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

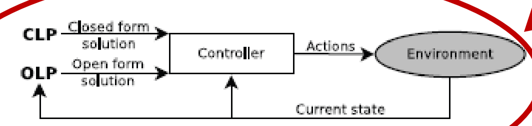


Figure 1: Interaction with the environment for closed loop planners (CLPs) and open loop planners (OLPs).

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

Camera-ready version doesn't have this picture

- 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

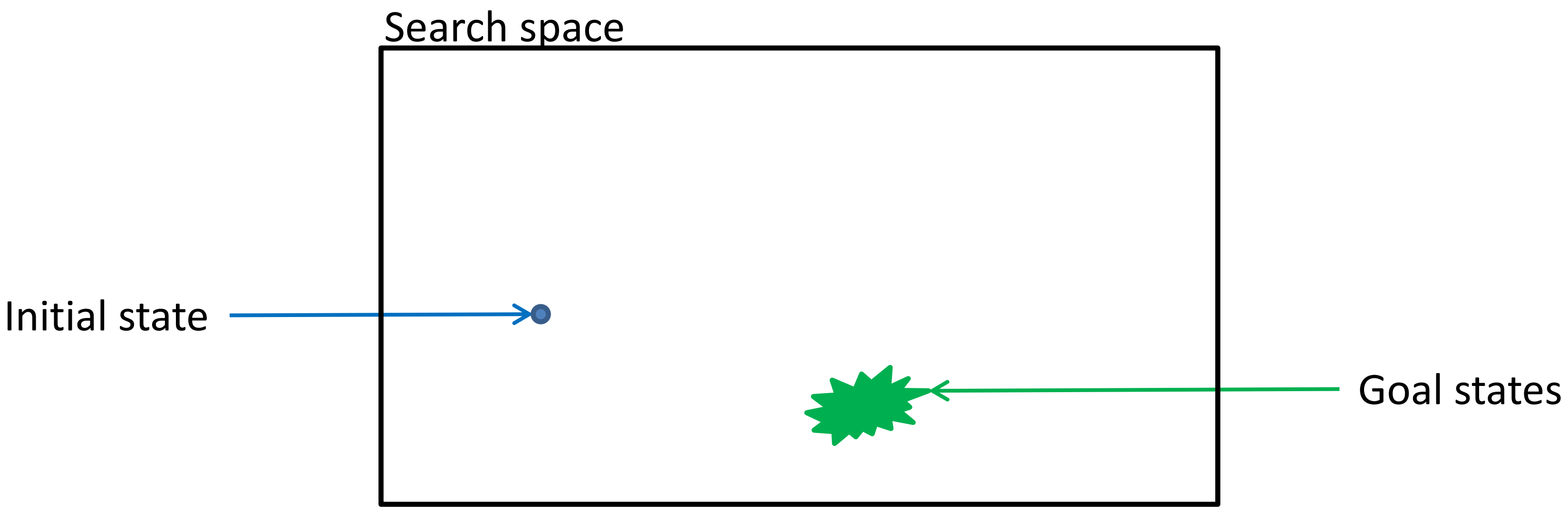
- Two classes of solutions to probabilistic planning problems:
  - Complete policy (a.k.a. universal plan):
    - Maps **every** state to an action
    - Never fails, i.e., no need to replan
    - Optimal
    - Doesn't scale up
  - Partial policy:
    - Maps **some** states to an action
    - Can fail, 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

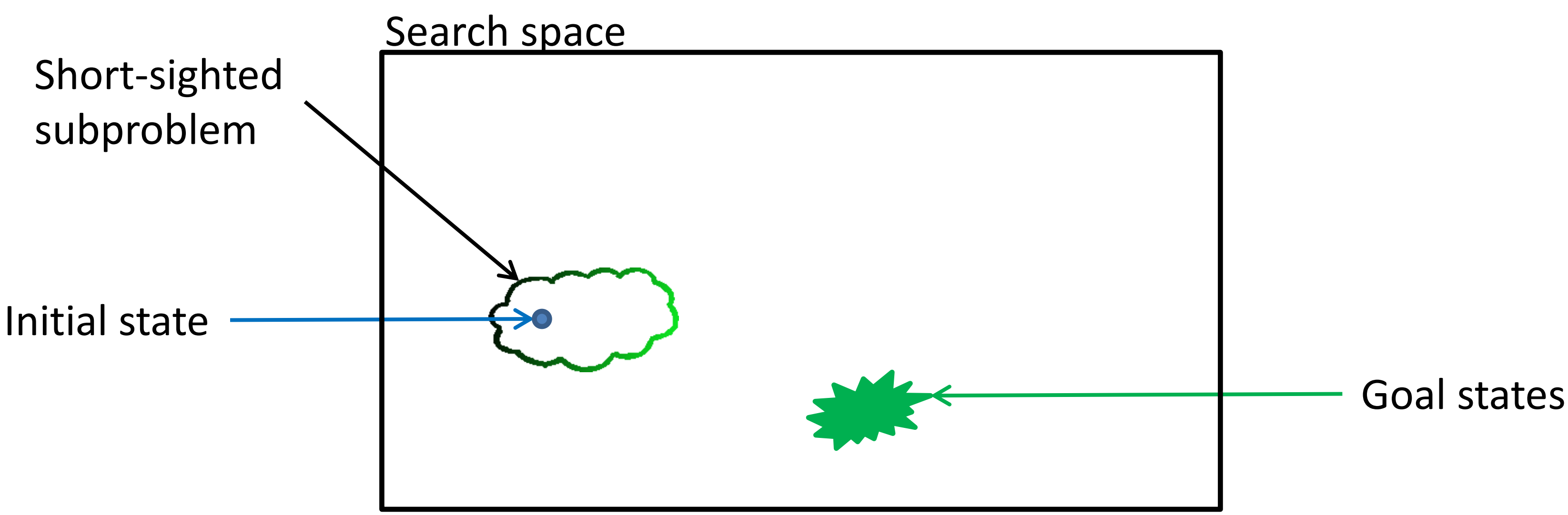
Big Picture

- Model problems as Stochastic Shortest Path Problems



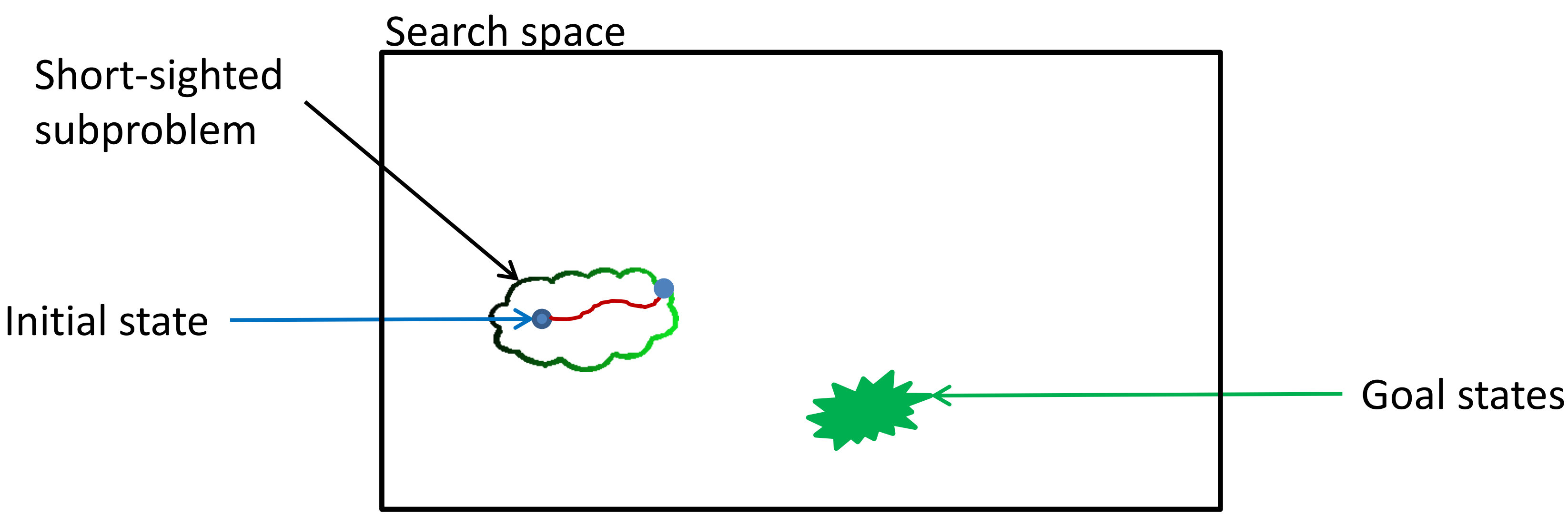
Big Picture

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



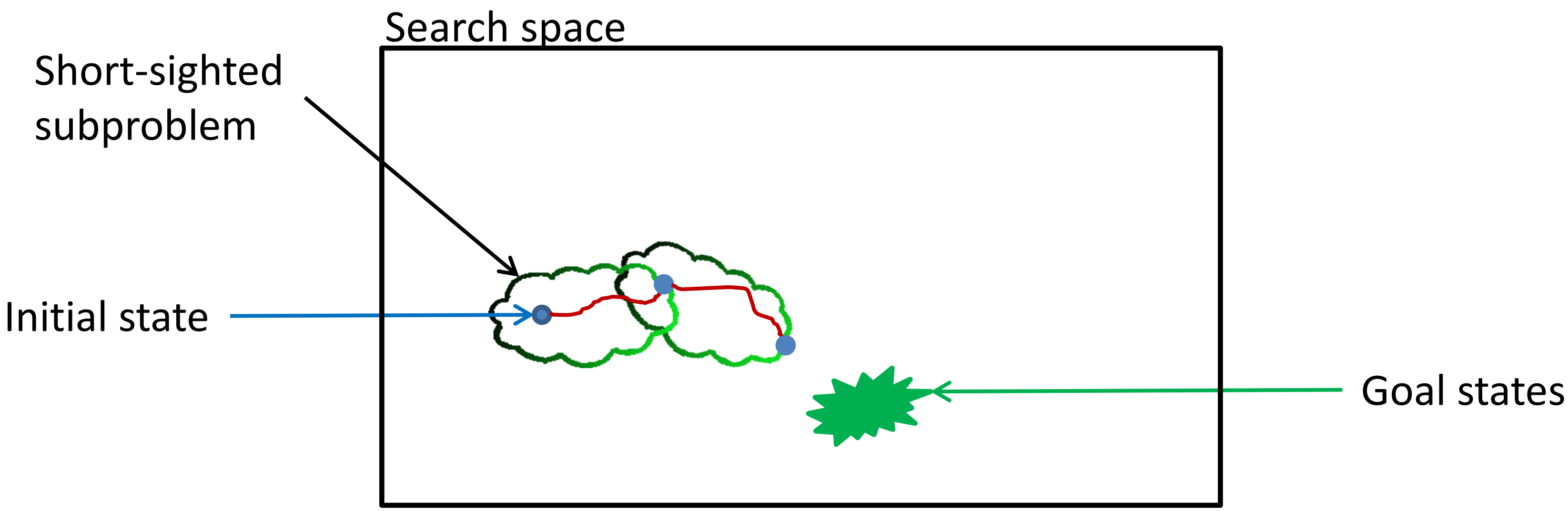
Big Picture

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



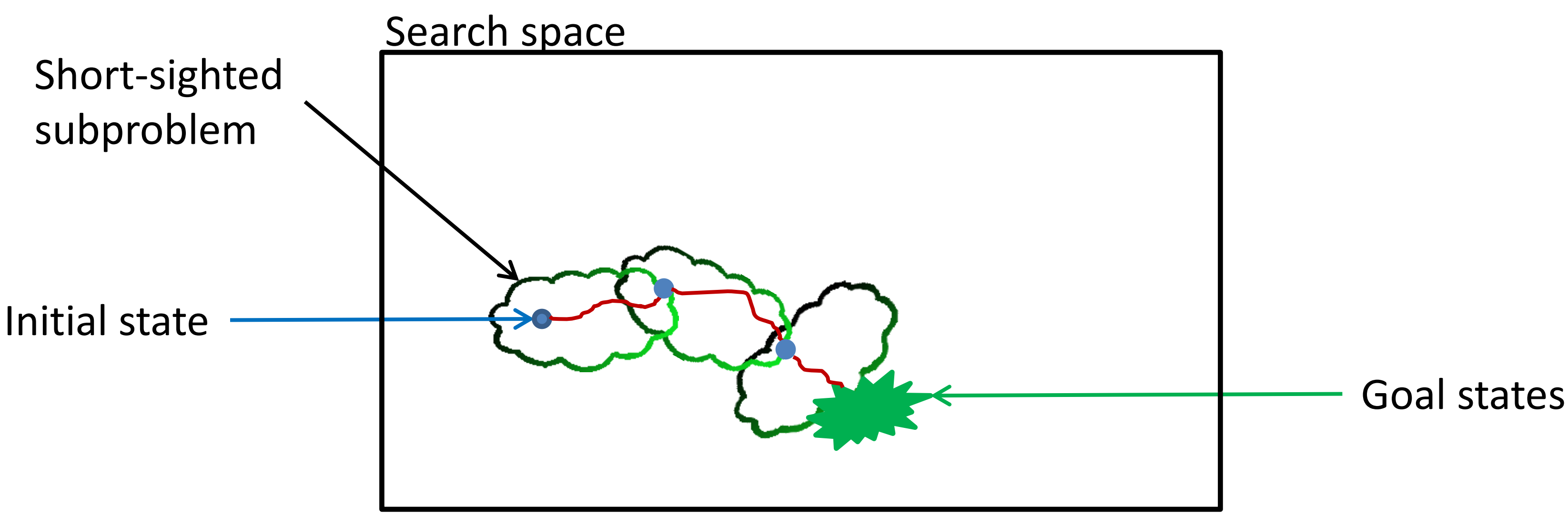
Big Picture

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



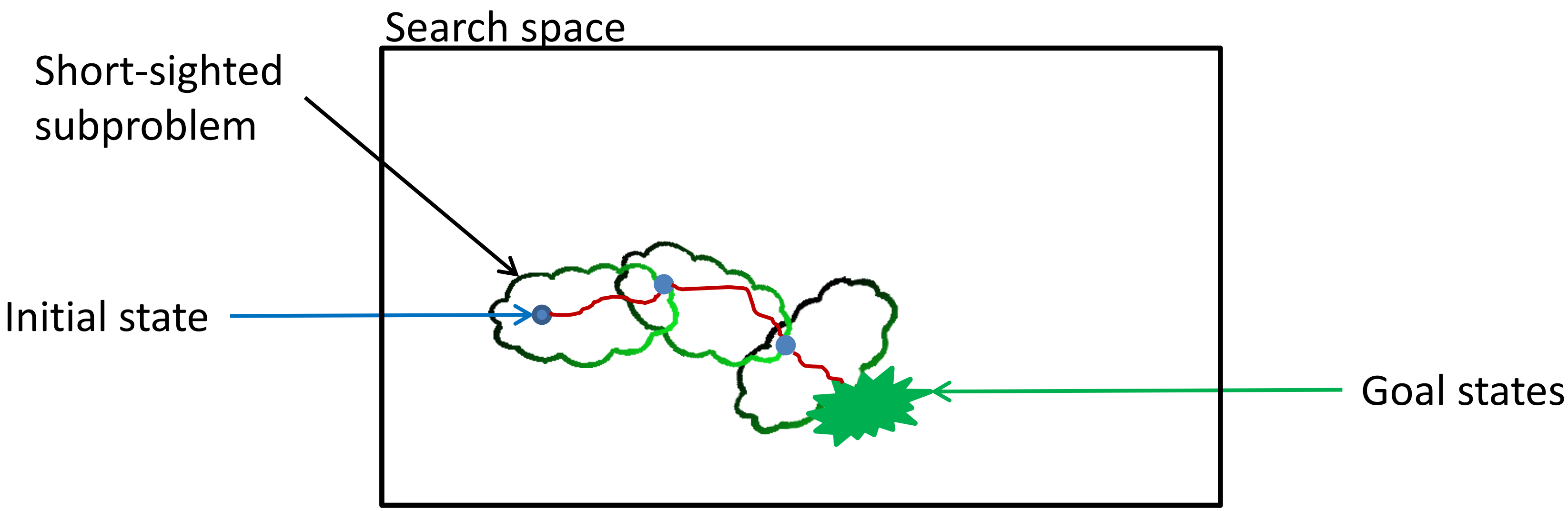
Big Picture

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



Big Picture

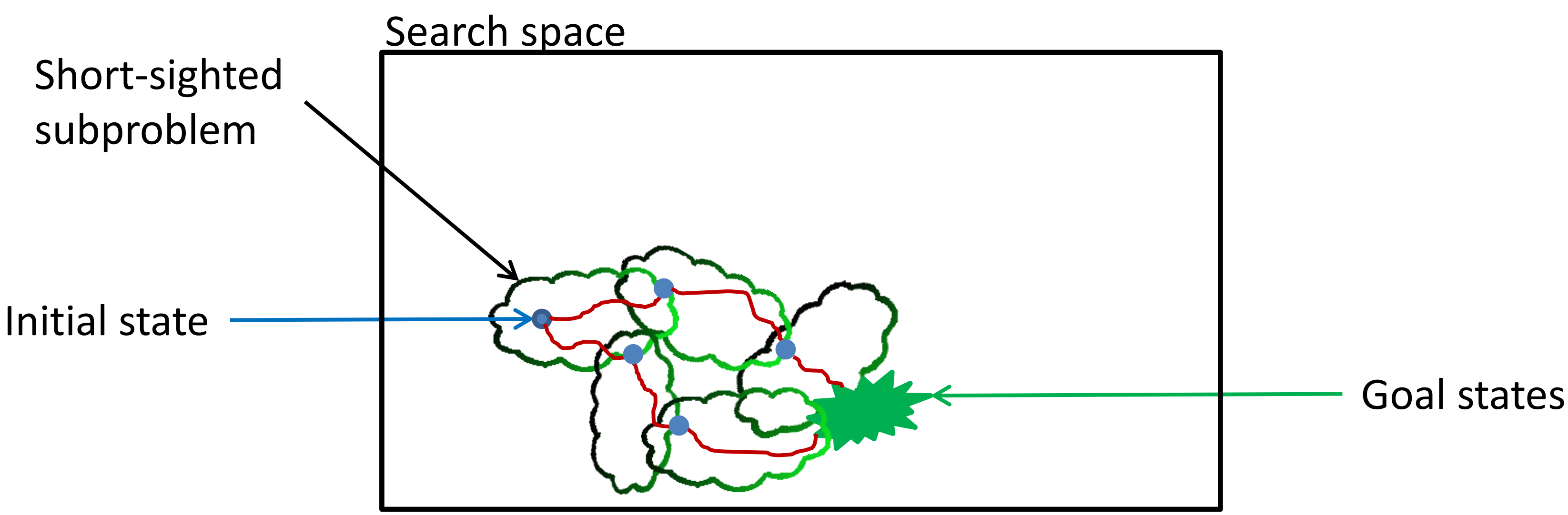
- Model problems as Stochastic Shortest Path Problems
- Generate short-sighted subproblems
- Solve the subproblems and execute this solution



- Analyze single and multiple execution cases

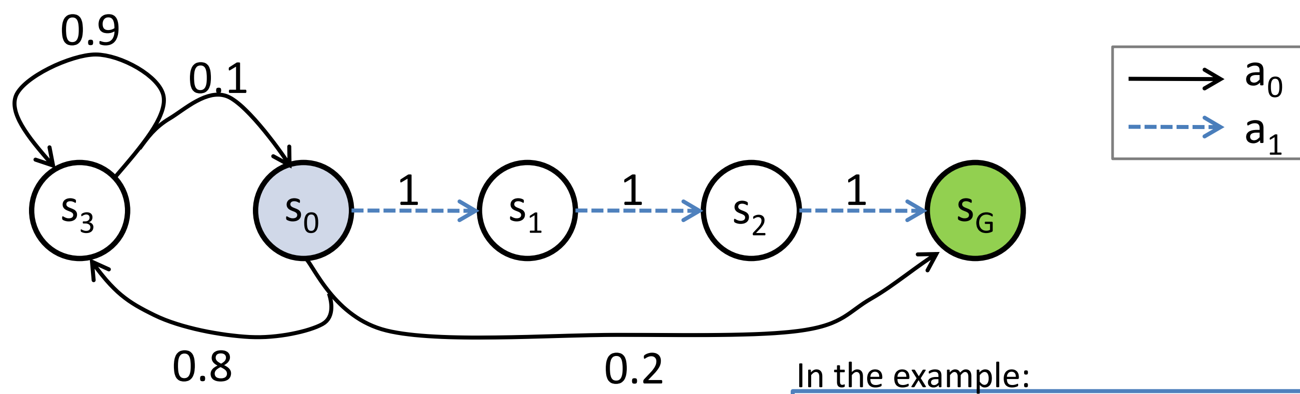
Big Picture

- 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 $\langle S, s_0, G, A, P, C \rangle$:

- 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:

$G = \{s_G\}$

$C(\ ,a_0,\ )$:

s/s'	s_0	s_3	s_G
s_0	--	5	1
s_3	1	5	--

$C(\ ,a_1,\ )$:

s/s'	s_1	s_2	s_G
s_0	1	--	--
s_1	--	1	--
s_2	--	--	30

Optimal policies

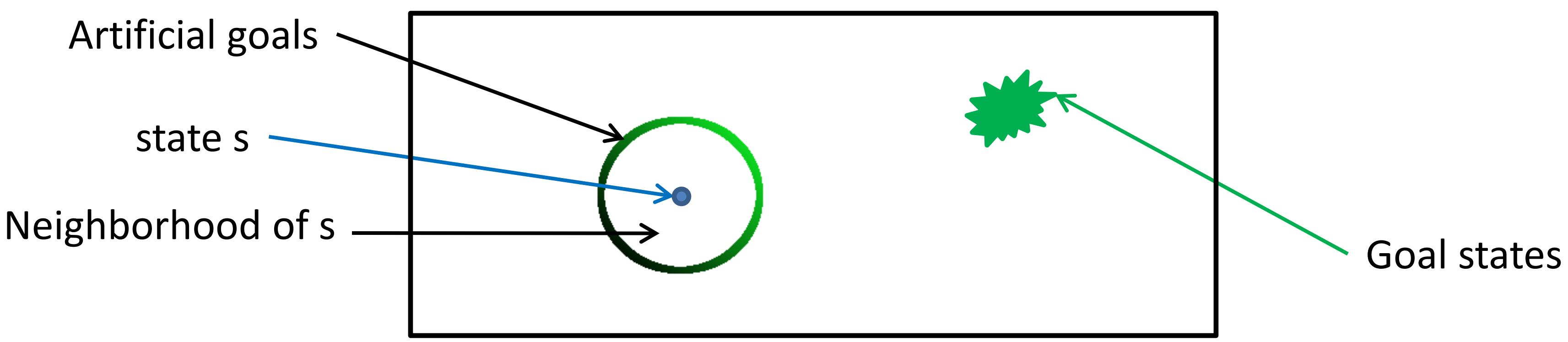
- An **optimal** policy π^* minimizes the expected cost to reach a goal state from s_0

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

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

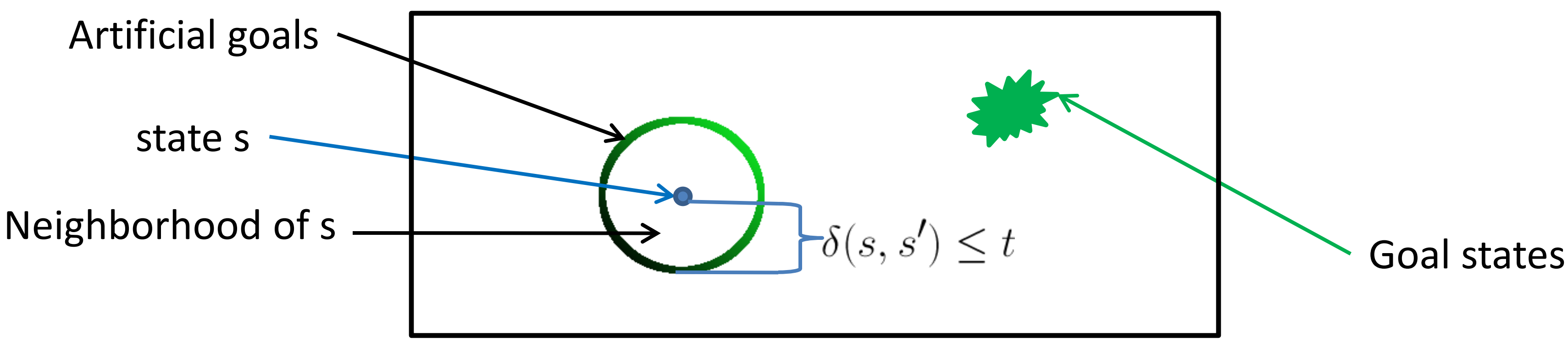
Short-Sighted SSPs: Idea

- 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 $\langle S, s_0, G, A, P, C \rangle$,
- $s \in S$
- $t > 0$ and
- a heuristic function H

the (s,t)-short-sighted SSP is $\langle S', s, G', A, P, C' \rangle$:

Short-Sighted SSPs: Definition

Given:
- an SSP <S,s_0,G,A,P,C>,
- $s \in \mathsf{S}$
- t > 0 and
- a heuristic function H

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

States reachable using **up to t** actions

- $\mathsf{S}' = \{s' \in \mathsf{S} | \delta(s, s') \leq t\}$

Short-Sighted SSPs: Definition

Given:
- an SSP <S,s_0,G,A,P,C>,
- $s \in \mathsf{S}$
- t > 0 and
- a heuristic function H

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

- $$\mathsf{S}' = \{s' \in \mathsf{S} | \delta(s, s') \leq t\}$$
- $$\mathsf{G}' = \{s' \in \mathsf{S} | \delta(s, s') = t\} \cup (\mathsf{G} \cap \mathsf{S}')$$

States reachable using **up to t** actions

Artificial goal: states reachable using **exactly t** actions

Short-Sighted SSPs: Definition

Given: • an SSP <S,s_0,G,A,P,C>,

- $s \in \mathsf{S}$
- t > 0 and
- a heuristic function H

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

States reachable using **up to t** actions

- $\mathsf{S}' = \{s' \in \mathsf{S} | \delta(s, s') \leq t\}$

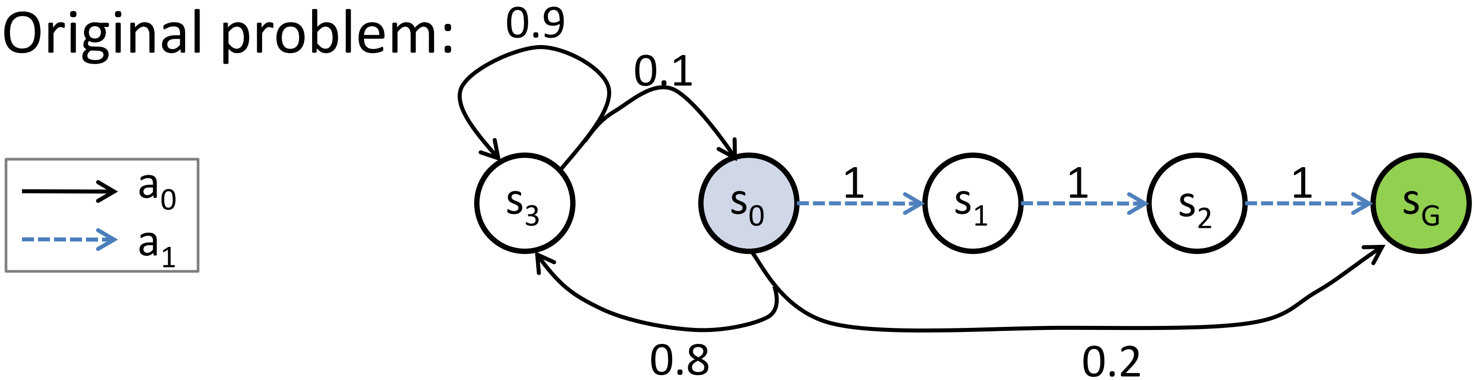
Artificial goal: states reachable using **exactly t** actions

- $\mathsf{G}' = \{s' \in \mathsf{S} | \delta(s, s') = t\} \cup (\mathsf{G} \cap \mathsf{S}')$
- $C'(s,a,s') = \begin{cases} C(s,a,s') + H(s') & \text{if } s' \in \mathsf{G}' \\ C(s,a,s') & \text{otherwise} \end{cases}$

If s' is an artificial goal, then its cost is incremented by its heuristic value

Short-Sighted SSPs: Examples

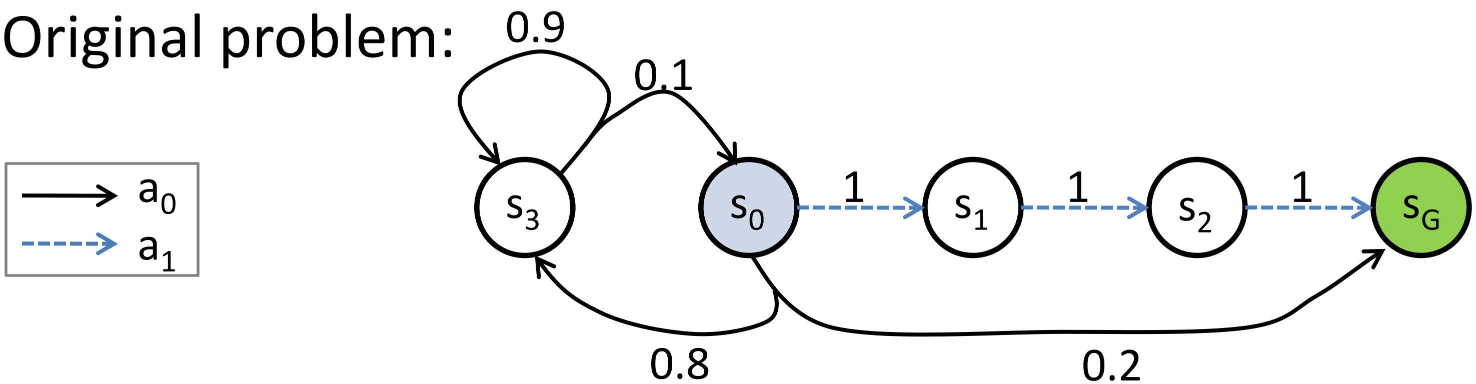
- Original problem:



	$\delta(s_0, s)$
s_0	0
s_1	1
s_2	2
s_3	1
s_G	1

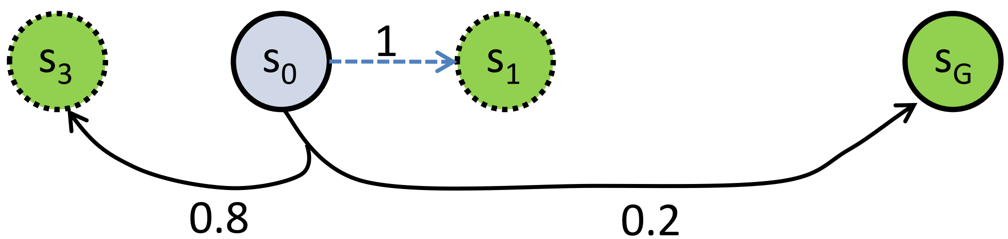
Short-Sighted SSPs: Examples

- Original problem:



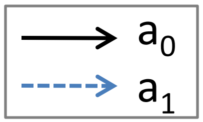
	$\delta(s_0, s)$
s_0	0
s_1	1
s_2	2
s_3	1
s_G	1

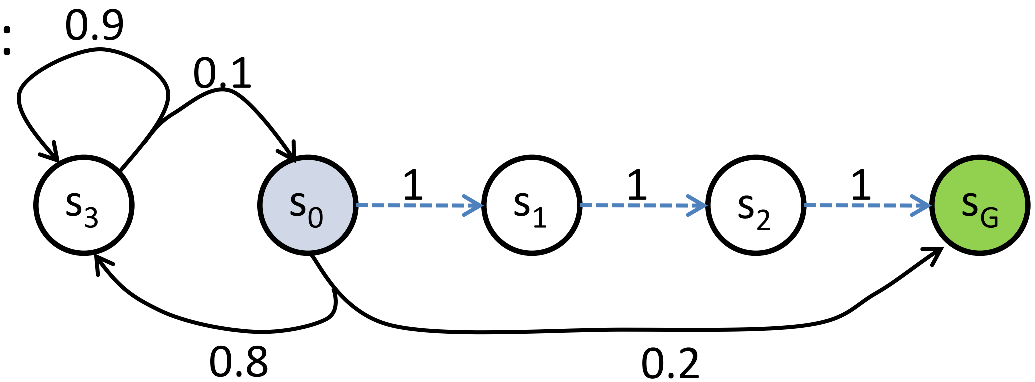
- $(s_0,1)$-short-sighted:



Short-Sighted SSPs: Examples

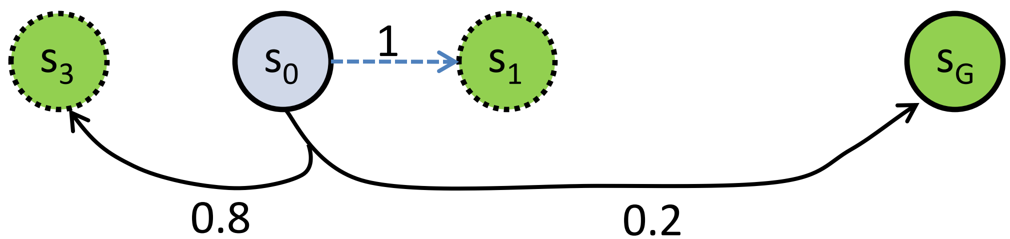
- Original problem:



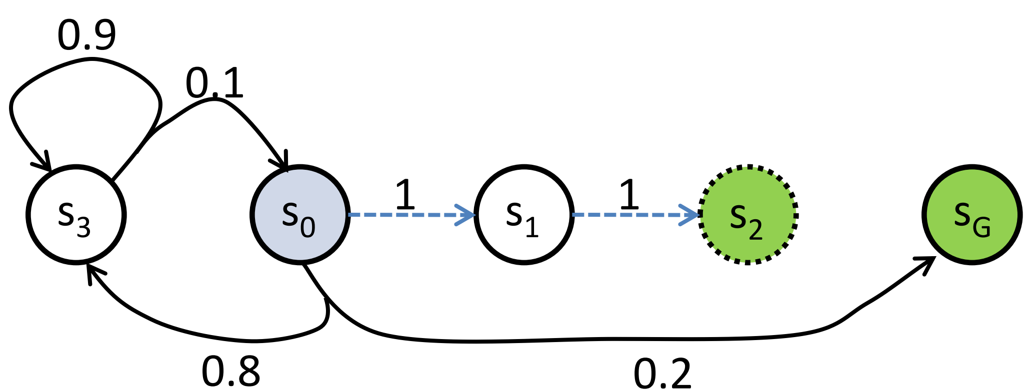


	$\delta(s_0, s)$
s_0	0
s_1	1
s_2	2
s_3	1
s_G	1

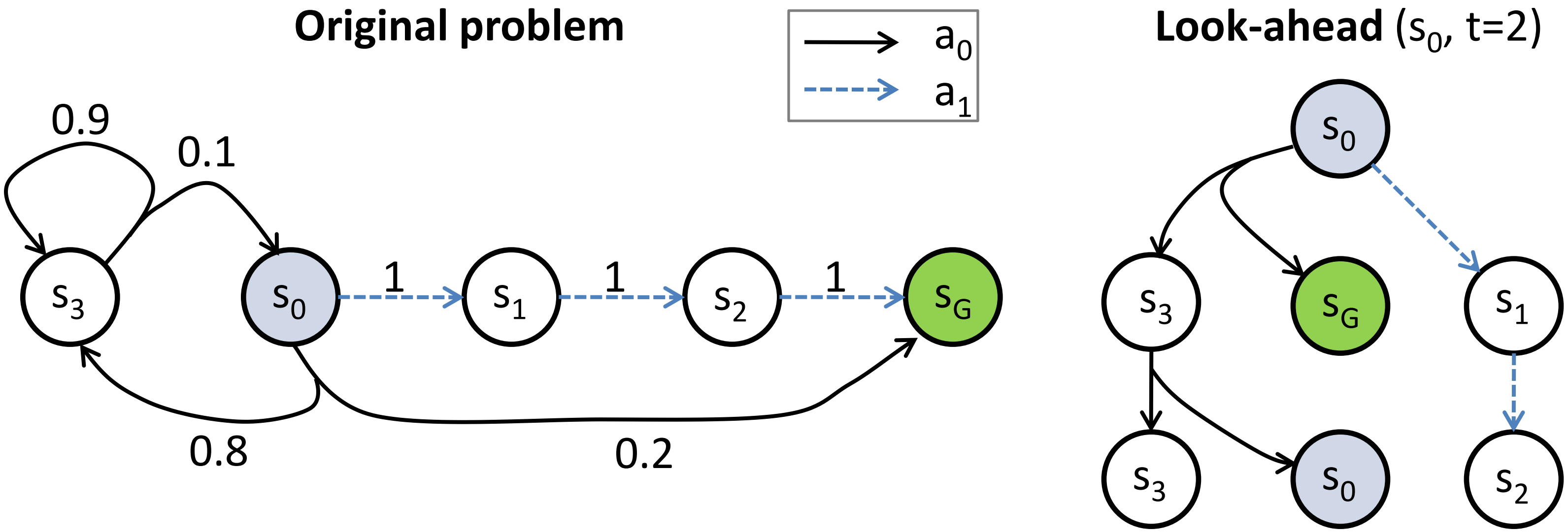
- $(s_0,1)$-short-sighted:



- $(s_0,2)$-short-sighted:

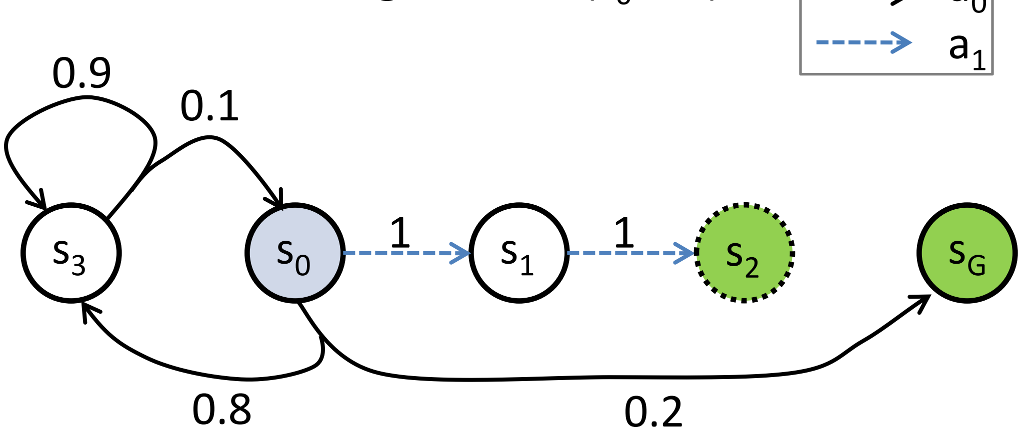


Short-Sighted SSPs and Look-ahead

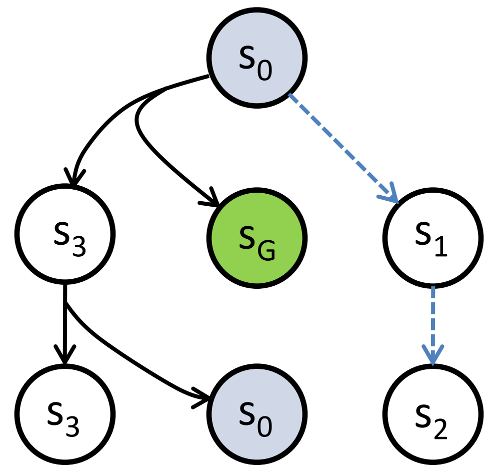


Short-Sighted SSPs and Look-ahead

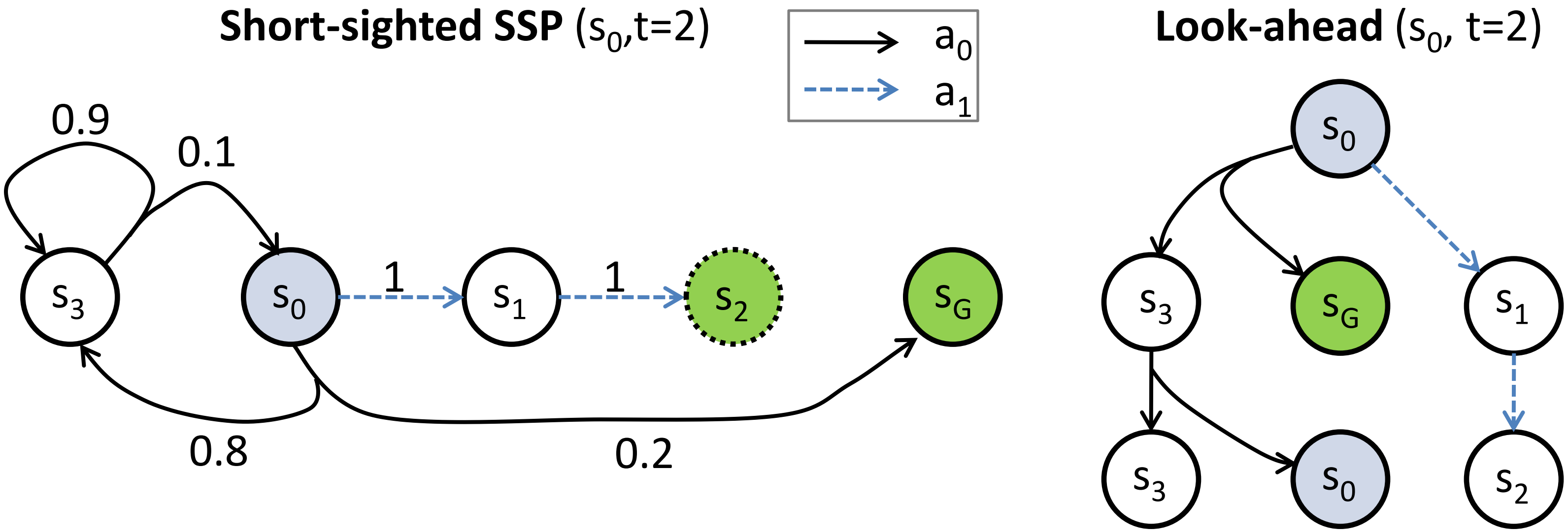
Short-sighted SSP (s_0,t=2)



Look-ahead (s_0, t=2)

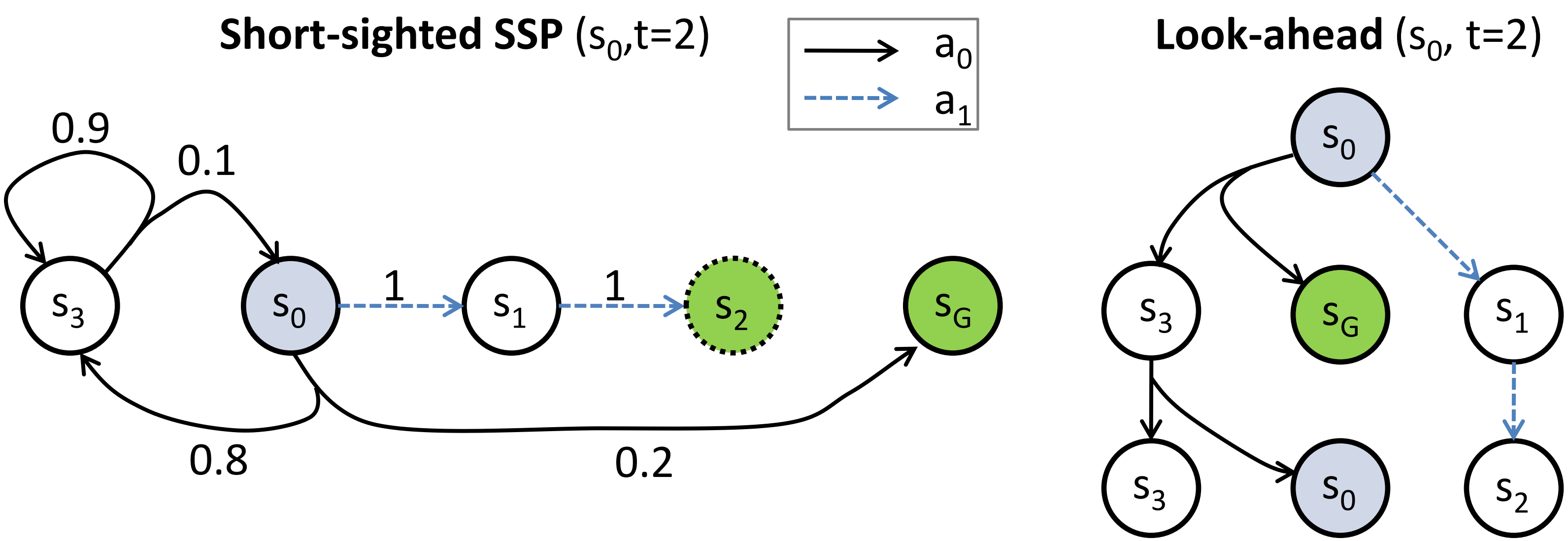


Short-Sighted SSPs and Look-ahead



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.,

$$L_t(s_0) \leq \hat{V}_t^*(s_0) \leq V^*(s_0)$$

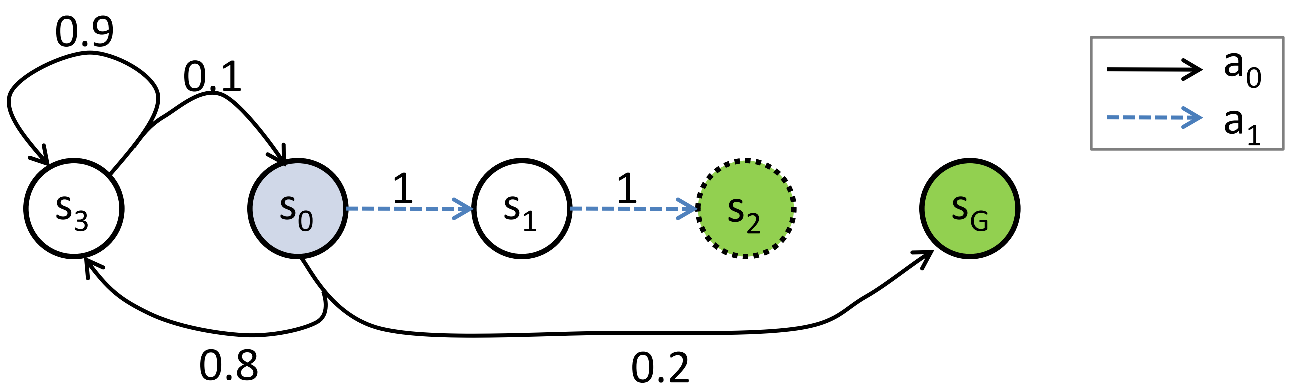
Short-Sighted Probabilistic Planner (SSiPP)

begin
  $s \leftarrow s_0$
  while $s \notin \mathbf{G}$ **do**
    $\langle \mathbf{S}', s, \mathbf{G}', \mathbf{A}, P, C' \rangle \leftarrow$ GENERATE-SHORT-SIGHTED-SSP$(\mathbb{S}, s, H)$
    $\hat{\pi}^* \leftarrow$ OPTIMAL-SSP-SOLVER$(\langle \mathbf{S}', s, \mathbf{G}', \mathbf{A}, P, C' \rangle, H)$
    while $s \notin \mathbf{G}'$ **do**
      $s \leftarrow$ execute-action$(\hat{\pi}^*(s))$
end

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



- The policy

s_0	s_3
a_0	a_0

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

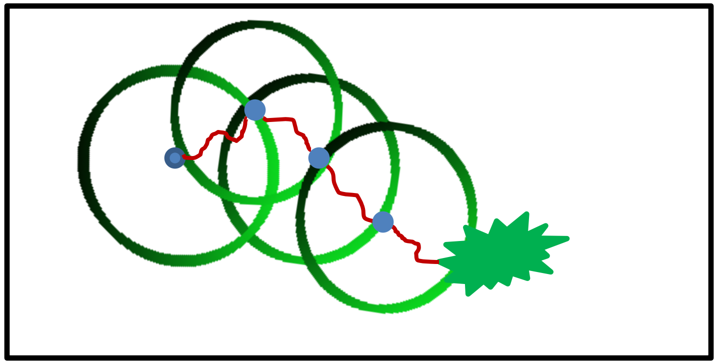
Multiple Runs of SSiPP

- 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.

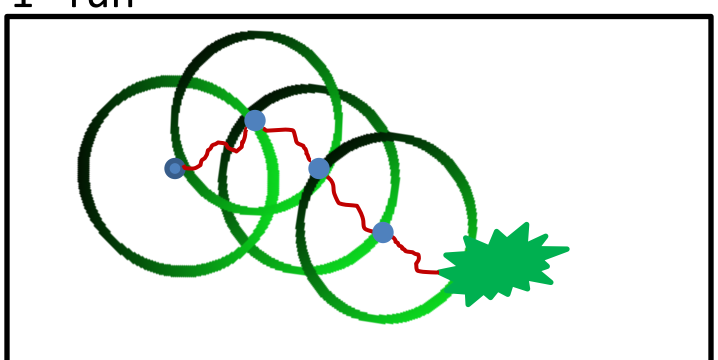
1st run



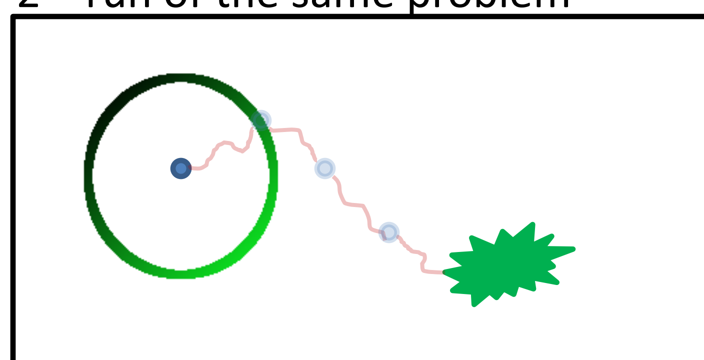
Multiple Runs of SSiPP

- 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.

1st run



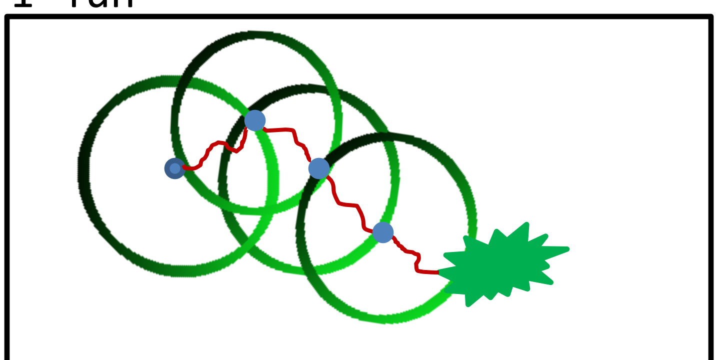
2nd run of the same problem



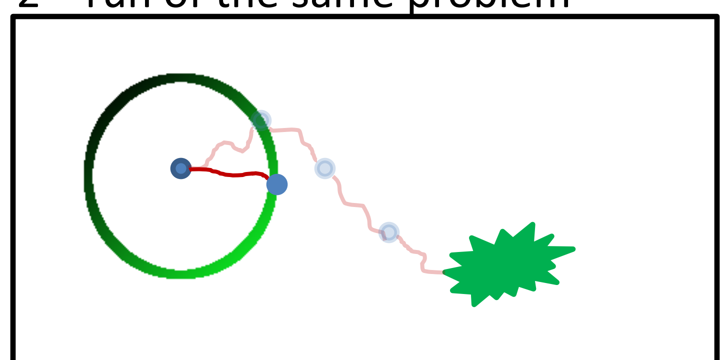
Multiple Runs of SSiPP

- 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.

1st run



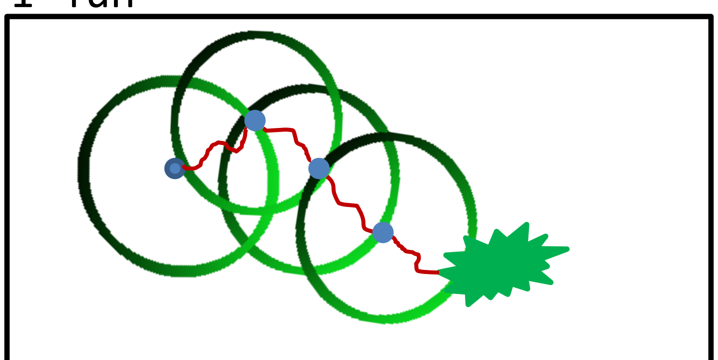
2nd run of the same problem



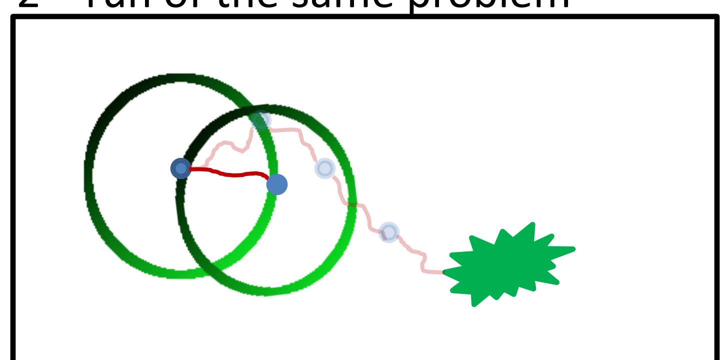
Multiple Runs of SSiPP

- 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.

1st run



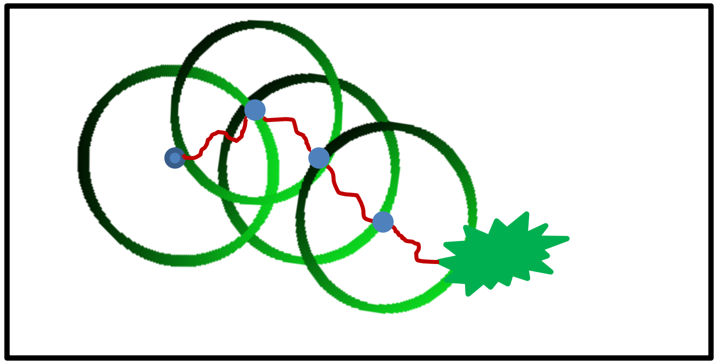
2nd run of the same problem



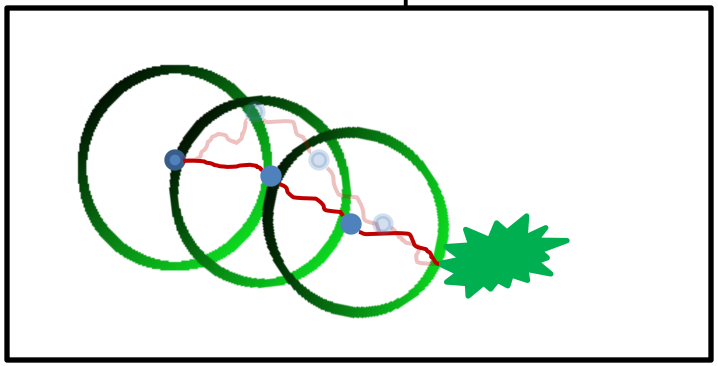
Multiple Runs of SSiPP

- 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.

1st run



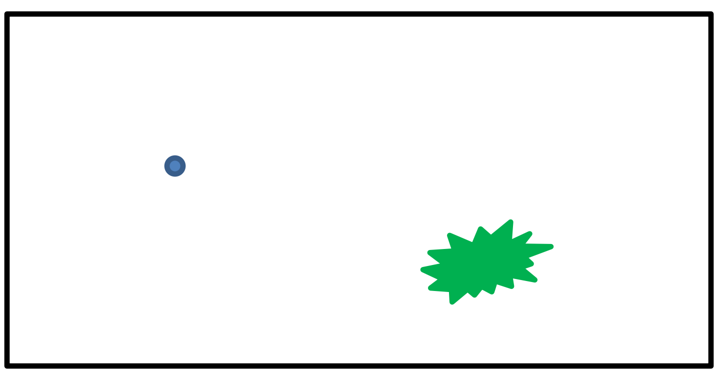
2nd run of the same problem



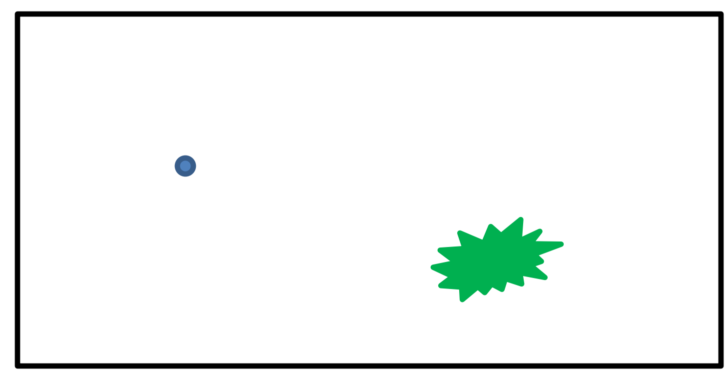
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



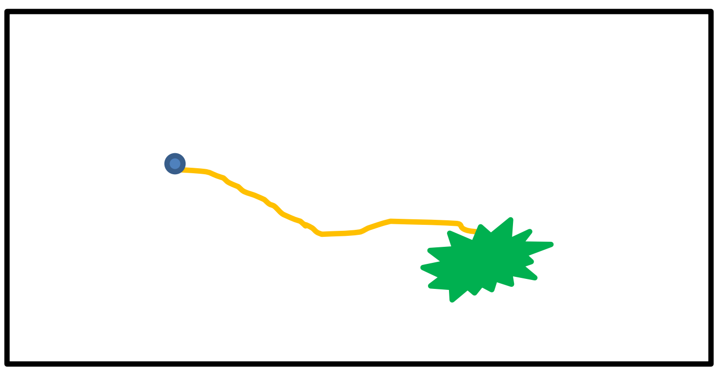
SSiPP



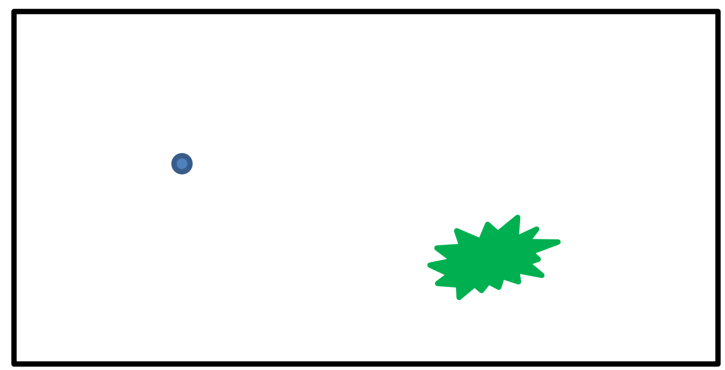
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



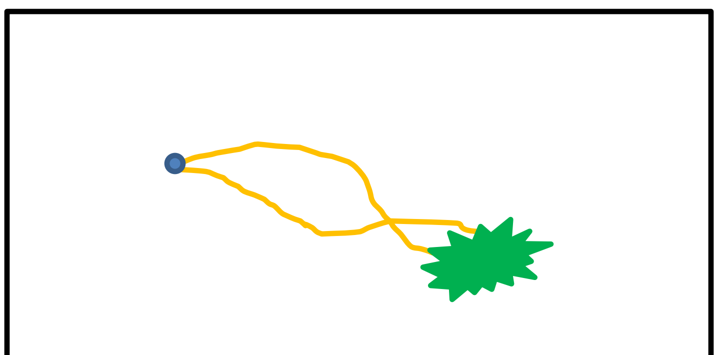
SSiPP



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)

RTDP

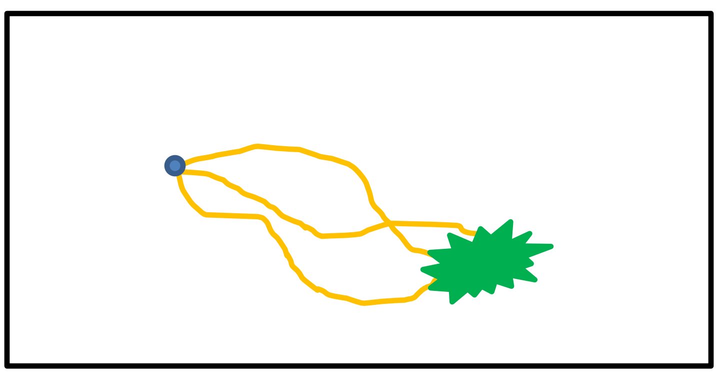


SSiPP

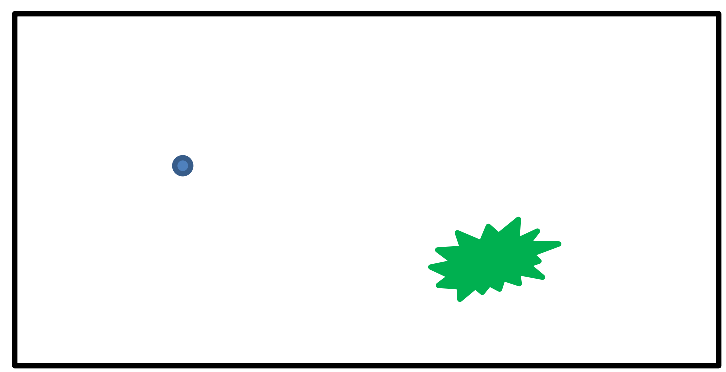
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



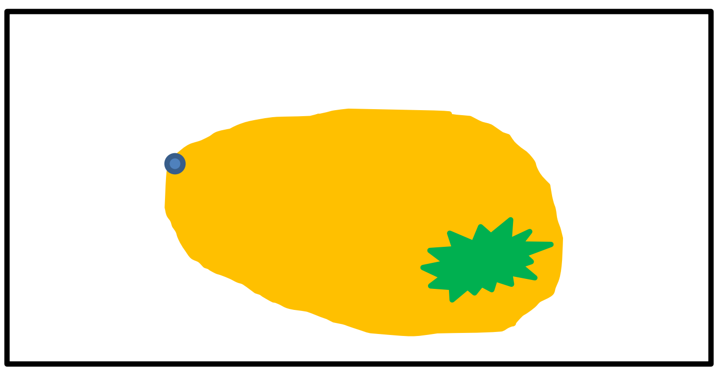
SSiPP



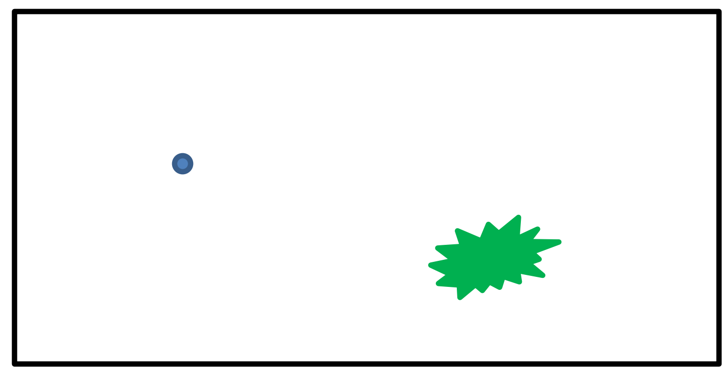
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



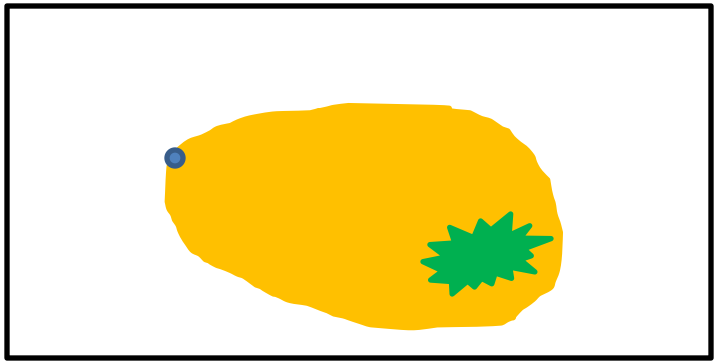
SSiPP



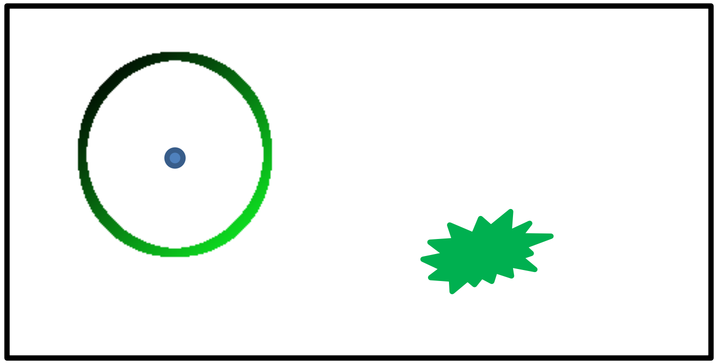
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



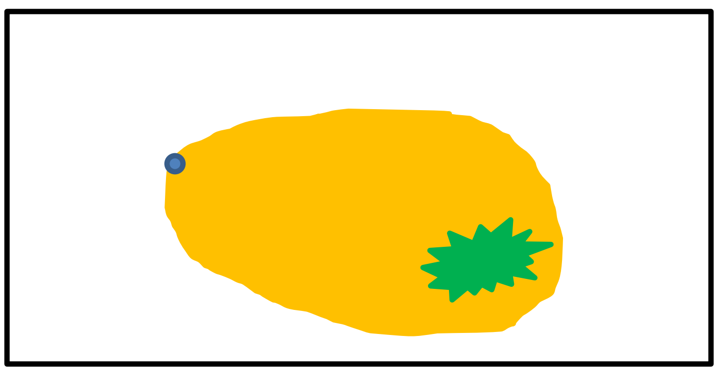
SSiPP



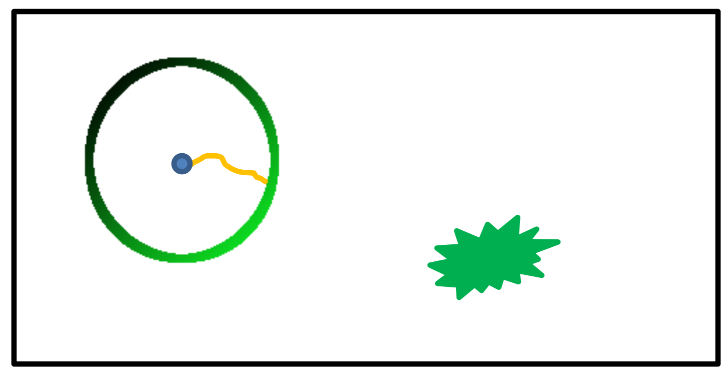
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



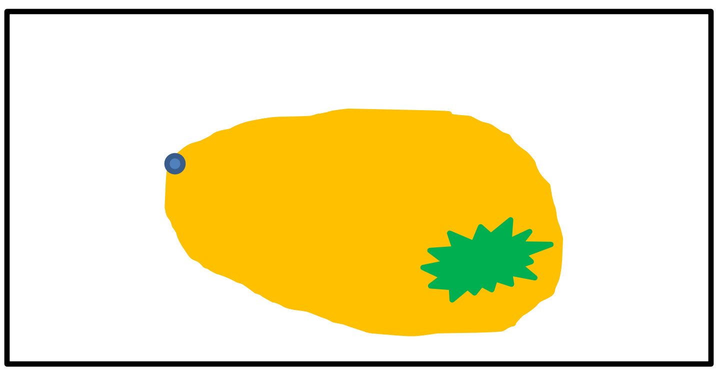
SSiPP



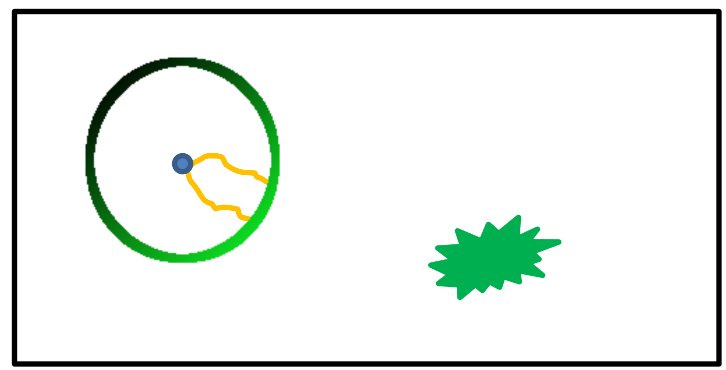
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



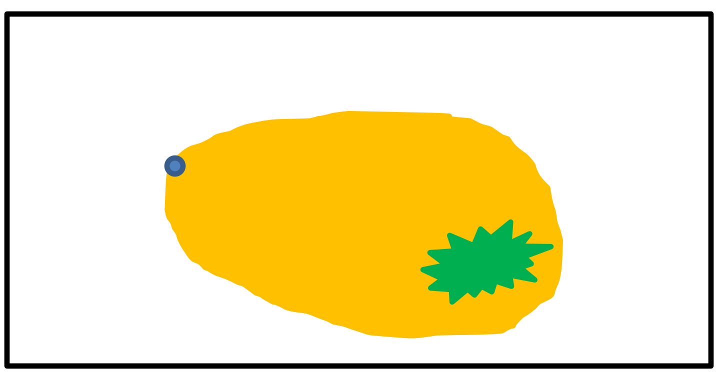
SSiPP



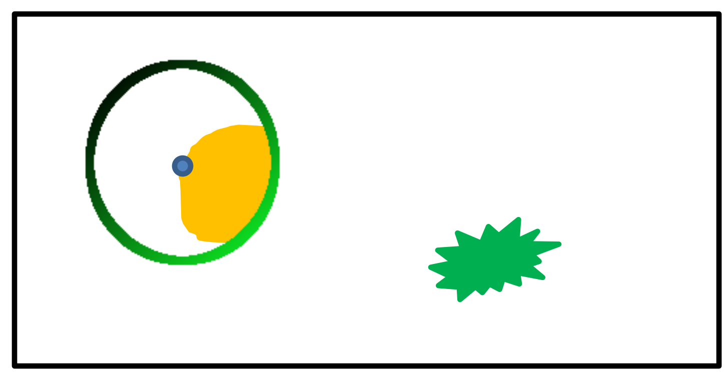
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



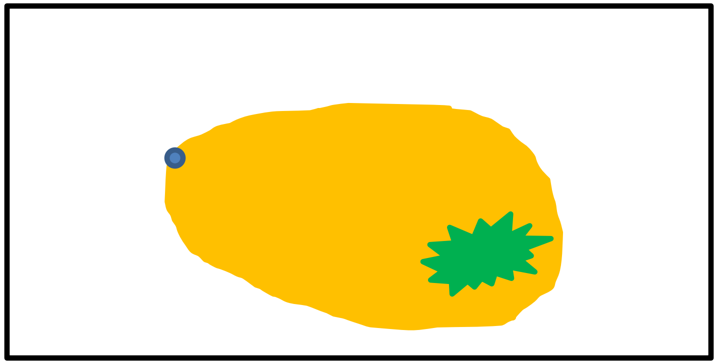
SSiPP



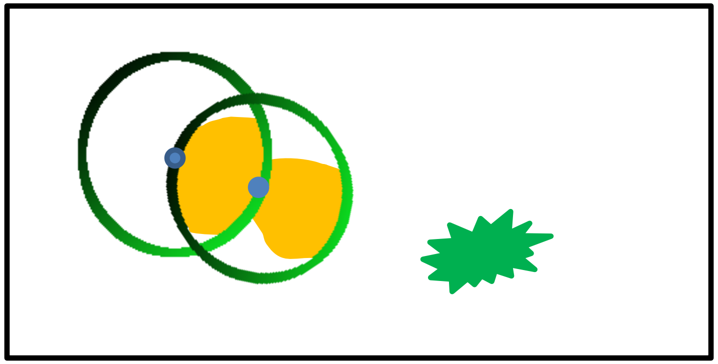
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



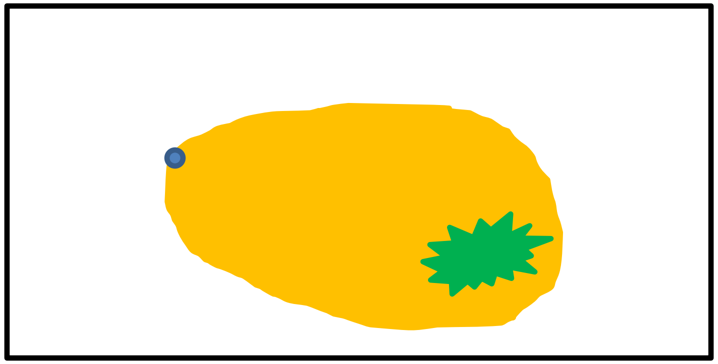
SSiPP



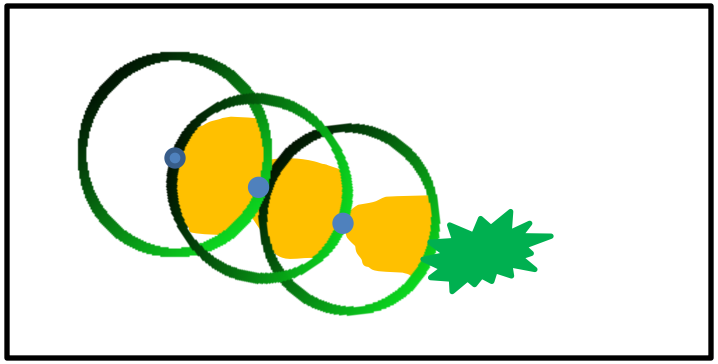
SSiPP and Real Time Dynamic Programming (RTDP)

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

RTDP



SSiPP



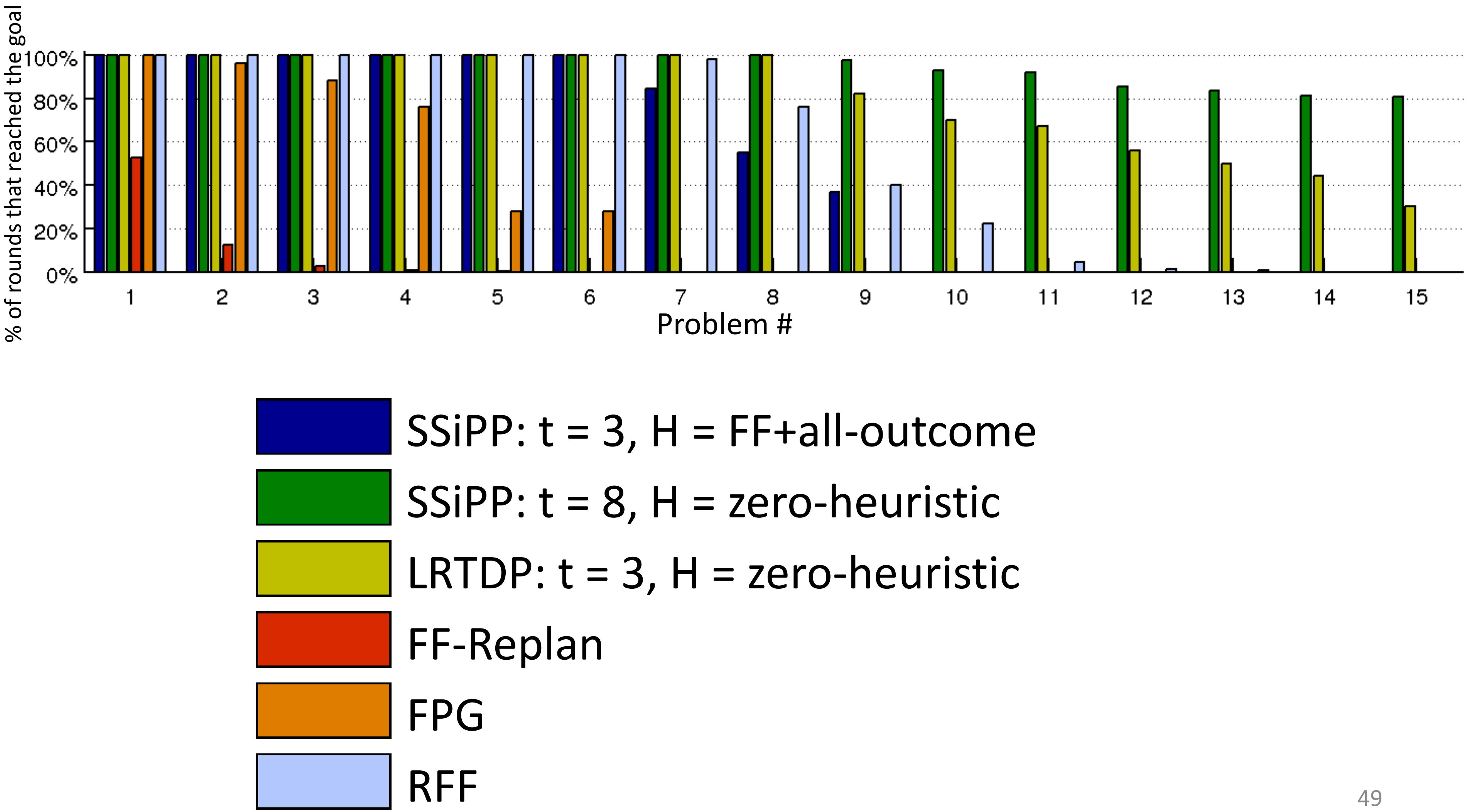
Experiments

- **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

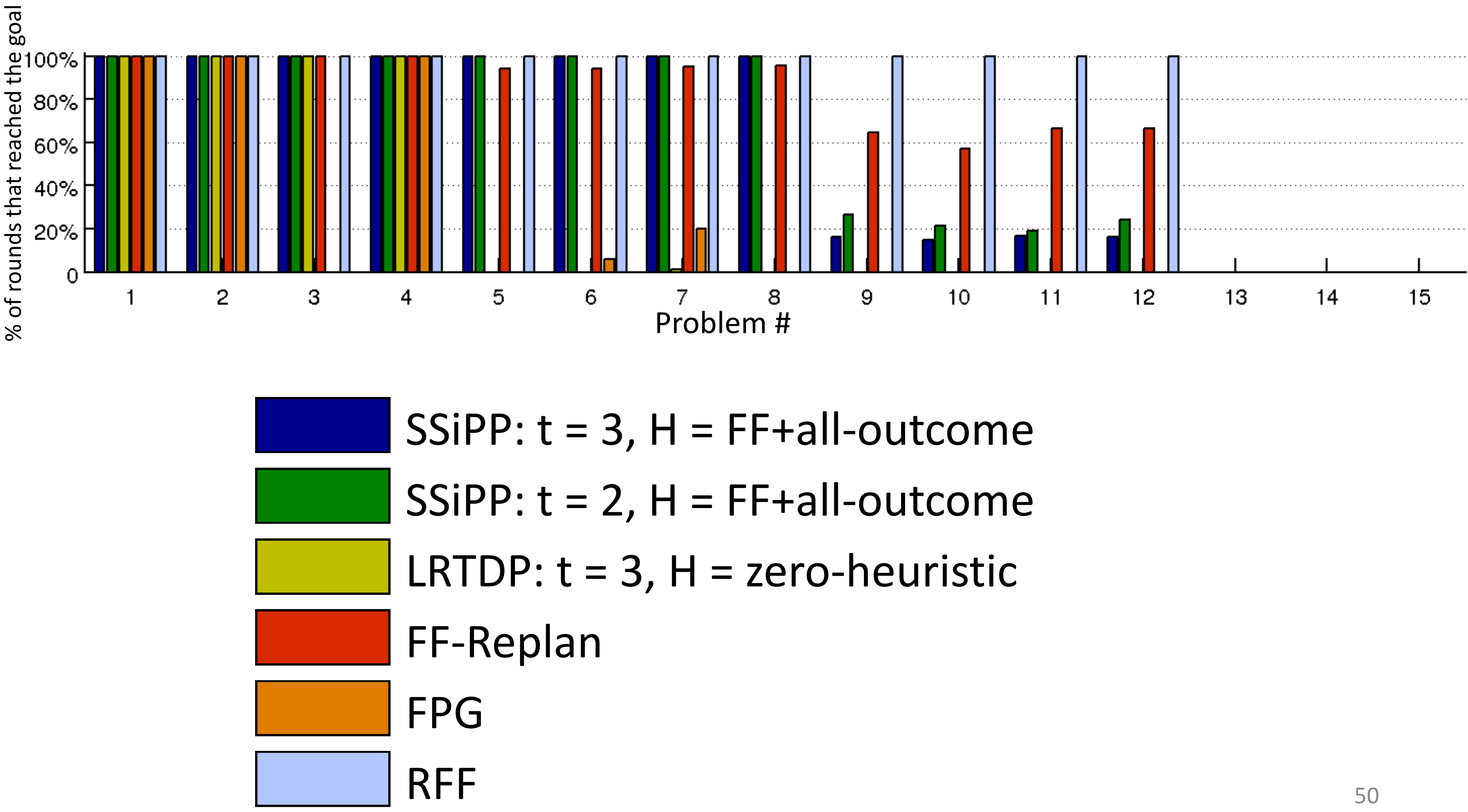
Planners

- We compare the following planners
  - **SSiPP**: using LRTDP as optimal solver
  - **LRTDP**: 2^{nd} place IPPC'04
  - **FF-Replan**: 1^{st} place IPPC'04
  - **FPG**: 1^{st} place IPPC'06
  - **RFF**: 1^{st} place IPPC'08
- Parametrizations for SSiPP and LRTDP:
  - $t \in \{1,2,3,\ldots,10\}$
  - H: zero-heuristic, FF+all-outcomes, min-min

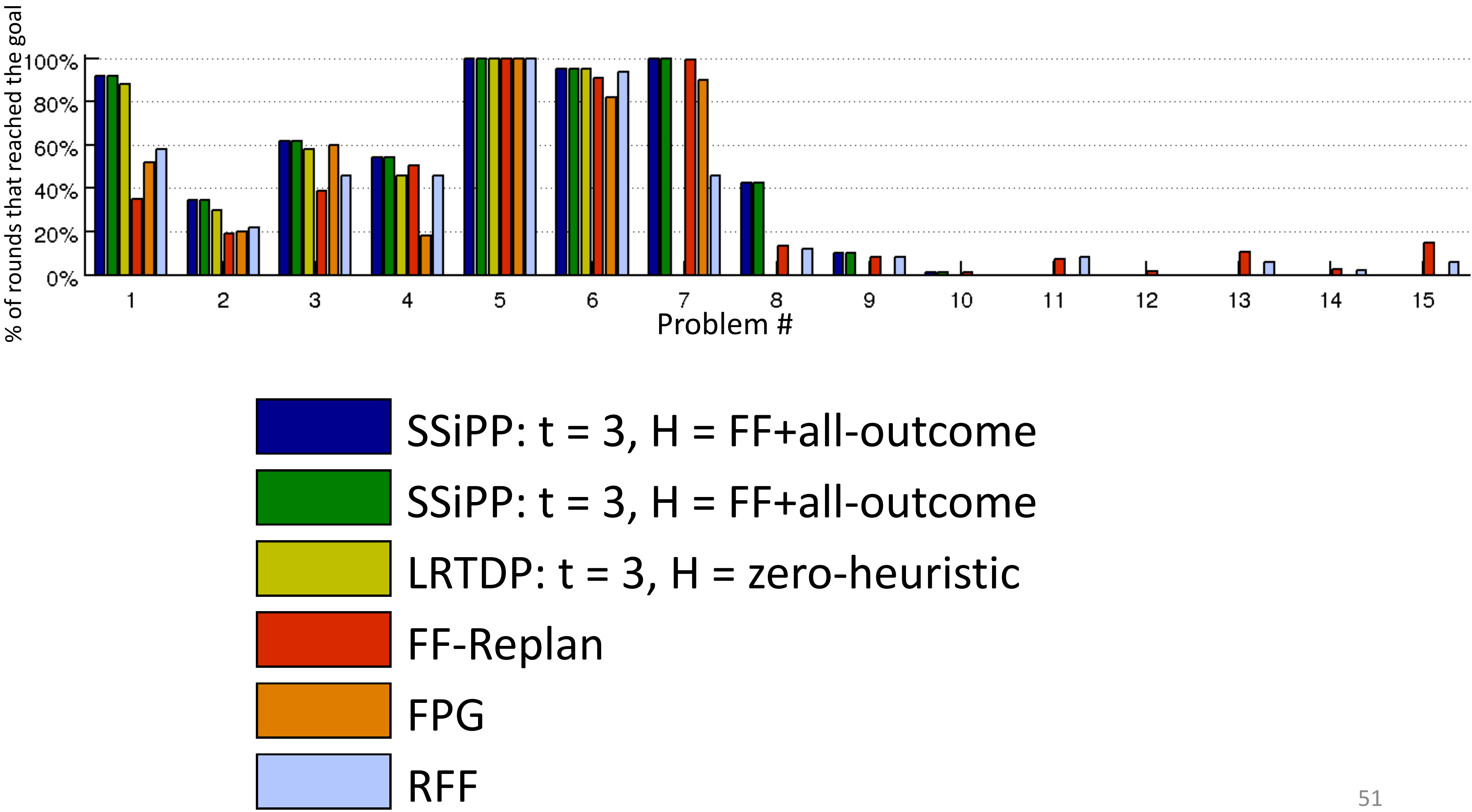
Triangle tireworld: results



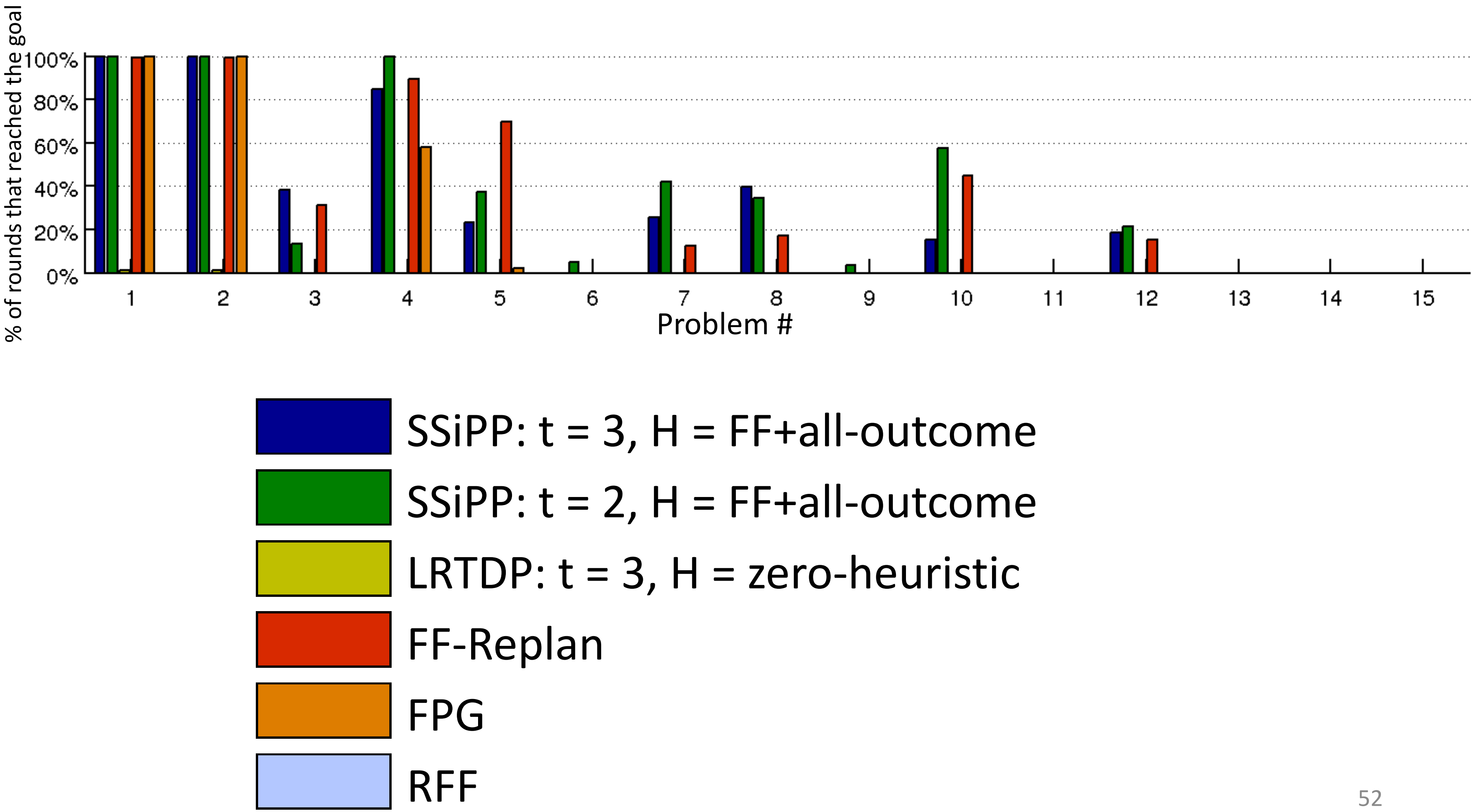
Blocks World: result



Exploding Blocks World: results



Zeno Travel: results



IPPC experiment: summary

	SSiPP Overall	SSiPP Per Domain
Outperforms all	**16.6%**	**36.6%**
Ties with the best	41.6%	53.3%

IPPC experiment: summary

	SSiPP Overall	SSiPP Per Domain
Outperforms all	**16.6%**	**36.6%**
Ties with the best	41.6%	53.3%

- In the considered problems:
  - SSiPP is never the last place in any of the problems
  - LRTDP never outperforms SSiPP

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?