- Unified approach for algorithm and generalized plan synthesis
Conclusions

• An approach for utilizing the powerful heuristic search capabilities of classical planners for:
  • Simultaneous exploration of infinitely many state spaces using abstraction
  • Generation of potentially useful paths of actions in the abstract state space

• Future Work:
  ▫ More robust termination tests
  ▫ Identification of unreachable/unsolvable open nodes
Thank you!
Sets of States as Abstract States

Define: **Role** of an element = set of unary predicates it satisfies

“Summary” elements .... Represent collections
Abstract States: Truth values

• Binary relationships with summary elements may become imprecise
• Represented using the truth value $\frac{1}{2}$, denoting “unknown”
• Integrity constraints clarify the sets of states represented by abstract states

\[\begin{array}{cccc}
on & 1 & 2 & 3 & 4 \\
1 & 0 & 1 & \frac{1}{2} & 0 \\
2 & 0 & 0 & 1 & \frac{1}{2} \\
3 & 0 & \frac{1}{2} & 0 & 1 \\
4 & 0 & 0 & 0 & 0 \\
\end{array}\]

\[\begin{array}{cccc}
on & a & b & c \\
a & 0 & \frac{1}{2} & 0 \\
b & 0 & \frac{1}{2} & \frac{1}{2} \\
c & 0 & 0 & 0 \\
\end{array}\]