Short-Sighted Stochastic Shortest Path Problems

Felipe Trevizan and Manuela Veloso
School of Computer Science
Carnegie Mellon University
June 29, 2012
Announcement

• Wrong version of the paper in the proceedings in the *thumbdrive* and the *cd-rom*:

Short-Sighted Stochastic Shortest Path Problems

Felipe W. Trevizan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
fwt@cs.cmu.edu

Manuela M. Veloso
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
mmv@cs.cmu.edu

Abstract

Two extreme approaches can be applied to solve a probabilistic planning problem, namely closed loop algorithms and open loop (a.k.a. replanning) algorithms. While closed loop algorithms invest significant computational effort to generate a closed form solution, open loop algorithms compute open form solutions and interact with the environment in order to refine the computed solution. In this paper, we introduce short-sighted Stochastic Shortest Path (SSP), a new model in which solutions computed based on it can be executed for at least $t$ steps as a closed form solution. Using short-sighted SSPs, we present a novel probabilistic planner

RTDP (Bonet and Geffner 2003), resulting in optimal algorithms with convergence bounds. Due to the pruning in

• The right version is in the online proceedings
Motivation

• Two classes of solutions to probabilistic planning problems:
  – Complete policy (a.k.a. universal plan):
    • Maps **every** state to an action
    • Never fails, i.e., no need to replan
    • Optimal
    • Doesn’t scale up
Motivation

• Two classes of solutions to probabilistic planning problems:
  – Complete policy (a.k.a. universal plan):
    • Maps every state to an action
    • Never fails, i.e., no need to replan
    • Optimal
    • Doesn’t scale up
  – Partial policy:
    • Maps some states to an action
    • Can fail, i.e., reaches an unpredicted state and replan from there
    • Non-optimal
    • Scales up
Contributions

• A framework that offers a **new trade-off** between complete and partial policies:
  – A new **model**: short-sighted Stochastic Shortest Path Problems
  – A new **planner**: short-sighted probabilistic planner
• Model problems as Stochastic Shortest Path Problems
Big Picture

- Model problems as Stochastic Shortest Path Problems
- Generate short-sighted subproblems
Big Picture

• Model problems as Stochastic Shortest Path Problems
• Generate short-sighted subproblems
• Solve the subproblems and execute this solution
Big Picture

- Model problems as Stochastic Shortest Path Problems
- Generate short-sighted subproblems
- Solve the subproblems and execute this solution
• Model problems as Stochastic Shortest Path Problems
• Generate short-sighted subproblems
• Solve the subproblems and execute this solution
• Model problems as Stochastic Shortest Path Problems
• Generate short-sighted subproblems
• Solve the subproblems and execute this solution

• Analyze single and multiple execution cases
• Model problems as Stochastic Shortest Path Problems
• Generate short-sighted subproblems
• Solve the subproblems and execute this solution

• Analyze single and multiple execution cases
Stochastic Shortest Path Problems (SSPs)

An SSP is the tuple $<S, s_0, G, A, P, C>$:

- Set of states $S$
- Initial state $s_0$
- Set of goal states $G \subseteq S$
- Set of actions $A$
- Transition prob. $P(s' | s, a)$
- Cost $C(s, a, s') > 0$
  - defined when $P(s' | s, a) > 0$

In the example:

$G = \{s_G\}$

$C(a_0, )$:

<table>
<thead>
<tr>
<th>$s/s'$</th>
<th>$s_0$</th>
<th>$s_3$</th>
<th>$s_G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>--</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>$s_3$</td>
<td>1</td>
<td>5</td>
<td>--</td>
</tr>
</tbody>
</table>

$C(a_1, )$:

<table>
<thead>
<tr>
<th>$s/s'$</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>$s_G$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_0$</td>
<td>1</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>$s_1$</td>
<td>--</td>
<td>1</td>
<td>--</td>
</tr>
<tr>
<td>$s_2$</td>
<td>--</td>
<td>--</td>
<td>30</td>
</tr>
</tbody>
</table>
Optimal policies

• An **optimal** policy $\pi^*$ minimizes the expected cost to reach a goal state from $s_0$

• The **minimum** expected cost to reach a goal state from a state $s$ is:

$$V^*(s) = \begin{cases} 
0 & \text{if } s \in G \\
\min_{a \in A} \sum_{s' \in S} P(s'|s, a)[C(s, a, s') + V^*(s')] & \text{otherwise}
\end{cases}$$
Short-Sighted SSPs: Idea

• Manage uncertainty by:
  – Considering the uncertainty structure in the **neighborhood** of the current state; and
  – Adding **artificial goals** to heuristically approximate the pruned states.
Short-Sighted SSPs: Idea

• Manage uncertainty by:
  – Considering the uncertainty structure in the **neighborhood** of the current state; and
  – Adding **artificial goals** to heuristically approximate the pruned states.

• $\delta(s, s')$: minimum number of actions to reach $s'$ from $s$
Short-Sighted SSPs: Definition

Given: • an SSP <S,s₀,G,A,P,C>,
   • s ∈ S
   • t > 0 and
   • a heuristic function H

the (s,t)-short-sighted SSP is <S’,s,G’,A,P,C’>:  

16
Short-Sighted SSPs: Definition

Given:
• an SSP \(<S, s_0, G, A, P, C>\),
  • \(s \in S\)
  • \(t > 0\) and
  • a heuristic function \(H\)

the \((s, t)\)-short-sighted SSP is \(<S', s, G', A, P, C'>\):

\[
S' = \{s' \in S | \delta(s, s') \leq t\}
\]
Short-Sighted SSPs: Definition

Given: • an SSP $<S,s_0,G,A,P,C>$,
   • $s \in S$
   • $t > 0$ and
   • a heuristic function $H$
the $(s,t)$-short-sighted SSP is $<S',s,G',A,P,C'>$:

- $S' = \{ s' \in S | \delta(s, s') \leq t \}$
- $G' = \{ s' \in S | \delta(s, s') = t \} \cup (G \cap S')$
Short-Sighted SSPs: Definition

Given: • an SSP \(<S,s_0,G,A,P,C>\),
• \(s \in S\)
• \(t > 0\) and
• a heuristic function \(H\)

the \((s,t)\)-short-sighted SSP is \(<S',s,G',A,P,C'>\):

\[
\begin{align*}
S' &= \{ s' \in S | \delta(s, s') \leq t \} \\
G' &= \{ s' \in S | \delta(s, s') = t \} \cup (G \cap S') \\
C'(s, a, s') &= \begin{cases} 
C(s, a, s') + H(s') & \text{if } s' \in G' \\
C(s, a, s') & \text{otherwise}
\end{cases}
\end{align*}
\]

States reachable using up to \(t\) actions

Artificial goal: states reachable using exactly \(t\) actions

If \(s'\) is an artificial goal, then its cost is incremented by its heuristic value
Short-Sighted SSPs: Examples

- Original problem:
Short-Sighted SSPs: Examples

• Original problem:

- $(s_0, 1)$-short-sighted:

$$
\begin{array}{c|c}
\delta(s_0, s) & \\
\hline
s_0 & 0 \\
\hline
s_1 & 1 \\
\hline
s_2 & 2 \\
\hline
s_3 & 1 \\
\hline
s_G & 1 \\
\end{array}
$$
Short-Sighted SSPs: Examples

- Original problem:

- $(s_0, 1)$-short-sighted:

- $(s_0, 2)$-short-sighted:
Short-Sighted SSPs and Look-ahead

Original problem

Look-ahead \((s_0, t=2)\)
Short-Sighted SSPs and Look-ahead
Short-Sighted SSPs and Look-ahead

Key difference: Short-sighted SSPs preserve the action structure, e.g., self-loop actions and loops of actions.
Short-Sighted SSPs and Look-ahead

Key difference: Short-sighted SSPs preserve the action structure, e.g., self-loop actions and loops of actions

Theorem: the optimal value-function for an (s,t)-short-sighted SSP is at least as good as the t-look-ahead value of s, i.e.,

\[ L_t(s_0) \leq \hat{V}_t^*(s_0) \leq V^*(s_0) \]
Short-Sighted Probabilistic Planner (SSiPP)

Since short-sighted SSPs are much smaller than the original problem, we can compute a complete policy for them.
SSiPP and replanning

• **Theorem**: at least $t$ actions are executed in the environment before replanning is needed

![Diagram of state transitions](image)

• The policy can be executed for more than 2 timesteps:

\[
S_0 \xrightarrow{a_0} S_3 \xrightarrow{a_0} S_3 \xrightarrow{a_0} S_3 \xrightarrow{a_0} \cdots
\]
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time

• **Theorem:** SSiPP is *asymptotically optimal*, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time

• **Theorem:** SSiPP is asymptotically optimal, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time

• **Theorem:** SSiPP is asymptotically optimal, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
Multiple Runs of SSiPP

- Two scenarios:
  - the same problem is solved more than once; or
  - simulation is allowed for a given amount of time

- **Theorem**: SSiPP is **asymptotically optimal**, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time

• Theorem: SSiPP is asymptotically optimal, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
Multiple Runs of SSiPP

• Two scenarios:
  – the same problem is solved more than once; or
  – simulation is allowed for a given amount of time

• **Theorem**: SSiPP is *asymptotically optimal*, i.e., if the same problem is solved sufficiently many times, then the optimal policy is found.
SSiPP and Real Time Dynamic Programming (RTDP)

• RTDP and *anytime* SSiPP:
  – can be seen as asynchronous value iteration
  – differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

• RTDP and *anytime* SSiPP:
  – can be seen as asynchronous value iteration
  – differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

• RTDP and *anytime* SSiPP:
  – can be seen as asynchronous value iteration
  – differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and \textit{anytime} SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
SSiPP and Real Time Dynamic Programming (RTDP)

- RTDP and *anytime* SSiPP:
  - can be seen as asynchronous value iteration
  - differ in the scheduling of Bellman updates (backups)
Experiments

- **Goal**: compare SSiPP against the winners of the previous International Probabilistic Planning Competitions (IPPCs)

- **Methodology**: (same as IPPC’04 and IPPC’06)
  - For each problem, planners are requested to solve it 50 times in 20 minutes
  - Learning is allowed between attempts of the same problem
  - The evaluation metric is the number of times the goal is reached
Planners

• We compare the following planners
  – **SSiPP**: using LRTDP as optimal solver
  – **LRTDP**: 2nd place IPPC’04
  – **FF-Replan**: 1st place IPPC’04
  – **FPG**: 1st place IPPC’06
  – **RFF**: 1st place IPPC’08

• Parametrizations for SSiPP and LRTDP:
  – $t \in \{1,2,3,\ldots,10\}$
  – $H$: zero-heuristic, FF+all-outcomes, min-min
Triangle tireworld: results

SSiPP: $t = 3$, $H = \text{FF+all-outcome}$
SSiPP: $t = 8$, $H = \text{zero-heuristic}$
LRTDP: $t = 3$, $H = \text{zero-heuristic}$
FF-Replan
FPG
RFF
Blocks World: result

% of rounds that reached the goal

Problem #

SSiPP: $t = 3, H = \text{FF+all-outcome}$

SSiPP: $t = 2, H = \text{FF+all-outcome}$

LRTDP: $t = 3, H = \text{zero-heuristic}$

FF-Replan

FPG

RFF
Exploding Blocks World: results

- SSiPP: $t = 3$, $H = \text{FF+all-outcome}$
- SSiPP: $t = 3$, $H = \text{FF+all-outcome}$
- LRTDP: $t = 3$, $H = \text{zero-heuristic}$
- FF-Replan
- FPG
- RFF

% of rounds that reached the goal
Zeno Travel: results

% of rounds that reached the goal

Problem #

SSiPP: $t = 3$, $H = \text{FF+all-outcome}$
SSiPP: $t = 2$, $H = \text{FF+all-outcome}$
LRTDP: $t = 3$, $H = \text{zero-heuristic}$
FF-Replan
FPG
RFF
## IPPC experiment: summary

<table>
<thead>
<tr>
<th></th>
<th>SSiPP Overall</th>
<th>SSiPP Per Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outperforms all</td>
<td>16.6%</td>
<td>36.6%</td>
</tr>
<tr>
<td>Ties with the best</td>
<td>41.6%</td>
<td>53.3%</td>
</tr>
</tbody>
</table>
In the considered problems:
- SSiPP is never the last place in any of the problems
- LRTDP never outperforms SSiPP

<table>
<thead>
<tr>
<th></th>
<th>SSiPP Overall</th>
<th>SSiPP Per Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outperforms all</td>
<td>16.6%</td>
<td>36.6%</td>
</tr>
<tr>
<td>Ties with the best</td>
<td>41.6%</td>
<td>53.3%</td>
</tr>
</tbody>
</table>
Conclusion and Future Work

• A framework that offers a new trade-off between complete and partial policies
  – Short-sighted SSPs
Conclusion and Future Work

• A framework that offers a new trade-off between complete and partial policies
  – Short-sighted SSPs
  – SSiPP
    • as replanner: no replanning needed for at least t actions
    • as anytime algorithm: same guarantees as RTDP
Conclusion and Future Work

• A framework that offers a new trade-off between complete and partial policies
  – Short-sighted SSPs
  – SSiPP
    • as replanner: no replanning needed for at least t actions
    • as anytime algorithm: same guarantees as RTDP

• Future work:
  – New definitions of short-sighted *spaces* (*S’*)
  – Improve scalability in problems that are not probabilistic interesting
Thank you!
Questions?