Generalized Planning with Loops under Strong Fairness Constraints

Giuseppe De Giacomo\textsuperscript{1}  Fabio Patrizi\textsuperscript{1}  Sebastian Sardiña\textsuperscript{2}

\textsuperscript{1}Dipartimento di Informatica e Sistemistica
SAPIENZA Università di Roma
Rome, Italy
degiacomo@dis.uniroma1.it
patrizi@dis.uniroma1.it

\textsuperscript{2}School of Computer Science and IT
RMIT University
Melbourne, Australia
sebastian.sardina@rmit.edu.au

KR 2010 - Toronto, May 12, 2010
Nondeterministic Planning Domains

Nondeterministic Planning Domain $\mathcal{D}$:

- A finite set $P$ of propositions –whose subsets are called states–, capturing all domain’s relevant features
- A finite set $A$ of actions, to be executed in the domain
- A transition relation $\rightarrow \subseteq 2^P \times A \times 2^P$, representing the (possibly non-deterministic) domain dynamics, subject to action executions

Example (Coin Tossing)

- $P = \{\text{head}, \text{tail}, \text{holding}\}$, $A = \{\text{grab}, \text{toss}, \text{turn}\}$
- $\rho = \{\emptyset \xrightarrow{\text{grab}} \{\text{holding}\}, \{\text{holding}\} \xrightarrow{\text{toss}} \{\text{head}\}, \{\text{holding}\} \xrightarrow{\text{toss}} \{\text{tail}\}, \{\text{head}\} \xrightarrow{\text{turn}} \{\text{tail}\}, \{\text{tail}\} \xrightarrow{\text{turn}} \{\text{head}\}, \{\text{tail}\} \xrightarrow{\text{grab}} \{\text{holding}\}, \{\text{head}\} \xrightarrow{\text{grab}} \{\text{holding}\}\}$
Conditional Planning Problems

Conditional Planning Under Full Observability

(For now, w/o loops)

- **INPUT:**
  - a nondeterministic domain $\mathcal{D} = \langle P, A, \rho \rangle$
  - an initial state $S_0 \subseteq P$
  - a propositional goal formula $\gamma$ over $P$

- **SOLUTION:** a *conditional* plan $\pi$ s.t. all executions achieve $\gamma$

- **COMPLEXITY:** EXPTIME-complete (also w/loops)

Example

$\langle S_0 = \emptyset, \gamma = \text{head} \rangle$ on *Coin Tossing* solved by plan:

1. *grab*
2. *toss*
3. *if*($\neg\text{head}$) *then* *turn*
Conditional Planning with Loops

Loops allowed in plans

Example

\[ \pi = \text{while } (\neg \text{head}) \{ \text{grab; toss} \} \]

- Clearly, \( \pi \) does not solve \( \langle S_0 = \emptyset, \gamma = \text{head} \rangle \)...
- ...however, in the real world, everyone would bet it eventually does!
- We want to assert non-local constraints!
Conditional Planning with Loops (2)

- Previous work on *Strong Cyclic Planning* [CPRT03] assumes *fair* action executions:
  - All action effects eventually occur
  - Cannot distinguish between fair and unfair action executions (either all or none!)
  - Thus, cannot make decisions based on this

**In this work**

- We *explicitly* characterize *relevant runs*, through *constraints*
- Capture two different flavours of nondeterminism:
  - **Uncertainty**: the effect will occur, but don’t know exactly when (e.g., rolling a die)
  - **Partial Knowledge**: may or may not occur (e.g.: picking cards from a deck with a possibly missing Ace)
Constraints on Runs

**Runs**: possible evolutions of a domain, generated by executing plans

**Example (π executions)**

Constraints: LTL formulas built from propositions in \( P \cup \bigcup_{a \in A} \{ (act = a) \} \)

**Example (Constraints on D runs)**

Tossing a coin infinitely often yields *head* to occur infinitely often

\[ \Box \Diamond (act = toss) \rightarrow \Box \Diamond (head) \]
Constraints on Runs (2)

- **Constraints**: LTL formulas to be evaluated on domain runs
- We use run constraints to *rule out* irrelevant runs
- Only runs satisfying *all* constraints are significant

Semantics of constraints on domain runs

Given:
- a planning domain \( D \) with a finite set \( C \) of constraints on runs
- initial state \( S_0 \) and a goal formula \( \gamma \)

A conditional plan \( \pi \) with loops achieves \( \gamma \) (\( \pi \models \top \mathcal{U} \gamma \)) if all of its executions *satisfying* all \( C \) constraints reach a state \( S \) s.t. \( S \models \gamma \)

- LTL is very natural: Conditional Planning focuses on single executions (run-by-run)
Constraints on Runs (3)

We use run constraints to assert non-local domain properties

Example (More realistic Coin Tossing domain)

If we assert the following constraints on Coin Tossing:

- $\Box \Diamond (act = toss) \rightarrow \Box \Diamond (head)$
- $\Box \Diamond (act = toss) \rightarrow \Box \Diamond (tail)$

Then plan $\pi = \text{while } (\neg head) \{\text{grab; toss}\}$ solves $\langle S_0 = \emptyset, \gamma = head \rangle$

Indeed, the only unsuccessful $\pi$ execution

\[ \emptyset \xrightarrow{\text{grab}} \{\text{holding}\} \xrightarrow{\text{toss}} \{\text{tail}\} \xrightarrow{\text{grab}} \{\text{holding}\} \xrightarrow{\text{toss}} \{\text{tail}\} \xrightarrow{\text{grab}} \cdot \cdot \cdot \]

is discarded by first constraint above
Strong Fairness Constraints

**Strong Fairness**

*If something \( \phi_s \) happens infinitely often, then something else \( \phi_e \) happens infinitely often*

\[ \Box \Diamond \phi_s \rightarrow \Box \Diamond \phi_e \]

(\( \phi_s \) and \( \phi_e \) essentially propositional, \( \bigcirc \) (Next) allowed)

- Strong Fairness captures also:
  - Weak fairness (something \( \phi \) happens infinitely often): \( \Box \Diamond \top \rightarrow \Box \Diamond \phi \)
  - Persistence (something \( \phi \) holds from a point on): \( \Box \Diamond \neg \phi \rightarrow \Box \Diamond \bot \)

- Not expressible in CTL [CGP99]

- Can phrase typical properties of our interest
- Good computational behavior (wrt Conditional Planning, see below)
- Existing results on LTL synthesis [PPS06, KPP05] apply
Synthesis in LTL

LTL Synthesis Problem:

System $\Phi_S(\mathcal{X}, \mathcal{Y})$\hspace{1cm}Controller $\Phi_C(\mathcal{X}, \mathcal{Y})$

- Uncontrolled ($\mathcal{X} = \{x_1, \ldots, x_n\}$) and controlled ($\mathcal{Y} = \{y_1, \ldots, y_m\}$) vars
- **System** assigns $\mathcal{X}$ vars (moves first), **Controller** assigns $\mathcal{Y}$ vars
- Both have their own *structural assumptions* (constraints on execution)

**Objective:**

- Find a controller strategy satisfying an LTL specification $\varphi$
  (Technically, $\varphi$ combines $\Phi_C$, $\Phi_S$, and desired requirements.
  In particular, $\varphi$ requires the strategy to meet $\Phi_C$ when interacting
  with $\Phi_S$)
Synthesis in LTL (2)

Complexity:
- For arbitrary $\varphi$, the problem is 2EXPTIME-complete [PR89]
- GR(1) formulas yield an EXPTIME bound [PPS06, KPP05]

Generalized Reactivity (1) Specifications:

$$\varphi = \varphi_a \rightarrow \varphi_r$$

- $\varphi_a$: System structural assumptions + possible (weak) fairness constraints
- $\varphi_r$: Controller structural assumptions + possible (weak) fairness constraints
- Express (desired) requirements of the form

$$\bigwedge_{n} \Box \Diamond \phi_i \rightarrow \bigwedge_{m} \Box \Diamond \psi_j$$

- Expressive enough for our problem!
Conditional Planning w/ Loops under SFC as LTL Synthesis

\[ \varphi = \varphi_a \rightarrow \varphi_r \]

In our case:

1. \( \varphi_a = \varphi_a^{\text{init}} \land \varphi_a^{\text{trans}} \land \varphi_a^{\text{rc}} \), where:
   - \( \varphi_a^{\text{init}} \): initial condition (\( \mathcal{D} \) state)
   - \( \varphi_a^{\text{trans}} \): \( \mathcal{D} \) transitions and goal achievement
   - \( \varphi_a^{\text{rc}} \): constraints on \( \mathcal{D} \) runs (phrased as weak fairness)

2. \( \varphi_r = \varphi_r^{\text{trans}} \land \varphi_r^{\text{goal}} \), where:
   - \( \varphi_r^{\text{trans}} \): one executable action at each point (plan structure)
   - \( \varphi_r^{\text{goal}} \): achieve desired goal \( \gamma \) (phrased as weak fairness)
Main Results

**Theorem (Soundness & Completeness)**
Conditional Planning w/ Loops under Strong Fairness Constraints can be reduced to LTL synthesis for GR(1) formulas

**Theorem (Complexity)**
Building a conditional plan with loops under strong fairness constraints is EXPTIME-complete

Same complexity class as Conditional Planning w/ Full Observability!

**Implementation**
- Actual system available: TLV
- Based on (global) Model Checking techniques
Other Results
(See paper)

In general, approach shown effective for:

1. Goals of the form $\varphi = \psi U \phi$ (achieve $\phi$ while maintaining $\psi$)
2. Planning Programs [DPS10], whose atomic instructions are goals $\phi U \psi$, can be captured and realized
3. Component-based Planning: actions offered by (finite-state) devices, possibly subject to strong fairness constraints
Conclusions and Future Directions

1. Conditional Planning w/ loops, with non-local constraints explicitly asserted
2. More general but not computationally harder than Conditional Planning w/ out Loops
3. Tackled as an LTL synthesis problem, actual system available
4. Suitable for extended scenarios (Planning Programs, Component-based Planning)

1. Performance evaluation to be carried out
2. Plan quality: e.g., avoid loops when not required
3. Planning-oriented techniques and heuristics
References

Edmund M. Clarke, Orna Grumberg, and Doron A. Peled.
Model checking.

A. Cimatti, M. Pistore, M. Roveri, and P. Traverso.
Weak, strong, and strong cyclic planning via symbolic model checking.

Giuseppe De Giacomo, Fabio Patrizi, and Sebastian Sardina.
Agent programming via planning programs.
To appear.

Yonit Kesten, Nir Piterman, and Amir Pnueli.
Bridging the gap between fair simulation and trace inclusion.

Nir Piterman, Amir Pnueli, and Yaniv Sa’ar.
Synthesis of reactive(1) designs.

Amir Pnueli and Roni Rosner.