Florent Teichteil-Königsbuch*, Ugur Kuter**,
Guillaume Infantes*

Incremental Plan Aggregation for Generating Policies in MDPs

May 14th, 2010

AAMAS 2010, Toronto, Canada

**ONERA-DCSD – 31055*
Toulouse Cedex 4, France

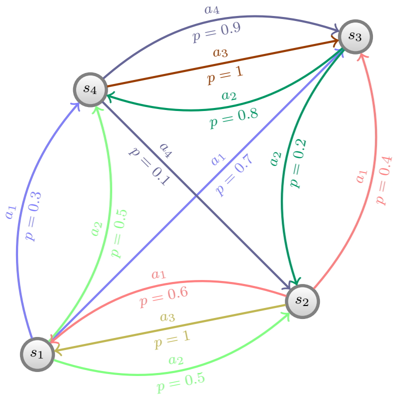
***University of Maryland,*
College Park, MD 20742, USA

Markov Decision Processes

Markov Decision Process: definition

$M = \langle \mathcal{S}, \mathcal{A}, app, Pr, R \rangle$:

- $\mathcal{S}$ and $\mathcal{A}$: finite sets of states and actions
- $app(s)$: set of all actions applicable in s
- $Pr(s, a, s')$: probability of the state transition $s \xrightarrow{a} s'$ *such that* $a \in app(s)$
- $R(s, a, s')$: reward of the state transition $s \xrightarrow{a} s'$ *such that* $a \in app(s)$



Markov Decision Process *(cont.)*

MDP planning problem

$\mathcal{P} = (M, s_0, G, \rho)$:

- $M = (S, A, app, Pr, R)$: an MDP
- $s_0 \in \mathcal{S}$: the initial state
- $G \subseteq \mathcal{S}$: set of goal states
- $0 < \rho \leqslant 1$: probability threshold

Markov Decision Process *(cont.)*

MDP planning problem

$\mathcal{P} = (M, s_0, G, \rho)$:

- $M = (S, A, app, Pr, R)$: an MDP
- $s_0 \in \mathcal{S}$: the initial state
- $G \subseteq \mathcal{S}$: set of goal states
- $0 < \rho \leqslant 1$: probability threshold

Solution π to an MDP planning problem

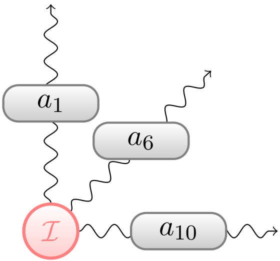
- π: partial function $S_\pi \to A$ for some set $S_\pi \subseteq S$
- $\Omega(s_0, \pi)$: probability of reaching from s_0 a state $s \notin \pi$
- $\boxed{\pi \text{ is solution of } \mathcal{P} = (M, s_0, G, \rho) \text{ iff } \Omega(s_0, \pi) < \rho}$

Forward heuristic search methods for MDPs

$\mathcal{I}$

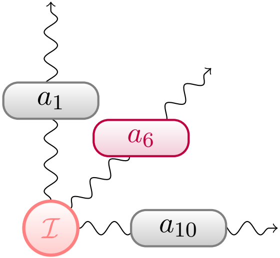
Known initial state $\mathcal{I}$

Forward heuristic search methods for MDPs



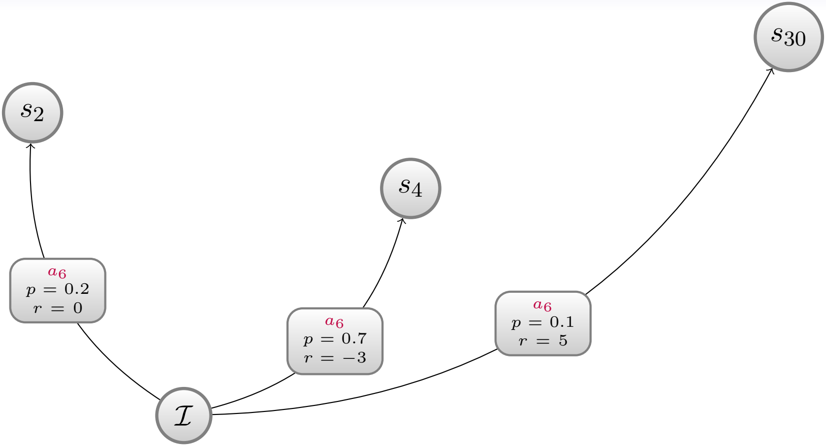
Function $\mathcal{S} \rightarrow \mathcal{A}$ indicating all applicable actions in an already instantiated state

Forward heuristic search methods for MDPs



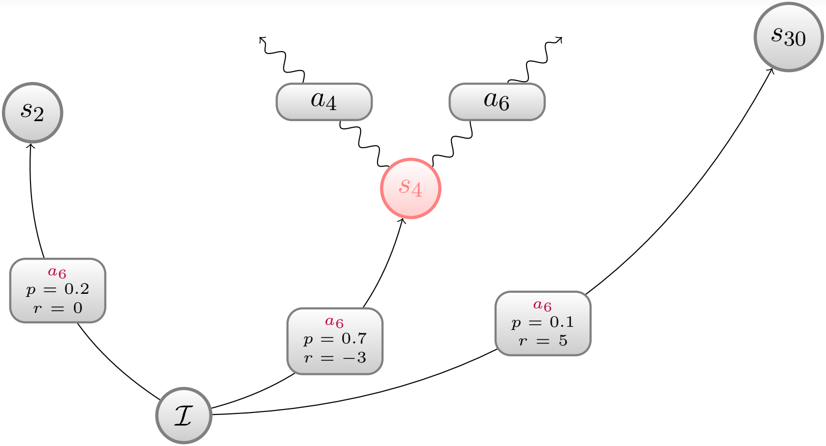
Expand action a_6 of state $\mathcal{I}$

Forward heuristic search methods for MDPs



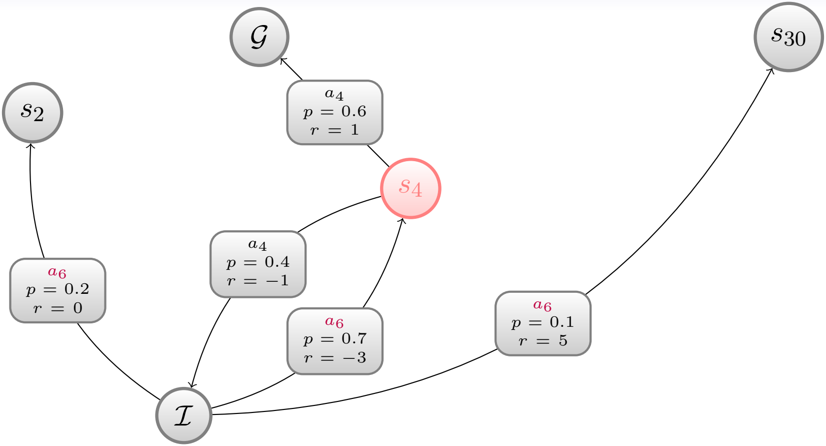
Expansion function $\mathcal{S} \times \mathcal{A} \rightarrow \mathcal{P}_r \left(2^{\mathcal{S} \times \mathbb{R}}\right)$ indicating all probabilistic successor states and rewards from a state and applying a given action

Forward heuristic search methods for MDPs



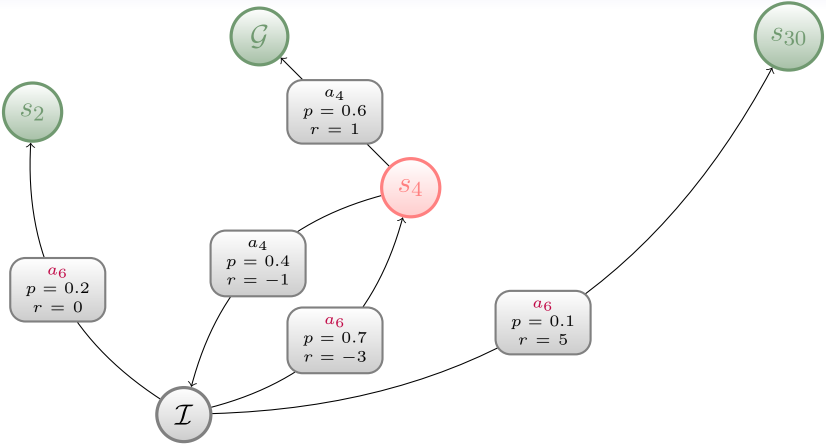
And so on from newly instantiated states...

Forward heuristic search methods for MDPs



And so on from newly instantiated states...

Forward heuristic search methods for MDPs



tip-nodes: non-expanded instantiated states

Forward heuristic search methods for MDPs *(cont.)*

- Use of a heuristic to choose the best non-expanded instantiated states to expand next
- Heuristic: means to guide the search towards the goals or the highest rewards with cheap computations
- Forward heuristic search algorithms:
  - (L)RTDP [Barto et al. 1995, Bonet & Geffner 2003]
  - LAO* [Hansen & Zilberstein 2001]
  - FPG [Buffet & Aberdeen 2007]
  - ...
- **Why not using a deterministic (classical) planner as a heuristic to expand states in the graph?**

Overview of RFF

Main idea

- Call a deterministic planner from many probabilistic reachable states
- Aggregate plans into a policy with a bounded probability of reaching its fringe at execution: $\Omega(s_0, \pi) \leqslant \rho$

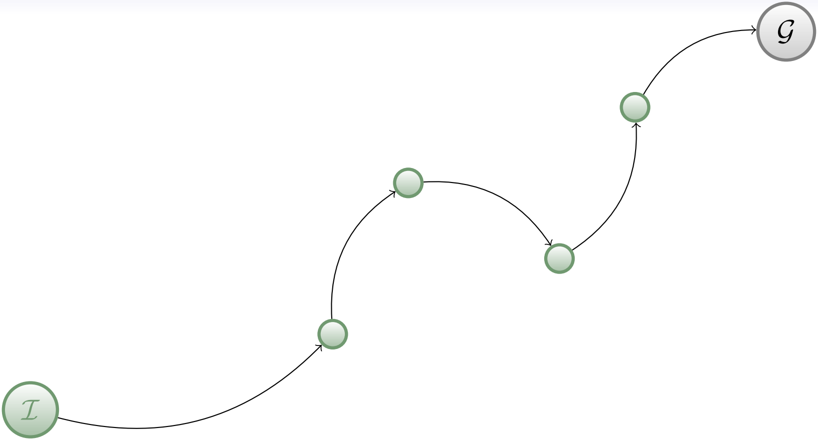
Main steps

1. Determinize the MDP planning problem
2. Generate a beam of unconditional plans with a deterministic planner
3. Aggregate all generated plans in a coherent partial policy π for the MDP
4. Compute the probability $\Omega(s_0, \pi)$ to reach the fringe of the partial policy π starting in s_0 and successively applying π
5. If $\Omega(s_0, \pi) > \rho$, then:
    1. Generate new intermediate goals for the deterministic planner
    2. Goto 2.

RFF: illustration of the algorithm

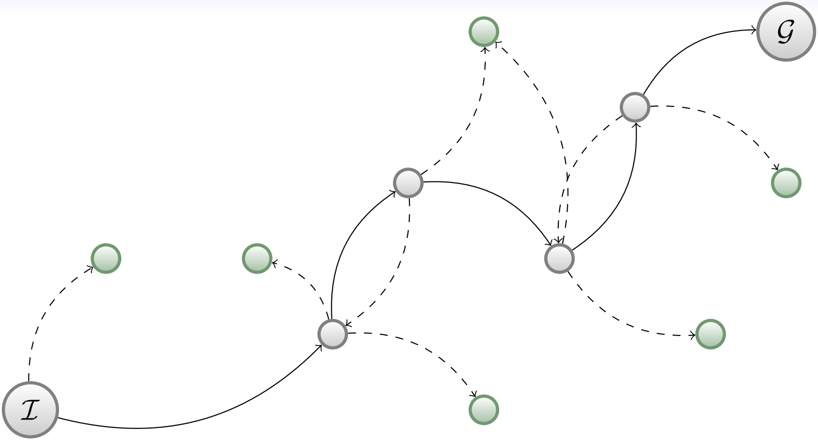
Initial input: initial state(s) $\mathcal{I}$ + goal state(s) $\mathcal{G}$ + expansion function $\mathcal{S} \times \mathcal{A} \to \mathcal{P}_r(2^{\mathcal{S}\times\mathbb{R}})$

RFF: illustration of the algorithm



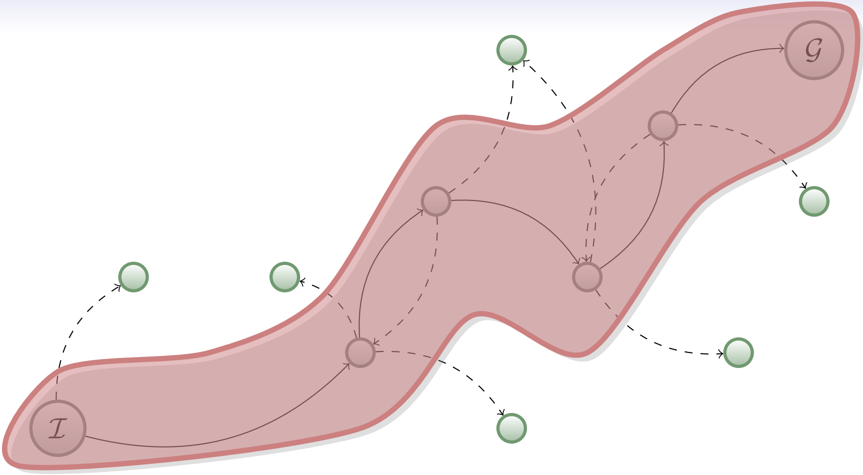
Generate an initial trajectory plan from $\mathcal{I}$ to $\mathcal{G}$ with a deterministic planner (FF) $\Rightarrow$ initial policy

RFF: illustration of the algorithm



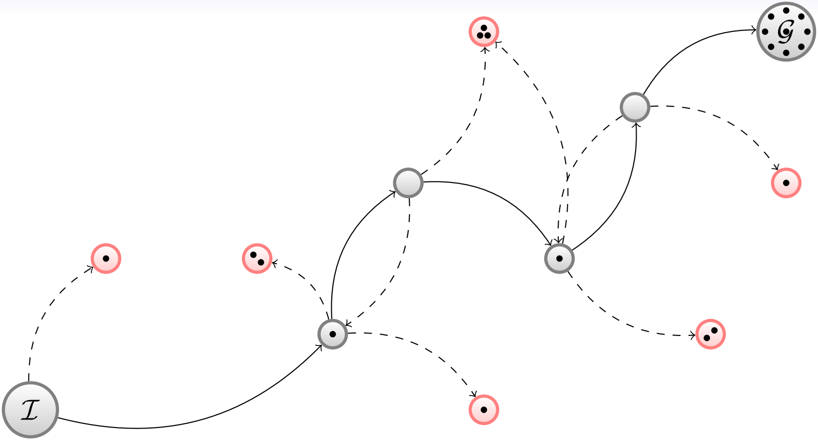
Select and expand tip-nodes by considering all probabilistic effects of each state in the graph ⇒ new tip-nodes

RFF: illustration of the algorithm



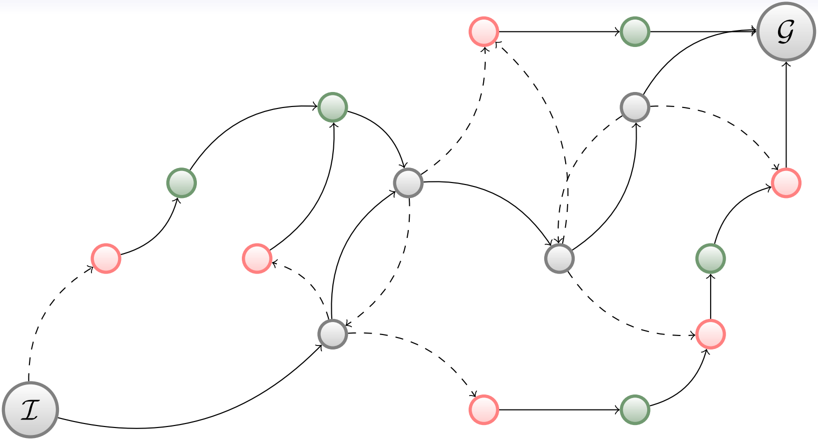
Policy reinforcement *(optional)*: shortest stochastic path from $\mathcal{I}$ to $\mathcal{G}$ on expanded nodes by considering tip-nodes as dead-ends

RFF: illustration of the algorithm



Estimate the probability of reaching any tip-node with Monte-Carlo sampling: $P_{tn} = \frac{10}{21} \approx 0.476$

RFF: illustration of the algorithm



If $P_{tn} > \rho$: generate new trajectory plans from reachable tip-nodes with the deterministic planner, and merge them in the policy graph

Computing the probability of reaching any tip-node

Fixed-point approximate computation

Let $\mathcal{T}$ be the current set of tip-nodes in the graph.
Probability P_{TN} of reaching any tip-node in $\mathcal{T}$ starting in $\mathcal{I}$ and successively applying π:

$$P_{TN} = \lim_{t \to +\infty} P_t(\mathcal{T}|\mathcal{I}) \text{ with } \begin{cases} P_0(\mathcal{T}|s) = \delta_{\mathcal{T}}(s) \\ P_t(\mathcal{T}|s) = \sum\limits_{s' \in s.successors} P(s'|s,\pi(s)) P_{t-1}(\mathcal{T}|s') \end{cases}$$

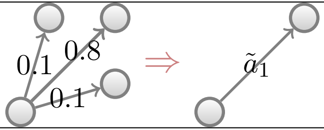
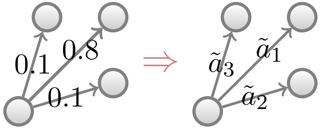
Costly $\Rightarrow$ statistic approximation with Monte-Carlo sampling

Statistic estimation with Monte-Carlo sampling

$$\begin{aligned} P_{TN} &= \frac{\text{Number of particles reaching a tip-node}}{\text{Number of thrown particles}} \\ &= \sum_{n:\text{reachable tip-nodes}} \frac{\text{Number of particles arrived in } n}{\text{Number of thrown particles}} \end{aligned}$$

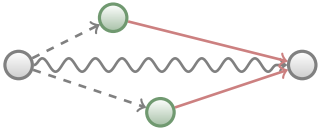
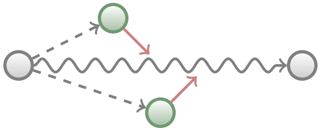
Deterministic relaxations of effects

How to transform probabilistic effects into deterministic ones?

Strategy	Pros/Cons (+/−)	Illustration
Most Probable Effect	+ few actions, find most probable paths − may not find any path	
1 effect = 1 action	+ non-zero probability of reaching $\mathcal{G}$ if paths to $\mathcal{G}$ exist − lot of actions, perhaps unlikely paths	

Goal states of the deterministic planner

How are the MDP goal states and the deterministic relaxed problem's goals related?

Strategy	Pros/Cons (+/−)	Illustration
FF goals = RFF goals	+ higher probability of reaching RFF goals, easier implementation − explore more states, larger FF computation times	
FF goals = k states with highest value explored by RFF	+ explore less states, smaller FF computation times − lower probability of reaching RFF goals, complex goals generation	

Theoretical results

Theorem 1: RFF termination

For every MDP planning problem $\mathcal{P} = (M, s_0, G, \rho)$, RFF terminates in finite time (number of iterations).

Theorem 2: probability of success

Let $\mathcal{P} = (M, s_0, G, \rho)$ be an MDP planning problem. If there are no unsolvable states in M, then the probability of success of any solution found by RFF is higher than $1 - \rho$.

Theoretical results *(cont.)*

MPO = Most Probable Outcome (effects determinization)
AO = All Outcomes (effects determinization)

Theorem 3: soundness of $\mathtt{RFF}_{\text{MPO}}$

For every MDP planning problem $\mathcal{P} = (M, s_0, G, \rho)$, every solution that $\mathtt{RFF}_{\text{MPO}}$ finds is correct.

Theorem 4: soundness of $\mathtt{RFF}_{\text{AO}}$

For every MDP planning problem $\mathcal{P} = (M, s_0, G, \rho)$, every solution that $\mathtt{RFF}_{\text{AO}}$ finds is correct.

Theorem 5: completeness of $\mathtt{RFF}_{\text{AO}}$

For every MDP planning problem $\mathcal{P} = (M, s_0, G, \rho)$, if a solution exists, it is found by $\mathtt{RFF}_{\text{AO}}$, otherwise $\mathtt{RFF}_{\text{AO}}$ returns $failure$.

Experimental results: RFF won the IPPC 2008

Team	Planners	Members	Algorithm
1	FSP*-(RBH/RDH)	Florent Teichteil Guillaume Infantes *(ONERA)*	forward heuristic search graph-based, optimal
	RFF-(BG/PG)	Ugur Kuter *(University of Maryland)*	domain determinization graph-based, plans fusion
4	LPPFF	Rajesh Kalyanam Robert Givan *(Purdue University)*	devide-and-conquer domain determinization deterministic subgoals
6	SEH	Jia-Hong Wu Rajesh Kalyanam *(Purdue University)*	domain determinization stochastic enforced hill-climbing local MDPs to escape basins
9	HMDPP	Emil Keyder *(Universitat Pompeu Fabra)* Hector Geffner *(ICREA & UPF)*	domain determinization self-loop relaxation heuristic pattern database heuristic lexicographic heuristic choice
11	FF-Replan	Sungwook Yoon *(Palo Alto Research Center)*	domain determinization plan & repair

Official results available in the competition booklet

Varying the probabilistic threshold ρ

- **Number of calls to `RFF` increases with ρ (= upper bound on the probability to replan)**
- Percentage of problems solved: *no general impact*
- Quality of solutions: *no general impact*
- Total time (all calls to `RFF` per problem + simulation): *no general impact*

Explanation

If the policy fails, we have to recompute a policy (= call to `RFF`) from the failure state ; and the failure probability of policies increases with ρ.

RFF with different method for relaxations of MDPs: Most Probable Outcome (MPO), and All Outcomes (AO)

- **Percentage of problems solved:** MPO $>$ AO
- **Quality of solutions:** MPO $>$ AO
- Total time (all calls to RFF per problem + simulation): *no general impact*

Explanation

The action makespan of the deterministic domain is larger with AO than with MPO. Policies generated with MPO are more likely to fail than the ones generated with AO, but MPO solves far more problem instances than AO.

RFF using the goal-selection strategies ProblemGoals (PG), Random Goals (RG), and Best Goals (BG)

- **Percentage of problems solved: RG is the best for `blocksworld` and `boxworld`**, no general impact on other domains
- **Quality of solutions: PG is the best for `(ex-)blocksworld`**, no general impact on other domains
- **Total time: RG is the best for `boxworld` and PG is the best for `(ex-)blocksworld`**, no general impact on other domains

Explanation

The goal-selection strategy impacts the way states are explored, so that its consequences in terms of solution quality highly depends on the domain.

Conclusions

- RFF is a **new MDP planner that uses a deterministic subplanner to generate policies that are robust to effect uncertainties**. RFF:
  - determinizes the given MDP model into a classical planning problem;
  - generates partial policies off-line by producing solution plans to the classical planning problem and incrementally aggregating them into a policy;
  - uses sequential Monte-Carlo (MC) simulations of the partial policies before execution, in order to assess the probability of replanning for a policy.
- **RFF generates policies whose probability of success is below a given threshold**
- the deterministic planner can be viewed as a **heuristic to explore new states** in the graph
- special use-case: $\texttt{RFF}(\rho = 1) \equiv \texttt{FF-replan}$ [Yoon et al. 2007]

Future work

- Use different deterministic planners and compare how they affect the aggregated policy
- Use different plan aggregation techniques (between merged policy optimization and action rewrite)
- Use different goal selection strategies
- Use different determinization strategies
- "Agressive" parallelization of calls to the deterministic planner using multi-core processors
- Is optimality achievable?
- Extensions:
  - Hybrid MDPs (discrete and continuous variables)
  - temporal planning: SMDPs and GSMDPs
  - partial observability: POMDPs

Questions?

The authors thank the anonymous reviewers for their helpful comments on the paper.

This research was supported by the French "Délégation Générale pour l'Armement" grant 07.60.031.00.470.75.01, DARPA's Transfer Learning and Integrated Learning programs, and NSF grant IIS0412812. The opinions in this paper are those of the authors and do not necessarily reflect those of the funders.