On Domain-Independent Heuristics for Planning with Qualitative Preferences

Jorge Baier    Sheila McIlraith

January 8th, 2007
## Motivation

### Classical Planning
- Plan must satisfy (final-state) goals.

### Planning with Qualitative Temporally Extended Preferences (QTEPs)
- Qualitative language to specify *preferred* plans.
  - E.g., Plans such that: `eventually(eat(tandooriChicken))` are preferred to those such that: `eventually(eat(spaghetti))`.
- Language allows temporally extended properties.
- We want a *most-preferred plan for the goal*. 
Motivation

Classical Planning

- Plan must satisfy (final-state) goals.
- Fastest state-of-the-art planners use lookahead heuristics.

Planning with Qualitative Temporally Extended Preferences (QTEPs)

- Qualitative language to specify *preferred* plans.
  E.g., Plans such that:  \textbf{eventually}(\textit{eat(tandooriChicken)})  
  are preferred to 
  those such that:  \textbf{eventually}(\textit{eat(spaghetti)}).

- Language allows temporally extended properties.
- We want a **most-preferred plan for the goal**.
Motivation

Classical Planning

- Plan must satisfy (final-state) goals.
- Fastest state-of-the-art planners use lookahead heuristics.

Planning with Qualitative Temporally Extended Preferences (QTEPs)

- Qualitative language to specify preferred plans.
  E.g., Plans such that: \( \text{eventually}(\text{eat(tandooriChicken)}) \) are preferred to those such that: \( \text{eventually}(\text{eat(spaghetti)}) \).
- Language allows temporally extended properties.
- We want a most-preferred plan for the goal.
- Current planners for qualitative preferences don’t use lookahead heuristics.

We propose a heuristic planner for QTEPs
Outline of the talk

- Background
  - LPP and planning
- Problem Simplification
- Heuristics for QTEP planning
- Algorithm
- Implementation of HPLAN-QP & Experimental Results
- Conclusions
Outline of the talk

- Background
  - LPP and planning
- Problem Simplification
- Heuristics for QTEP planning
- Algorithm
- Implementation of HPLAN-QP & Experimental Results
- Conclusions
We consider LPP [Bienvenu et al., 2006], a rich language for temporally extended preferences.

Main element APFs:

Atomic Preference Formulae (APFs)

Used to express preferences over alternative properties. Form:

\[ \varphi_0[v_0] \gg \varphi_1[v_1] \gg \ldots \gg \varphi_n[v_n], \]

where \( v_1 < v_2 < \cdots < v_n \in \mathcal{V} \), and \( \mathcal{V} \) is a totally ordered qualitative finite set, and \( \varphi_i \) is an formulae of a linear temporal logic (LTL).
Examples of APFs

Let $\mathcal{V} = \{\text{best, great, good, ok, bad}\}$. Examples of APFs:

\[
P_{food} \overset{\text{def}}{=} \text{eventually}(\text{occ}(\text{eat}(\text{pizza}))[\text{best}] \gg \\
\text{eventually}(\text{occ}(\text{eat}(\text{spag}))[\text{great}] \gg \\
\text{eventually}(\text{occ}(\text{eat}(\text{crêpes}))[\text{good}] \gg \\
\text{eventually}(\text{occ}(\text{eat}(\text{taoChicken}))[\text{ok}]}
\]

\[
P_{home} \overset{\text{def}}{=} \text{always}(\text{at}(\text{home})) \land \forall x \neg \text{eventually}(\text{occ}(\text{cook}(x))[\text{best}] \gg \\
\text{always}(\text{at}(\text{home})) \land \exists x \text{ eventually}(\text{occ}(\text{cook}(x))[\text{good}]}
\]
LPP allows combining preferences through *general preference formulae* (GPFs).

If $\gamma$ is an LTL formula, and $\Psi_1$ and $\Psi_2$ are APFs:

<table>
<thead>
<tr>
<th>GPF</th>
<th>Informal semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma : \Psi_1,$</td>
<td>If $\gamma$ holds in the plan, preferences given by $\Psi_1$</td>
</tr>
<tr>
<td>$\Psi_1 &amp; \Psi_2$</td>
<td>Prefer to satisfy both $\Psi_1$ and $\Psi_2$</td>
</tr>
<tr>
<td>$\Psi_1 \mid \Psi_2$</td>
<td>Indifferent between $\Psi_1$ and $\Psi_2$</td>
</tr>
</tbody>
</table>

Examples:

\[
P_{\text{home}} \& P_{\text{food}} \quad \text{IsSnowing} : P_{\text{home}}
\]
The semantics of LPP are defined in the situation calculus [Bienvenu et al., 2006].

The $w$ function is such that if $s_1$ and $s_2$ are situations and $\Psi$ is an GPF,

$$w_{s_1}(\Psi) < w_{s_2}(\Psi) \text{ iff } s_1 \text{ is preferred to } s_2 \text{ with respect to } \Psi.$$
<table>
<thead>
<tr>
<th>Definition (Classical Planning)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given a Situation Calculus theory of action ( D ) and a goal formula ( G ), find a situation ( S ) such that: ( D \models G(S) )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Definition (Preference-Based Planning)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Given a Situation Calculus theory of action ( D ), a goal formula ( G ), and a GPF ( \Psi ) find an ( S ) such that: ( D \models G(S) \land \neg \exists s' [G(s') \land w_{s'}(\Psi) &lt; w_S(\Psi)] )</td>
</tr>
</tbody>
</table>
State of the art

Best Classical Planners:
- Use some form search
- Guided by heuristics measuring progress towards achieving the goal.

Planners for QTEPs:
- Use some form search
- Guiding not based on progress towards achieving preferences
State of the art

Best Classical Planners:
- Use some form search
- *Guided* by *heuristics* measuring *progress* towards achieving the goal.

Planners for QTEPs:
- Use some form search
- Guiding *not based on progress* towards achieving preferences

**Our goal: apply heuristics for efficient QTEP planning**
The Challenge

Efficient classical planners use **heuristics**.
- Designed for single goals
- Designed for final-state goals

In planning with LPP preferences:
- GPFs composed by several properties, interacting in complex ways.
- Properties are temporal.

We need to solve **two problems**:
- **Identify the properties** that characterize preferred plans
- Guide search with a **single heuristic function**
Outline of the talk

- Background
  - LPP and planning
- Problem Simplification
- Heuristics for QTEP planning
- Algorithm
- Implementation of $\texttt{HPLAN-QP}$ & Experimental Results
- Conclusions
We simplify the planning problem, generating a new one such that:

- All GPFs are replaced by APFs ⇒ **reduce interaction** among BDFs.
- **Replace temporal** prefs. by equivalent **non-temporal** prefs.
We prove that:

**Theorem**

Let $\Psi$ be an arbitrary GPF over the set of preference values $\mathcal{V}$, then it is possible to construct an equivalent APF $\phi_\Psi$, over $\mathcal{V}$.

This means that all our preferences look like:

$$\phi_0[v_0] \gg \phi_1[v_1] \gg \ldots \gg \phi_n[v_n],$$

However, still the $\phi_i$’s is temporal.
Simplifying temporal formulae

In previous work [Baier and McIlraith, 2006], we proved that:

**Theorem**

Let $P$ be a planning problem, and $\varphi$ be a first-order LTL formulae. $P$ can be extended with a new additional predicate, $\text{Sat}_{\varphi}$, that is true in the final state iff $\varphi_i$ is true.

This means that now our preferences now look like:

$$\varphi_0[v_0] \gg \varphi_1[v_1] \gg \ldots \gg \varphi_n[v_n],$$

Where the $\varphi_i$’s are all non-temporal.
Outline of the talk

- Background
  - LPP and planning
- Problem Simplification
- Heuristics for QTEP planning
- Algorithm
- Implementation of HPLAN-QP & Experimental Results
- Conclusions
We always want to achieve our goal
We always want to achieve our goal

Goal distance \((G)\)

A distance-to-the-goals function computed from the expanded relaxed graph. In our implementation, is the additive heuristic by [Bonet and Geffner, 2001] adapted for ADL operators.
Heuristic functions for guiding search

We always want to achieve our goal

Goal distance \( (G) \)

A distance-to-the-goals function computed from the expanded relaxed graph. In our implementation, is the additive heuristic by [Bonet and Geffner, 2001] adapted for ADL operators.

Guide search towards preferred properties
Heuristic functions for guiding search

We always want to achieve our goal

Goal distance ($G$)
A distance-to-the-goals function computed from the expanded relaxed graph. In our implementation, is the additive heuristic by [Bonet and Geffner, 2001] adapted for ADL operators.

Guide search towards preferred properties

Preference distance function ($P$)
A distance-to-the-preferences function computed from the expanded relaxed graph. If the preference is

$$
\varphi_0[v_0] \gg \varphi_1[v_1] \gg \ldots \gg \varphi_n[v_n],
$$

Then $P = (p_0, \ldots, p_n)$, where $p_i$ is estimates how hard it is to achieve $\varphi_i$. 
Heuristics for pruning

if found plan with weight $W$, don't extend plans that won't reach a better weight

Best Relaxed Metric ($B$)

- An *estimation* of the best metric weight that plan that traverses the current state can achieve
- Corresponds to the best weight in the relaxed worlds.
Putting pieces together: adding the goal

Still unanswered: **When is** $s_1$ **better than** $s_2$?

Let $G_1$ and $G_2$ be the value of the goal distance function for $s_1$ and $s_2$.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Check whether</th>
<th>If tied, check whether</th>
</tr>
</thead>
<tbody>
<tr>
<td>goal-value</td>
<td>$G_1 &lt; G_2$</td>
<td>Is $s_1$’s best weighted preferred property easier than that of $s_2$?</td>
</tr>
<tr>
<td>goal-easy</td>
<td>$G_1 &lt; G_2$</td>
<td>Is $s_1$’s easiest preferred property easier than that of $s_2$?</td>
</tr>
</tbody>
</table>

**value-goal** and **easy-goal** do the tests in reverse order.
Outline of the talk

- Background
  - LPP and planning
- Problem Simplification
- Heuristics for QTEP planning
- Algorithm
- Implementation of \texttt{HPLAN-QP} & Experimental Results
- Conclusions
The Algorithm

**Input:** goal; APF preference; a bound for the plan \( k \)

**Output:** sequence plans for goal with incrementally better weight

Perform **best-first** search, where:

- States are ordered using one of the strategies proposed.
The Algorithm

**Input:** goal; APF preference; a bound for the plan $k$

**Output:** sequence plans for goal with incrementally better weight

Perform **best-first** search, where:

- States are ordered using one of the strategies proposed.
- If best plan found has weight $W$, then prune states whose $B$ function value is worse than $W$.
- Prune plans whose length exceed $k$.
- Output a plan when its weight is the best found so far.
- Execute until the search space is empty.

This is a **heuristic, incremental** planner for QTEPs.
Properties

**Definition (k-optimal)**
A planning algorithm is $k$-optimal, if it eventually returns the best-weighted plan among all those of length bounded by $k$.

**Theorem**
*Our proposed algorithm is k-optimal.*

**Observation**
This theorem does not mean that the *first* plan that is output...
Implementation of **HPLAN-QP**

- **Preprocessor:**
  - Parses a domain with atomic preferences (in an extended PDDL3!)
  - Performs the temporal simplification.
  - Generates TLPlan files.

- **Modified TLPlan:**
  - Compute heuristic estimates using relaxed graphs
  - Handle efficiently the automata updates.
We compared our planner to the PPLAN planner [Bienvenu et al., 2006].

Characteristics of PPLAN:
- Best-first search, admissible heuristics.
- $k$-optimal; first plan is optimal.
- Not optimized for speed
We compared our planner to the PPLAN planner [Bienvenu et al., 2006].

Characteristics of PPLAN:
- Best-first search, admissible heuristics.
- $k$-optimal; first plan is optimal.
- Not optimized for speed

Examples performed over a dinner domain.
Table: Number of **expanded nodes**.

<table>
<thead>
<tr>
<th>Prob#</th>
<th>PPLAN</th>
<th>goal-easy</th>
<th>goal-value</th>
<th>easy-goal</th>
<th>value-goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>29</td>
<td>34</td>
<td>20</td>
<td>27</td>
<td>8</td>
</tr>
<tr>
<td>8</td>
<td>42</td>
<td>12</td>
<td>12</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>55</td>
<td>13</td>
<td>13</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>11*</td>
<td>57</td>
<td>107</td>
<td>45</td>
<td>102</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>92</td>
<td>33</td>
<td>33</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>13</td>
<td>171</td>
<td>11617</td>
<td>11617</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>14</td>
<td>194</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>313</td>
<td>58</td>
<td>58</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>17</td>
<td>13787</td>
<td>12</td>
<td>12</td>
<td>7562</td>
<td>7</td>
</tr>
<tr>
<td>19*</td>
<td>&gt;20000</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>&gt;20000</td>
<td>71</td>
<td>71</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>22*</td>
<td>&gt;20000</td>
<td>85</td>
<td>30</td>
<td>7</td>
<td>145</td>
</tr>
<tr>
<td>23*</td>
<td>&gt;20000</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>24*</td>
<td>&gt;20000</td>
<td>49</td>
<td>22</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

*: Best value BDF preference cannot be achieved
We have proposed a **heuristic** algorithm for **QTEPs**.

**Key enablers:**
- Simplification of preference formulae.
- Transformation of temporal preferences into non-temporal ones.

We have **implemented** this algorithm **TLPlan**.

The algorithm shows **better performance** than existing planners.
Related Work

Languages and planners for QTEP

- [Delgrande et al., 2004]: Temporally extended preference language
- [Son and Pontelli, 2004]: Using Answer Set Programming.
- [Bienvenu et al., 2006]: Optimal Best-First Planning

2006 Planning Competition (Quantitative)

- Final-state preferences: Yochan^{PS} [Benton et al., 2006].
- Temporally extended preferences: SGPlan_5 [Hsu et al., 2007], MIPS-XXL [Edelkamp, 2006], MIPS-BDD [Edelkamp et al., 2006], HPLAN-P [Baier et al., 2007].
Let $P_1$ and $P_2$ be the preference vectors of states $s_1$ and $s_2$.

When is $s_1$ better than $s_2$?
Let $P_1$ and $P_2$ be the preference vectors of states $s_1$ and $s_2$.

When is $s_1$ better than $s_2$?

Regarding preferences, we have defined two criteria:

- $P_1 \prec_{\text{VALUE}} P_2$ Best-weighted BDF preference of $P_1$ estimated easier than that of $P_2$.

- $P_1 \prec_{\text{EASY}} P_2$ means that either $P_1$ contains a preference formula that has been estimated to be easier than all those in $P_2$. 
Putting pieces together: adding the goal

When is $s_1$ better than $s_2$?

Now we consider the goal:

- Let $G_1$ and $G_2$ be the value of the goal distance function for $s_1$ and $s_2$.
- The following strategies guide search towards the preferences and the goal.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Check whether</th>
<th>If tied, check whether</th>
</tr>
</thead>
<tbody>
<tr>
<td>goal-value</td>
<td>$G_1 &lt; G_2$</td>
<td>$P_1 &lt;_{\text{VALUE}} P_2$</td>
</tr>
<tr>
<td>goal-easy</td>
<td>$G_1 &lt; G_2$</td>
<td>$P_1 &lt;_{\text{EASY}} P_2$</td>
</tr>
<tr>
<td>value-goal</td>
<td>$P_1 &lt;_{\text{VALUE}} P_2$</td>
<td>$G_1 &lt; G_2$</td>
</tr>
<tr>
<td>easy-goal</td>
<td>$P_1 &lt;_{\text{EASY}} P_2$</td>
<td>$G_1 &lt; G_2$</td>
</tr>
</tbody>
</table>
References I

Jorge A. Baier and Sheila A. McIlraith.
Planning with first-order temporally extended goals using heuristic search.

J. Baier, F. Bacchus, and S. McIlraith.
A heuristic search approach to planning with temporally extended preferences.
To appear.

J. Benton, Subbarao Kambhampati, and Minh B. Do.
YochanPS: PDDL3 simple preferences and partial satisfaction planning.
Meghyn Bienvenu, Christian Fritz, and Sheila McIlraith.
Planning with qualitative temporal preferences.

Blai Bonet and Hector Geffner.
Planning as heuristic search.

James P. Delgrande, Torsten Schaub, and Hans Tompits.
Domain-specific preferences for causal reasoning and planning.
Stefan Edelkamp, Shahid Jabbar, and Mohammed Naizih.
Large-scale optimal PDDL3 planning with MIPS-XXL.

Stefan Edelkamp.
Optimal symbolic PDDL3 planning with MIPS-BDD.

Chih-Wei Hsu, Benjamin Wah, Ruoyun Huang, and Yixin Chen.
Constraint partitioning for solving planning problems with trajectory constraints and goal preferences.
To appear.
Tran Cao Son and Enrico Pontelli.
Planning with preferences using logic programming.