Basics of AI and Machine Learning

Other Planning Formalisms

David Speck

Linköping University

Slides modified from Basel AI group, with permission

Automated Planning: Overview

Chapter overview: automated planning

- Introduction
- The STRIPS Planning Formalism
- Other Planning Formalisms
- Planning Heuristics
- Alternatives to Heuristic Search

Four Formalisms

Reminder: Four Planning Formalisms

- A description language for state spaces (planning tasks) is called a planning formalism.
- We introduce four planning formalisms:
  1. STRIPS (Stanford Research Institute Problem Solver)
  2. ADL (Action Description Language)
  3. SAS$^+$ (Simplified Action Structures)
  4. PDDL (Planning Domain Definition Language)

ADL, SAS$^+$ and PDDL

Basic Concepts of ADL

basic concepts of ADL:

- Like STRIPS, ADL uses propositional variables (true/false) as state variables.
- preconditions of actions and goal are arbitrary logic formulas (action applicable/goal reached in states that satisfy the formula)
- in addition to STRIPS effects, there are conditional effects: variable v is only set to true/false if a given logical formula is true in the current state

Basic Concepts of SAS$^+$

basic concepts of SAS$^+$:

- very similar to STRIPS: state variables not necessarily binary, but with given finite domain (cf. CSPs)
- states are assignments to these variables (cf. CSPs)

Basic Concepts of SAS$^+$

basic concepts of SAS$^+$:

- very similar to STRIPS: state variables not necessarily binary, but with given finite domain (cf. CSPs)
- states are assignments to these variables (cf. CSPs)
- preconditions and goals given as partial assignments

  example: $\{v_1 \mapsto a, v_3 \mapsto b\}$ as preconditions (or goals)
  - If $s(v_1) = a$ and $s(v_3) = b$,
    then the action is applicable in s (or goal is reached)
  - values of other variables do not matter

Basic Concepts of SAS$^+$

basic concepts of SAS$^+$:

- very similar to STRIPS: state variables not necessarily binary, but with given finite domain (cf. CSPs)
- states are assignments to these variables (cf. CSPs)
- preconditions and goals given as partial assignments

  example: $\{v_1 \mapsto a, v_3 \mapsto b\}$ as preconditions (or goals)
  - If $s(v_1) = a$ and $s(v_3) = b$, then the action is applicable in s (or goal is reached)
  - values of other variables do not matter
- effects are assignments to subset of variables

  example: effect $\{v_1 \mapsto b, v_2 \mapsto c\}$ means
  - In the successor state s', $s'(v_1) = b$ and $s'(v_2) = c$.
  - All other variables retain their values.

Basic Concept of PDDL

- PDDL is the standard language used in practice to describe planning tasks.
- descriptions in (restricted) predicate logic instead of propositional logic ($\rightsquigarrow$ even more compact)
- other features like numeric variables and derived variables (axioms) for defining "macros"
  (formulas that are automatically evaluated in every state and can, e.g., be used in preconditions)
- There exist defined PDDL fragments for STRIPS and ADL; many planners only support the STRIPS fragment.

Examples: see `http://editor.planning.domains/`

Summary

Summary

planning formalisms:

- STRIPS: particularly simple, easy to handle for algorithms
  - binary state variables
  - preconditions, add and delete effects, goals: sets of variables
- ADL: extension of STRIPS
  - logic formulas for complex preconditions and goals
  - conditional effects
- SAS$^+$: extension of STRIPS
  - state variables with arbitrary finite domains
- PDDL: input language used in practice
  - based on predicate logic (more compact than propositional logic)
  - only partly supported by most algorithms (e.g., STRIPS or ADL fragment)