Automated Action Planning

Classical Planning for Non-Classical Planning Formalisms

Carmel Domshlak

Automated Action Planning
— Classical Planning for Non-Classical Planning Formalisms

Beyond Classical Planning

Richer models people are working on

1. Temporal Planning (action have duration)
2. Metric Planning (continuous variables)
3. Planning with Preferences
4. Planning with Resource Constraints
5. Net-benefit Planning (maximize net value of goals achieved)
6. Generalized Planning (complex control structures, such as loops)
7. Multi-agent Planning
8. Planning Under Uncertainty:
   - 8.1 Conformant Planning
   - 8.2 Contingent Planning
   - 8.3 Markov Decision Processes (MDPs)
   - 8.4 Partially Observable MDPs
   - 8.5 Conformant Probabilistic Planning (Fully Unobservable POMDPs)

How many courses on planning do we need?

Key Insights:

- ☺ Classical planning offers a wealth of ideas for generating good solutions, fast.
- ☹ Importing these ideas to each of the above non-classical formalisms is difficult, and often simply does not work.

Yet:

- ☺ Goal oriented sequencing of actions is a fundamental computational problem at the heart of all planning problems.
- ☺ Classical planners have reached a certain performance level that makes them attractive for addressing this problem.

So...

Two Strategies

1. Top-down:
   Develop native solvers for more general class of models
   - $+$: generality
   - $-$: complexity
2. Bottom-up: Extend the scope of 'classical' solvers
   - $+$: efficiency
   - $-$: generality

We now explore the second approach

Using Classical Planners within Non-Classical Planners

Two Key Techniques:

1. Replanning: the classical problem is an optimistic view of the original problem
2. Compilation: the classical problem is equivalent to the original problem
   (possibly under certain reasonable conditions)

Replanning

An online method for solving planning problems with some uncertainty

1. Make some assumptions → get a simpler model
2. Solve simpler model
3. Execute until your observation contradict your assumptions
4. Repeat (Replan)

An established technique:

- Underlies many closed loop controllers
- Used in motion planning under uncertainty

Motivation: Why Analyzing the Expressive Power?

- Expressive power is the motivation for designing new planning languages

⇝ Often there is the question: *Syntactic sugar* or *essential feature*?

- *Compiling away* or change planning algorithm?
- If a feature can be compiled away, then it is apparently only *syntactic sugar*.
- However, a compilation can lead to much larger planning domain descriptions or to much longer plans.

⇝ This means the planning algorithm will probably choke, i.e., it cannot be considered as a compilation

Example: DNF Preconditions

- Assume we have **DNF preconditions** in STRIPS operators
- This can be **compiled away** as follows
- Split each operator with a DNF precondition $c_1 \vee \ldots \vee c_n$ into n operators with the same effects and c_i as preconditions

$\leadsto$ If there exists a plan for the original planning task there is one for the new planning task and *vice versa*

$\longrightarrow$ The planning task has almost the same size

$\longrightarrow$ The shortest plans have the same size

Example: Conditional effects

- Can we compile away **conditional effects** to STRIPS?
- Example operator: $\langle a, b \rhd d \wedge \neg c \rhd e \rangle$
- Can be translated into four operators:
  $\langle a \wedge b \wedge c, d \rangle, \langle a \wedge b \wedge \neg c, d \wedge e \rangle, \ldots$
- Plan existence and plan size are identical
- Exponential blowup of domain description!

$\longrightarrow$ Can this be avoided?