Generalized Planning: Non-Deterministic Abstractions and Trajectory Constraints

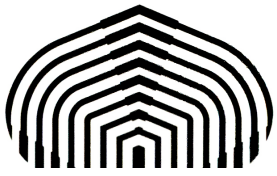
B. Bonet[1] G. De Giacomo[2] H. Geffner[3] S. Rubin[4]

[1] Universidad Simón Bolívar, Venezuela

[2] Sapienza Università di Roma, Italy

[3] ICREA & Universitat Pompeu Fabra, Spain

[4] Università degli Studi di Napoli Federico II, Italy



SAPIENZA
UNIVERSITÀ DI ROMA

upf. Universitat Pompeu Fabra Barcelona

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

Generalized Planning: Example

Want policy that works for many (possibly ∞) problem instances

Example: Problem Counter-n:

- Counter problem with **single variable** X with initial value $X = n$
- Agent **senses** whether $X = 0$ or $X > 0$
- Agent can **increase** or **decrease** value of X
- **Observable goal** is to reach $X = 0$

Policy: "if $X > 0$, decrease X" works:

- for any $n \geq 0$
- problems with more than one possible initial state
- even if actions may fail sometimes; e.g. decrease don't work sometimes

(Srivastava et al. 2008, 2011; Hu & Levesque 2010; Hu & De Giacomo 2011; B. & Geffner 2015; Belle & Levesque 2016; etc)

Generalized Planning: Formulation

- In Hu & De Giacomo (2011) formulation, collection $\mathcal{P}$ of instances assumed to share **common pool of observations and actions**
- Policy μ mapping observations into actions said to **generalize** to $\mathcal{P}$ if it solves **all problems** in $\mathcal{P}$
- **General finite-state controllers** can be defined in same way

Generalized Planning: Computation

Top-down appproach:

– If $\mathcal{P}$ is finite, **compile** $\mathcal{P}$ into a regular planning problem or do **search** in controller space (Hu & De Giacomo 2011)

– If $\mathcal{P}$ is infinite, finite subset of $\mathcal{P}$ sometimes ensures generalization to $\mathcal{P}$ (e.g. 1D problems; Hu & Levesque 2010)

Bottom-up approach:

– Solve **single** "representative instance" P of $\mathcal{P}$ and **prove** that solution ensures generalization (B. et al. 2009)

– **Example:** solution to Counter problem with **two** possible initial states **generalizes** to class $\mathcal{P} = \{\text{Counter-}n : n \geq 0\}$

Goal for this paper

Key question in **bottom-up approach:**

– What's the **common structure** between **single problem** P and class $\mathcal{P}$ that yields the generalization?

Question partially answered in earlier work:

Theorem (B. & Geffner 2015)

If P **reduces** *to* P' *and* μ *is* **strong cyclic solution** *for* P'*, then* μ **solves** P *if it* **terminates** *in* P *over fair trajectories*

In this work, we:

– analyze necessity of **termination** in B. & Geffner (2015) formulation

– show how to **get rid** of termination condition

Outline

- Basic framework
- Observation projections abstractions
- Trajectory constraints
- New generalization theorems
- Generalized planning as LTL Synthesis
- Generalized planning over QNPs as FOND planning
- Wrap up

PONDPs and Classes

Partially obs. non-det. problem $P = (S, I, \Omega, Act, T, A, obs, F)$:

- S is state space (finite or infinite)
- $I \subseteq S$ is set of initial states
- Ω is set of observations
- Act is set of actions
- $T \subseteq S$ is set of goal states
- $A : S \to 2^{Act}$ is available-actions function
- $obs : S \to \Omega$ is **observation function**
- $F : Act \times S \to 2^S \setminus \{\emptyset\}$ is **non-deterministic transition function**

Class $\mathcal{P}$ of PONDPs with **observable goals** and **action preconditions,** and where all problems share common:

- set of actions Act
- set of observations Ω
- **subset T_Ω of goal observations;** $\forall P \, \forall s : s \in T_P$ **iff** $obs_P(s) \in T_\Omega$
- **subsets A_ω of actions:** $\forall P \, \forall s : A_P(s) = A_{obs(s)}$

Standard Solution Concepts

Policy is function $\mu : \Omega^+ \to Act$

Policy μ is **valid** for problem P if it selects **applicable actions**

Let P be a problem and μ be a valid policy for P:

– μ is **(strong) solution** for P iff every μ-trajectory is goal reaching

– μ is **fair solution** or **strong cyclic solution** for P iff every **fair** μ-trajectory is goal reaching

Henceforth, we focus on valid policies

Abstractions: Observation Projection

Project **entire class** $\mathcal{P}$ into single **non-deterministic** problem P^o:

- **state space:** $S^o = \Omega$
- **initial states:** $\omega \in I^o$ iff $obs_P(s) = \omega$ for some P and $s \in I_P$
- **actions:** $Act^o = Act$ and $A^o(\omega) = A_\omega$
- **goal states:** $T^o = T_\Omega$
- **transitions:** $\omega' \in F^o(a, \omega)$ iff $s' \in F_P(a, s)$ for some problem P in $\mathcal{P}$, and states s and s' with $a \in A_P(s)$, $obs_P(s) = \omega$ and $obs_P(s') = \omega'$

Example: For class of Counter-n problems, P^o features:

- **2 states** (observations): $[X = 0]$ and $[X > 0]$
- **non-deterministic** transitions; e.g. $[X > 0]$ transitions under decrease action to both $[X = 0]$ and $[X > 0]$

Need for More Structure

Policy $\mu =$ "if $X > 0$, decrement X" solves all Counter-n problems but doesn't solve projection P^o

P^o is **non-deterministic** and μ may get trapped into loop where Decrement X doesn't work

Projection P^o misses important **structural property** that all Counter-n problems share but that is lost projection:

> *If variable X is decreased infinitely often and increased only a finite number of times, it eventually reaches $X = 0$*

In this work we extend the model to make such properties **explicit**

Trajectory Constraints

Trajectory constraint C over P is subset of **infinite state-action sequences** (i.e. $C \subseteq (S \times Act)^\infty$) or subset of **infinite observation-action sequences** (i.e. $C \subseteq (\Omega \times Act)^\infty$)

Trajectory τ **satisfies** C if τ is finite, or either $\tau \in C$ (if $C \subseteq (S \times Act)^\infty$), or $obs(\tau)$ in C (if $C \subseteq (\Omega \times Act)^\infty$) where

$$obs(\langle s_0, a_0, s_1, a_1, \ldots \rangle) \;=\; \langle obs(s_0), a_0, obs(s_1), a_1, \ldots \rangle$$

- Problem P extended with constraint C is denoted by P/C
- Problem P **satisfies constraint** C if all trajectories in P satisfy C

New solution concept: μ solves P/C iff every μ-trajectory τ that satisfies C is goal reaching

Example: $C = \{\tau : \tau$ is infinite and satisfies the crucial property$\}$ for $\mathcal{P}$

New Generalization Theorems

Theorem (Generalization)

Let $\mathcal{P}$ be a class of FONDP and C a constraint such that every P in $\mathcal{P}$ satisfies C. Then, μ **solves all problems** *in $\mathcal{P}$ if μ solves P^o/C*

Example: trajectories in P^o that satisfy C happen to be fair. Thus, μ must be fair solution (P^o has no strong solution by non-determinism). Theorem asserts μ solves all instances in which decrease action **satisfies constraint**

Theorem (Completeness)

If P^o is obs. projection for class $\mathcal{P}$ and μ solves **all problems** *in $\mathcal{P}$, there is constraint C over P^o such that every P in $\mathcal{P}$ satisfies C and μ solves P^o/C*

Generalized Planning as LTL Synthesis

When trajectory constraints can be expressed in LTL (over language $\Sigma = Act \cup \Omega$), LTL techniques can be used to obtain general plans

Theorem

Let P^o/C be obs. projection with constraint C expressed in LTL as Ψ. Then, solving P^o/C (and hence all P/C for $P \in \mathcal{P}$) is **2EXPTIME-complete;** *it's* **double-exponential** *in $|\Psi| + |T^o|$ and* **polynomial** *in $|P^o|$*

Sketch: Idea is to think of policies μ as Ω-**branching** Act-**labeled** graph:

- Build **tree-automaton** accepting policies μ such that every μ-trajectory satisfies formula $\Phi = \Psi \supset \Diamond T^o$ where $\Diamond T^o$ is **reachability goal** in $\mathcal{P}$
- Check **non-emptiness** of language accepted by tree-automaton; this test yields witness (i.e. policy) if it exists

Generalized Planning over QNPs as FOND Planning

Qualitative Numerical Planning

- Problem R_V with set V of non-negative numeric variables (don't have to be integer variables) and standard Boolean propositions
- Actions can affect propositions and also increase or decrease value of numeric variables **non-deterministically**
- Propositions are fully observable while only $X = 0$ and $X > 0$ can be observed for each var X
- Paper describes syntax for specifying class of QNPs sharing same set of vars, fluents, actions, observations, ...

Example: General problem of stacking a block x on a block y in instance with any number of blocks can be cast as QNP

Abstractions for some QNPs appear in (Srivastava et al., 2011, 2015)

Solving QNPs with FOND Planners

Given QNP R_V, obs. projection R_V^o constructed **syntactically:**

- Projection contains only propositions and **no numeric variables**
- For each variable X, there are propositions $X > 0$ and $X = 0$
- Each effect $Inc(X)$ replaced by atom $X > 0$, and effect $Dec(X)$ replaced by non-det. effect $X > 0 \,|\, X = 0$

Non-determinism in P^o isn't **fair** (Srivastava et al. 2011); i.e. strong cyclic plan for R_V^o isn't guaranteed to be solution

Projection R_V^o is **modified** to target interesting subclasses of QNPs:

Theorem (Soundness and Completeness)

Let R_V be QNP such that a) actions with $Dec(X)$ effects have prec. $X > 0$, and b) actions have decrement effects for **at most one** *variable. μ is fair solution to* **modified** R_V^o **iff** μ *solves* **all problems in class** *defined by R_V*

Related Work

- QNPs related to problems considered by (Srivastava et al. 2011, 2015)
- 1D problems (Hu & Levesque 2010; Hu & De Giacomo 2011) is infinite class of "identical" problems characterized by single integer parameter
- Hu & De Giacomo (2011) construct a single "large enough" abstraction whose solution provides a solution to the class
- Sardiña et al. (2006) also analyze tasks in which "global properties" are lost in observation projection; we recover such properties with constraints
- De Giacomo et al. (2016) show that trace constraints are necessary for belief construction to work on infinite domains

Summary

- Bottom-up approach for generalized planning where general policies are obtained from **solutions of single instances**
- Non-deterministic abstraction P^o extended with **trajectory constraints** avoid need for checking termination for solutions
- Solutions to class $\mathcal{P}$ of problems that satisfy constraint C obtained from solutions to P^o/C
- P^o/C can be solved using LTL (if constraints are LTL-expressible) or, in some cases, using more efficient FOND planners

Discussion

- There are many constraints that are satisfied by given target class of instances; Which constraints to make explicit?
- Can we automate the discovery of relevant constraints?
- Extend scope of QNPs that can be solved using FOND planners; General results?
- Analyze and test LTL synthesis for specific and relevant types of problems/constraints; Can existing LTL synthesis techniques be **effectively** used to solve interesting generalized planning tasks?