Experience feedback about asynchonous policy iteration and observable time MDPs

Abstract

Representing time-dependency and temporal interactions in stochastic decision processes raises many questions, both from the modeling and the resolution points of view. In this talk, I will try to provide some feedback from my personal experience on these two topics. Several different options in the MDP (and related) literature have been adopted to model temporal stochastic decision processes. By focusing on the problems of time-dependency and concurrency, I will explain why Generalized Semi-Markov Decision Processes (GSMDPs) are a natural way of modeling temporal problems. In particular, we will point out an interesting link with Partially Observable MDPs which will emphasize the complexity of their resolution. Then, from the resolution point of view, I will introduce a methodology based on Asynchronous Policy Iteration and direct utility estimation designed for observable-time GSMDPs. Based on the experience feedback of this work, we will emphasize ideas concerning local value functions and policies and relate them to some recent advances in machine learning.

Experience feedback about asynchonous policy iteration and observable time MDPs

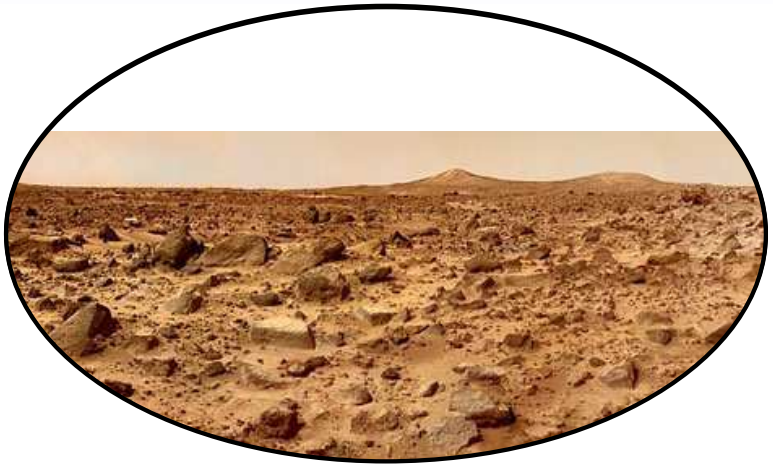
Emmanuel Rachelson

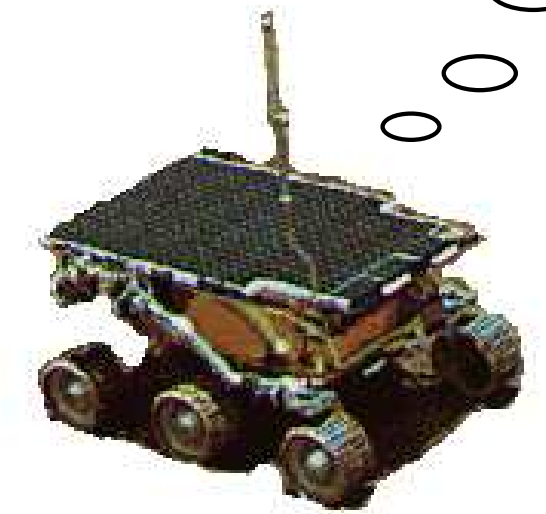
Technical University of Crete, Chania, Greece
(formerly ONERA, Toulouse, France)

May 29th 2009

Motivation

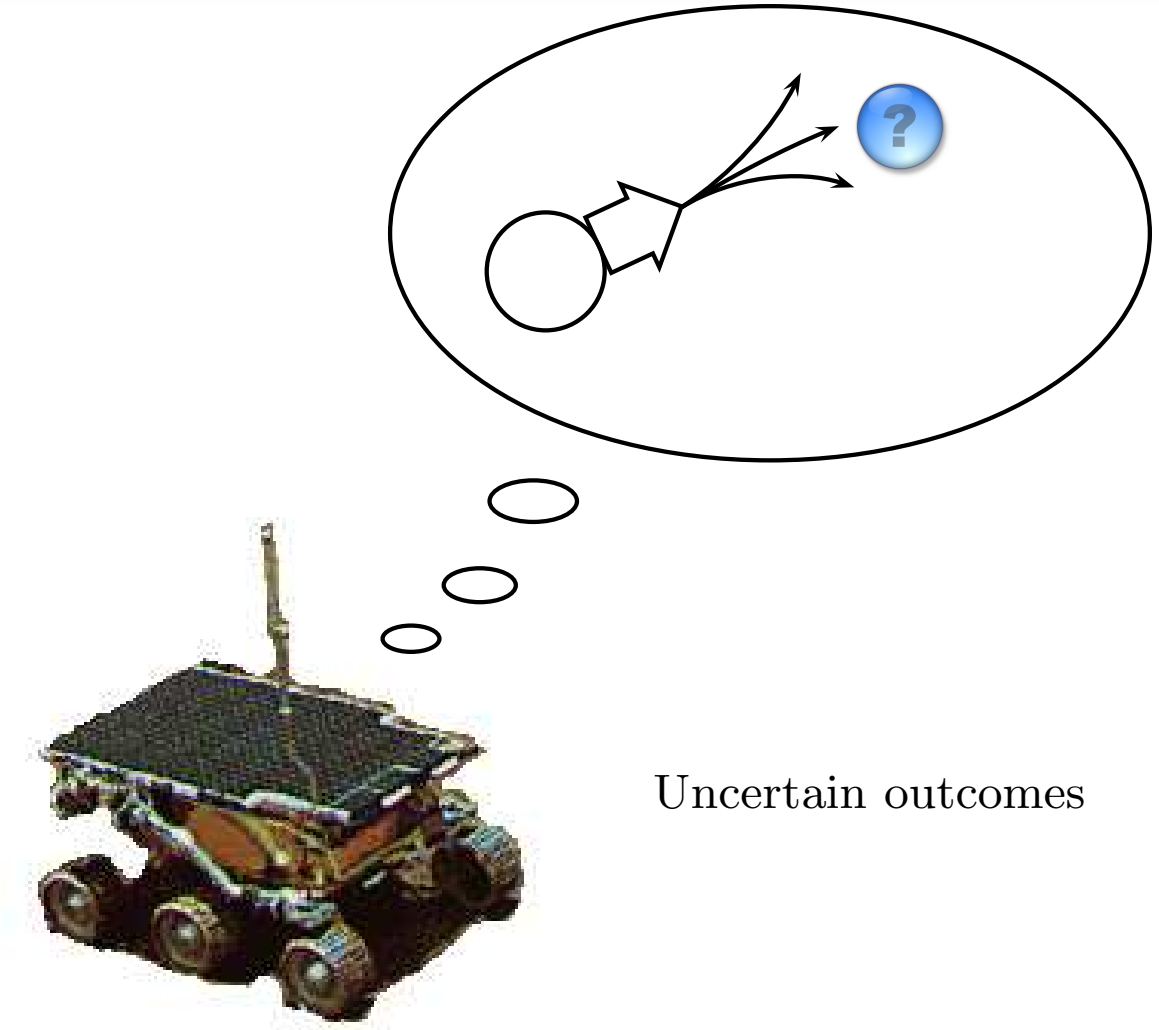
Motivation





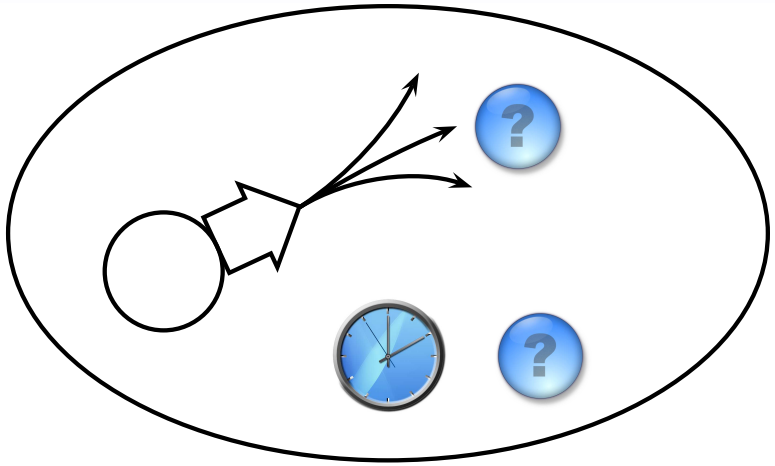
Performing
"as well as possible"

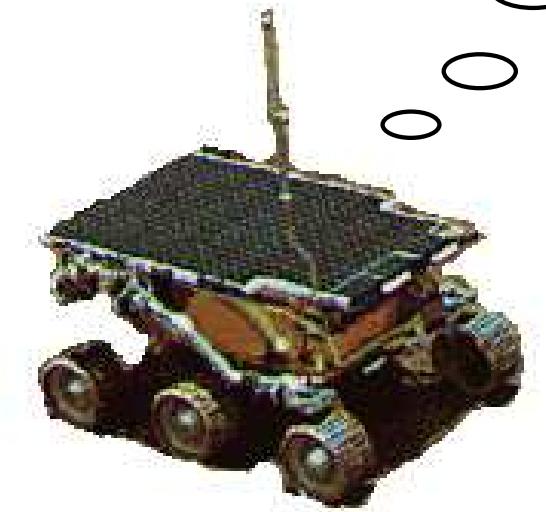
Motivation



Uncertain outcomes

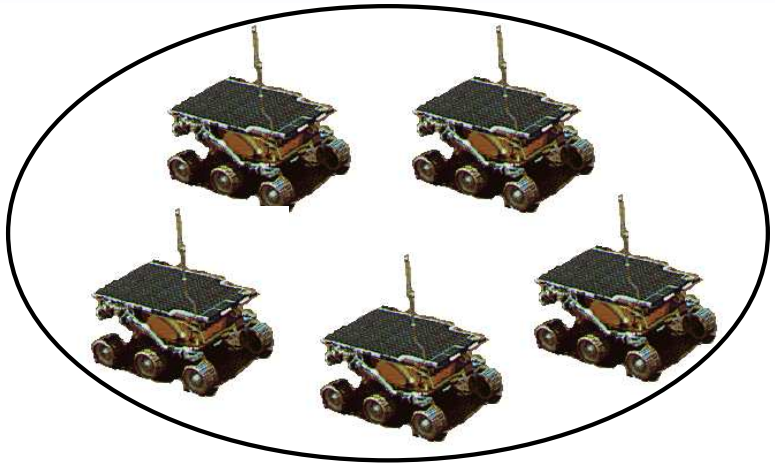
Motivation

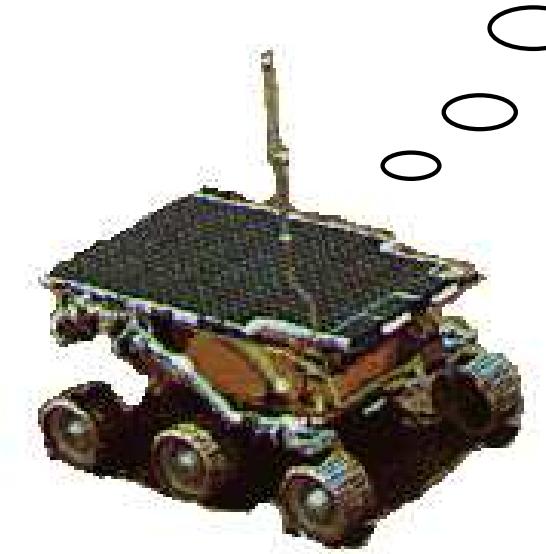




Uncertain durations

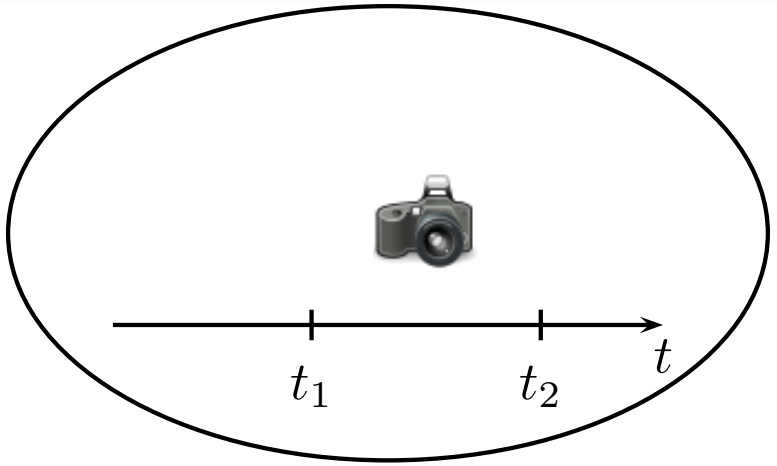
Motivation

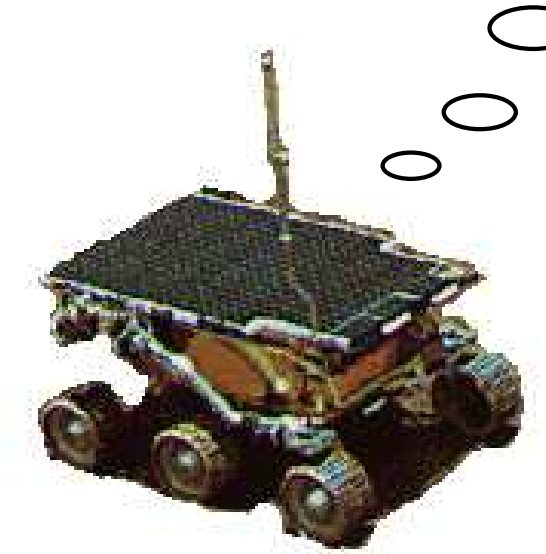




Time-dependent environment

Motivation





Time-dependent goals and rewards

Problem statement

We want to build a **control policy**
which allows the agent to **coordinate** its **durative** actions
with the **continuous evolution** of its **uncertain** environment
in order to **optimize** its behaviour w.r.t. a given criterion.

Problem statement

We want to build a **control policy**
which allows the agent to **coordinate** its **durative** actions
with the **continuous evolution** of its **uncertain** environment
in order to **optimize** its behaviour w.r.t. a given criterion.

Problem statement

We want to build a **control policy**
which allows the agent to **coordinate** its **durative** actions
with the **continuous evolution** of its **uncertain** environment
in order to **optimize** its behaviour w.r.t. a given criterion.

Problem statement

We want to build a **control policy**
which allows the agent to **coordinate** its **durative** actions
with the **continuous evolution** of its **uncertain** environment
in order to **optimize** its behaviour w.r.t. a given criterion.

Outline

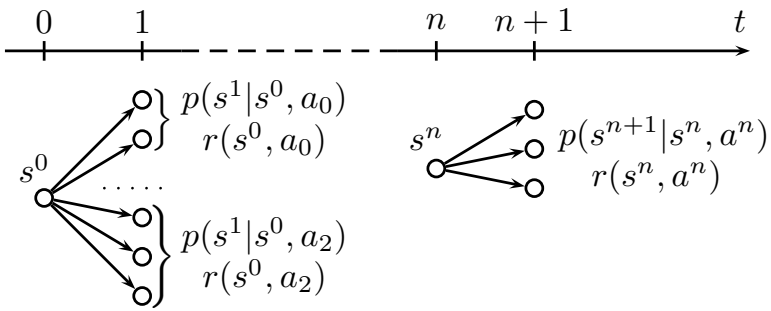
Modeling background

Sequential decision under probabilistic uncertainty:

Markov Decision Process

Tuple $\langle S, A, p, r, T \rangle$
Markovian transition model $p(s'|s,a)$
Reward model $r(s,a)$
T is a set of timed decision epochs $\{0, 1, \ldots, H\}$

Infinite (unbounded) horizon: $H \longrightarrow \infty$



Optimal policies for MDPs

Value of a sequence of actions

$$\forall (a_n) \in A^{\mathbb{N}}, V^{(a_n)}(s) = E\left(\sum_{\delta=0}^{\infty} \gamma^{\delta} r(s^{\delta}, a_{\delta})\right)$$

Stationary, deterministic, Markovian policy

$$\mathscr{D} = \left\{ \pi \ : \ \begin{array}{ccc} S & \rightarrow & A \\ s & \mapsto & \pi(s) = a \end{array} \right\}$$

Optimality equation

$$V^*(s) = \max_{\pi \in \mathscr{D}} V^{\pi}(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \right\}$$

Continuous durations in stochastic processes

MDPs: the set T contains integer-valued dates.

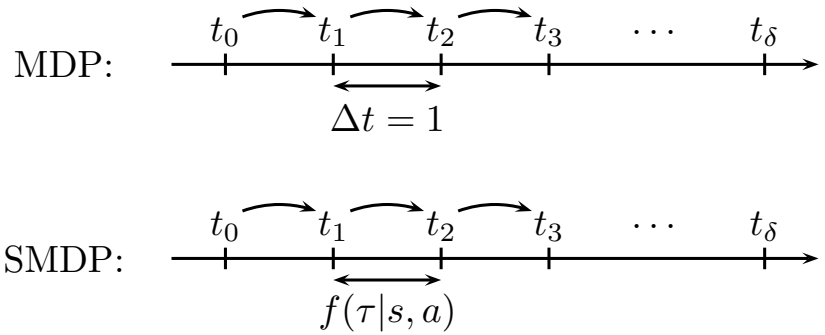
$\longrightarrow$ more flexible durations?

Semi-Markov Decision Process

Tuple $\langle S, A, p, f, r \rangle$

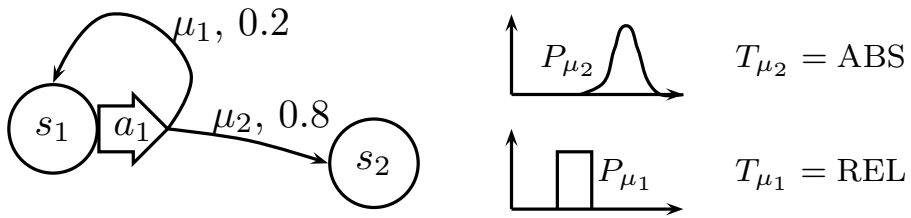
Duration model $f(\tau|s,a)$

Transition model $p(s'|s,a)$ or $p(s'|s,a,\tau)$



Modeling time-dependency in MDPs

- Many (more or less) related formalisms.
- TiMDPs [Boyan and Littman, 2001]



- Existence of an optimality equation [Rachelson et al., 2008a])
- Exact and approximate methods for solving TiMDPs (and beyond) [Feng et al., 2004, Li and Littman, 2005, Rachelson et al., 2009].

Is that sufficient?

"A well-cast problem is a half-solved problem."

Initial example: obtaining the model is not trivial.

$\longrightarrow$ the "first half" (modeling) is not solved.

A natural model for continuous-time decision processes?

Is that sufficient?

"A well-cast problem is a half-solved problem."

Initial example: obtaining the model is not trivial.

$\longrightarrow$ the "first half" (modeling) is not solved.

A natural model for continuous-time decision processes?

Is that sufficient?

"A well-cast problem is a half-solved problem."

Initial example: obtaining the model is not trivial.

$\longrightarrow$ the "first half" (modeling) is not solved.

A natural model for continuous-time decision processes?

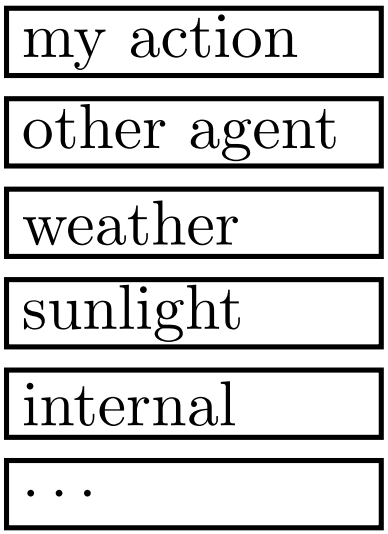
Concurrent exogeneous events

Explicit-event modeling:
a natural description of the systems complexity.

Concurrent exogeneous events

Explicit-event modeling:
a natural description of the systems complexity.

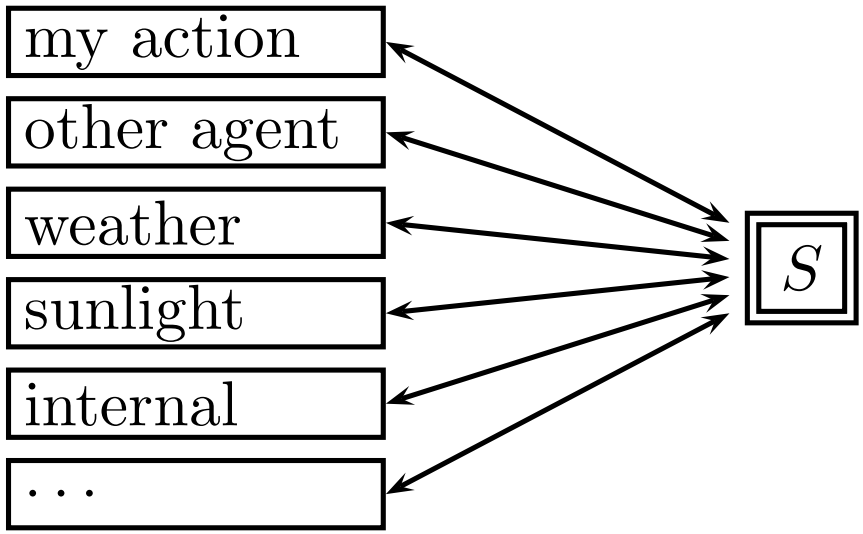
Aggregating the contribution of concurrent temporal processes. . .



Concurrent exogeneous events

Explicit-event modeling:
a natural description of the systems complexity.

Aggregating the contribution of concurrent temporal processes. . .



. . . all affecting the same state space

GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

Glynn, P. (1989).
A GSMP Formalism for Discrete Event Systems.
Proc. of the IEEE, 77.

Younes, H. L. S. and Simmons, R. G. (2004).
Solving Generalized Semi-Markov Decision Processes using Continuous Phase-Type Distributions.
In *AAAI Conference on Artificial Intelligence*.

GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

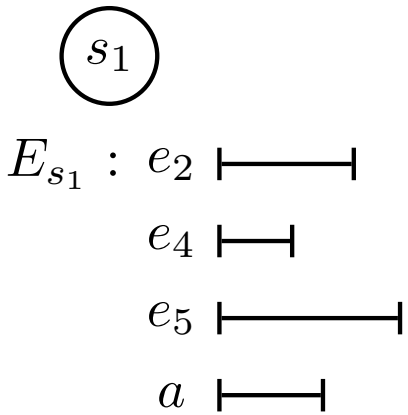
E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

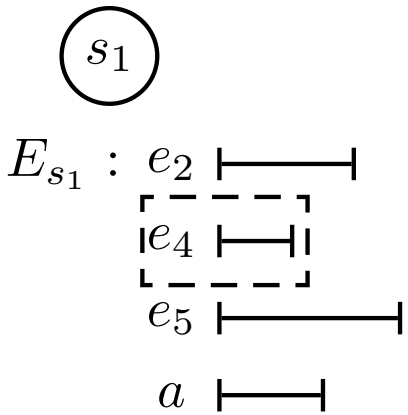


GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

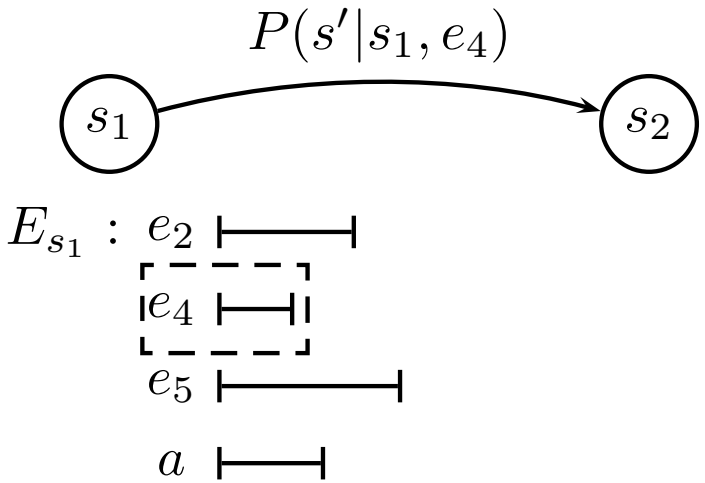


GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

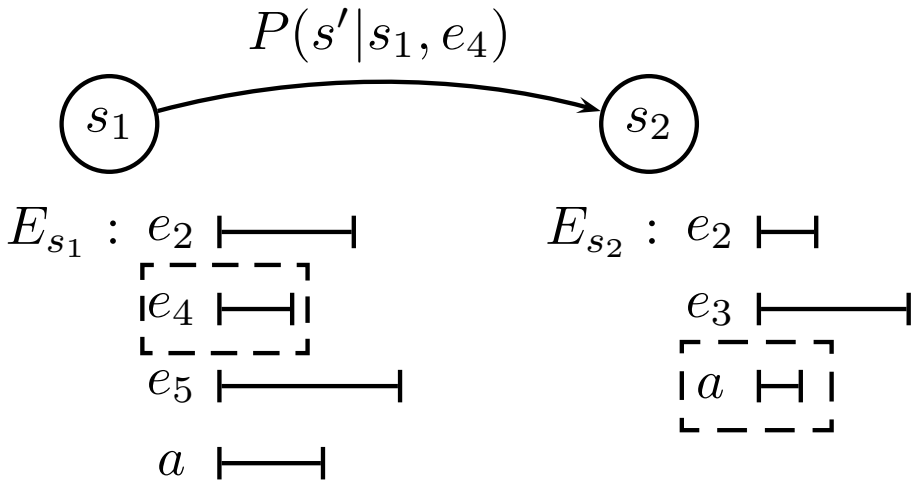


GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

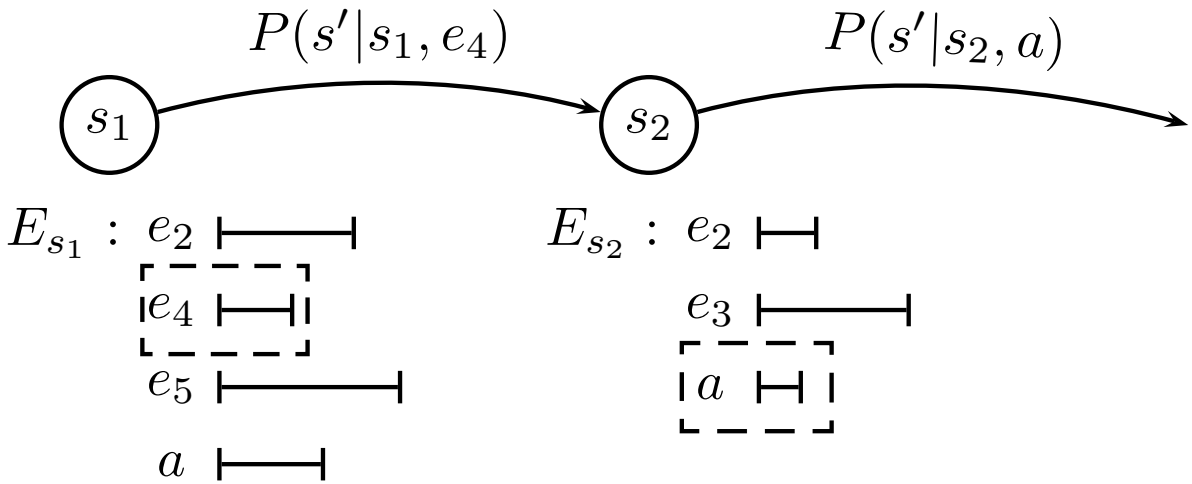


GSMDPs

Generalized Semi-Markov Decision Process

Tuple $\langle S, E, A, p, f, r \rangle$

E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.



Modeling claim

A natural model for temporal processes

Observable time GSMDPs are a natural way of modeling stochastic, temporal decision processes.

Properties

Markov property

The process defined by the **natural state** s of a GSMDP does not retain Markov's property.

No guarantee of an optimal $\pi(s)$ policy.

Markovian state: (s, c)
$\longrightarrow$ often non-observable.

Properties

Working hypothesis

In time-dependent GSMDPs, the state (s, t) is a good approximation of the Markovian state variables (s, c).

GSMDPs and POMDPs

Observations and hidden process

The natural state s of a GSMDP corresponds to observations on a hidden Markov process (s, c).

$$\begin{cases} (s,c) & \leftrightarrow & \text{hidden state} \\ s & \leftrightarrow & \text{observations} \end{cases}$$

Working hypothesis

In time-dependent GSMDPs, the state (s, t) is a good approximation of the associated POMDP's belief state.

From temporal modeling to temporal resolution

- Large, hybrid, metric state spaces.
- Many concurrent events.
- Long, time-bounded episodes.
- Example: subway problem.

From temporal modeling to temporal resolution

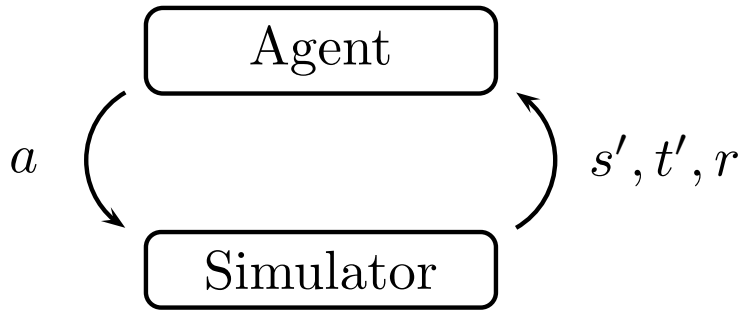
Remark

Even though GSMDPs are non-Markov processes, they provide a straightforward way of building a simulator.

How can we search for a good policy?

$\longrightarrow$ Learning from the interaction with a GSMDP simulator.

Learning from interaction with a simulator



Planning: using model $\begin{cases} P(s', t'|s, t, a) \\ r(s, t, a) \end{cases}$ $\searrow$

Learning: using samples (s, t, a, r, s', t') $\nearrow$

to get good $\begin{cases} V(s, t) \\ \pi(s, t) \end{cases}$

Simulation-based Reinforcement Learning

3 main issues:

- Exploration of the state space
- Update of the value function
- Improvement of the policy

How should we use our temporal process' simulator to learn policies?

Simulation-based Reinforcement Learning

3 main issues:

- Exploration of the state space
- Update of the value function
- Improvement of the policy

How should we use our temporal process' simulator to learn policies?

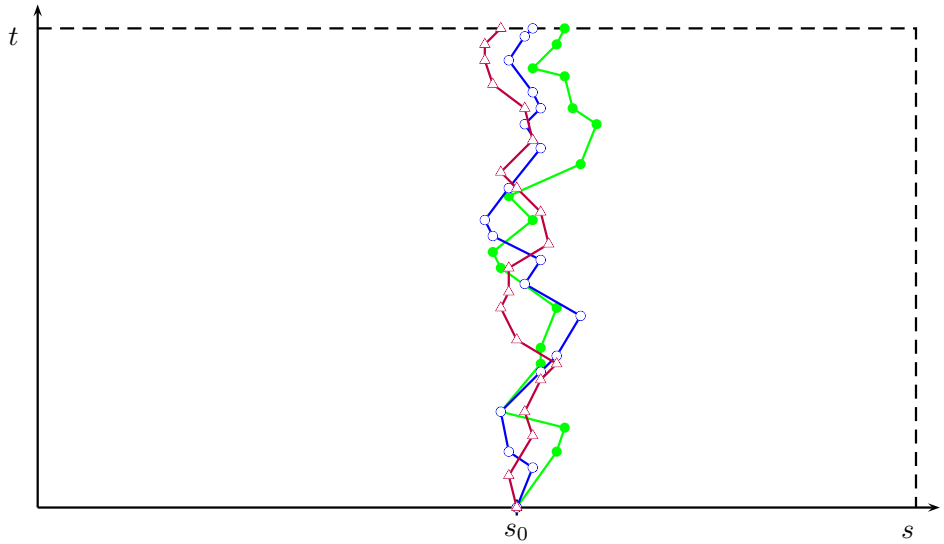
Intuition

Our (lazy) approach

Improve the policy in the situations which are likely to be encountered.
Evaluate the policy in the situations needed for improvement.

Exploiting info from episodes?

episode = observed simulated trajectory through the state space.



Model-free, simulation-based local search

Input initial state s_0, t_0,
initial policy π_0,
process simulator.

Goal improve on π_0

"simulator" $\longrightarrow$ simulation-based

Model-free, simulation-based local search

Input initial state s_0, t_0,
initial policy π_0,
process simulator.

Goal improve on π_0

"simulator" $\longrightarrow$ simulation-based
"local" $\longrightarrow$ asynchronous

Model-free, simulation-based local search

Input initial state s_0, t_0,
initial policy π_0,
process simulator.

Goal improve on π_0

"simulator"	$\longrightarrow$	simulation-based
"local"	$\longrightarrow$	asynchronous
"incremental π improvement"	$\longrightarrow$	policy iteration

Model-free, simulation-based local search

Input initial state s_0, t_0,
initial policy π_0,
process simulator.

Goal improve on π_0

"simulator"	$\longrightarrow$	simulation-based
"local"	$\longrightarrow$	asynchronous
"incremental π improvement"	$\longrightarrow$	policy iteration

Model-free, simulation-based local search

Input initial state s_0, t_0,
initial policy π_0,
process simulator.

Goal improve on π_0

"simulator"	$\longrightarrow$	simulation-based
"local"	$\longrightarrow$	asynchronous
"incremental π improvement"	$\longrightarrow$	policy iteration

for temporal problems:

iATPI

Asynchronous Dynamic Programming

Asynchronous Bellman backups

As long as every state is visited infinitely often for Bellman backups on V or π, the sequences of V_n and π_n converge to V^* and π^*.
$\longrightarrow$ Asynchronous Policy Iteration.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996).
Neuro-Dynamic Programming.
Athena Scientific.

iATPI performs greedy exploration

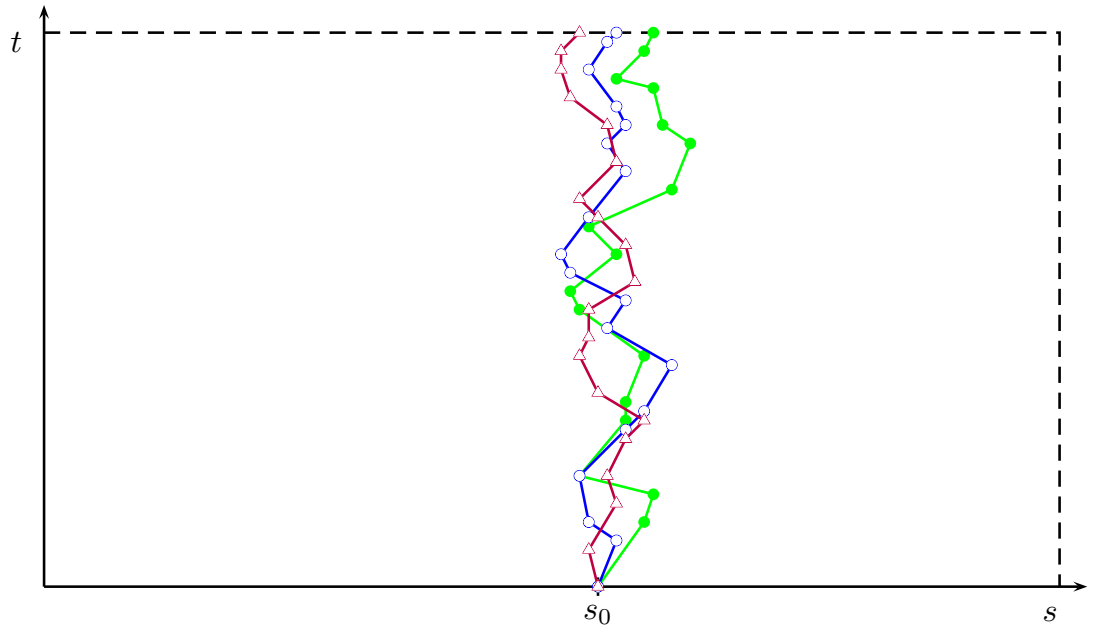
Once an improving action a is found in (s, t), the next state (s', t') picked for Bellman backup is chosen by applying a.
Observable time $\Rightarrow$ this (s', t') is picked according to $P(s', t' | s, t, \pi_n)$.

Monte Carlo evaluations for temporal problems

Simulating π in (s, t)

$$\Downarrow$$

$$\left\{ \left((s_0, t_0), a_0, r_0, \ldots, (s_{l-1}, t_{t-1}), a_{l-1}, r_{l-1}, (s_l, t_l)\right) \;\middle|\; \begin{array}{l} (s_0, t_0) = (s, t) \\ a_i = \pi(s_i, t_i) \\ t_l \geq T \end{array} \right\}$$



Monte Carlo evaluations for temporal problems

Simulating π in (s,t)

$$\Downarrow$$

$$\left\{ \left((s_0,t_0),a_0,r_0,\ldots,(s_{l-1},t_{t-1}),a_{l-1},r_{l-1},(s_l,t_l)\right) \;\middle|\; \begin{array}{l} (s_0,t_0)=(s,t) \\ a_i=\pi(s_i,t_i) \\ t_l \geq T \end{array} \right\}$$

$$\Downarrow$$

$$\mathit{ValueSet} = \left\{ \tilde{R}(s_i,t_i) = \sum_{k=i}^{l-1} r_i \right\}$$

Monte Carlo evaluations for temporal problems

$$\text{Simulating } \pi \text{ in } (s,t)$$

$$\Downarrow$$

$$\left\{ \left((s_0,t_0),a_0,r_0,\ldots,(s_{l-1},t_{t-1}),a_{l-1},r_{l-1},(s_l,t_l)\right) \;\middle|\; \begin{array}{l}(s_0,t_0)=(s,t)\\ a_i=\pi(s_i,t_i)\\ t_l \geq T\end{array} \right\}$$

$$\Downarrow$$

$$\mathit{ValueSet} = \left\{ \tilde{R}(s_i,t_i) = \sum_{k=i}^{l-1} r_i \right\}$$

Value function estimation

$$V^\pi(s,t) = E(R(s,t))$$
$$\tilde{V}^\pi \leftarrow \mathit{regression}(\mathit{ValueSet})$$

In practice

Algorithm sketch

Given the current policy π_n,
the current process state (s, t),
the current estimate $\tilde{V}^{\pi_n}$

Compute the best action a^* with respect to $\tilde{V}^{\pi_n}$

Pick (s', t') according to a^*

Until $t' > T$

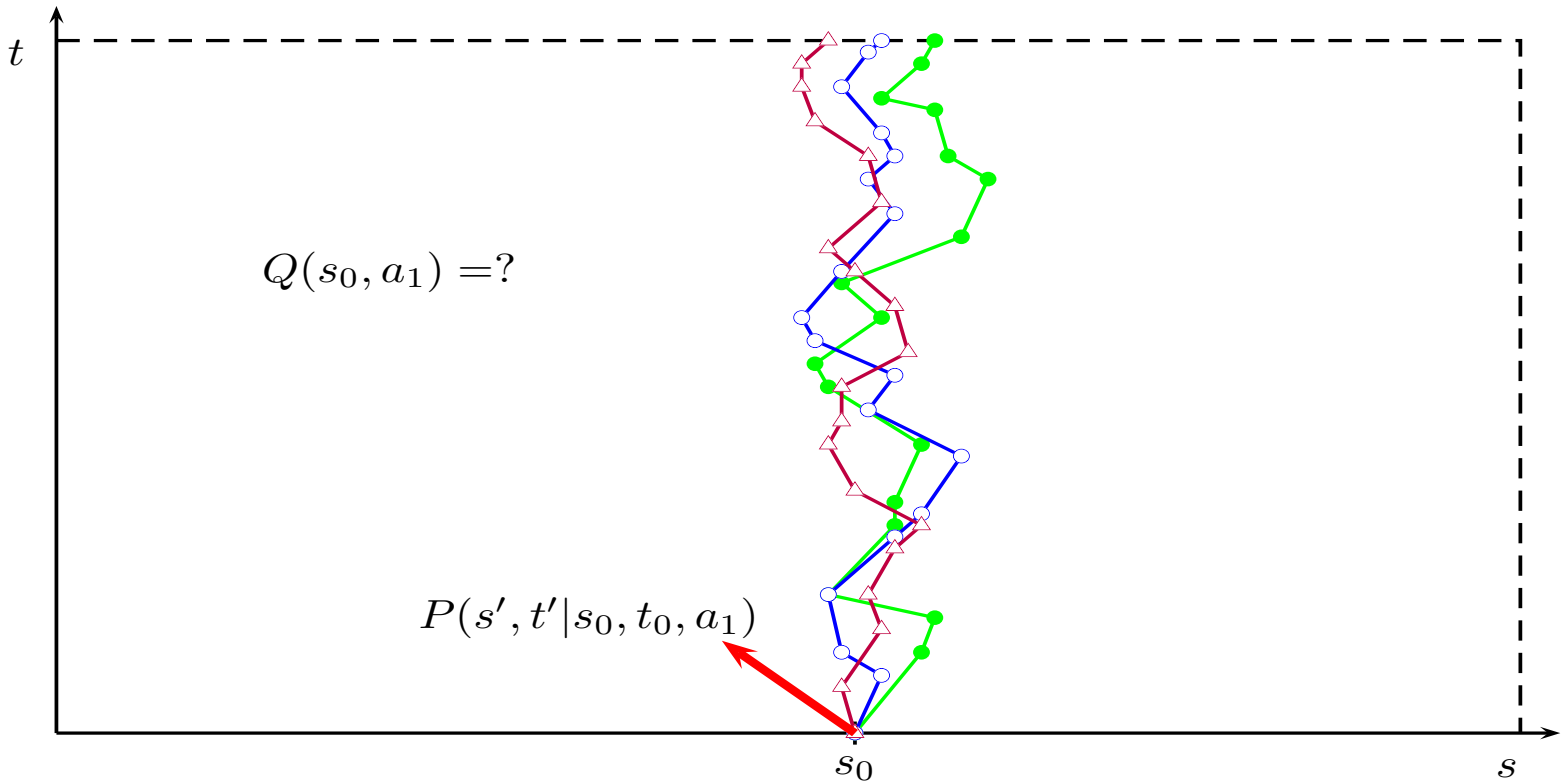
Compute $\tilde{V}^{\pi_{n+1}}$ for the last(s) episode(s)

But . . .

Avoiding the pitfall of partial exploration

The $\tilde{R}(s,t)$ are not drawn i.i.d. (only independently).
$\longrightarrow \tilde{V}^{\pi}$ is a biased estimator.

$\tilde{V}^{\pi}$ is only valid locally $\longrightarrow$ local confidence in $\tilde{V}^{\pi}$



Avoiding the pitfall of partial exploration

The $\tilde{R}(s,t)$ are not drawn i.i.d. (only independently).
$\longrightarrow \tilde{V}^{\pi}$ is a biased estimator.

$\tilde{V}^{\pi}$ is only valid locally $\longrightarrow$ local confidence in $\tilde{V}^{\pi}$

Confidence function C^V

Can we trust $\tilde{V}^{\pi}(s,t)$ as an approximation of V^{π} in (s,t)?

$$C^V : \begin{cases} S \times \mathbb{R} & \longrightarrow & \{\top, \bot\} \\ s,t & \longmapsto & C^V(s,t) \end{cases}$$

$$\tilde{V}^{\pi}(s,t) \longrightarrow C^V(s,t)$$
$$\pi(s,t) \longrightarrow C^{\pi}(s,t)$$

iATPI

iATPI

iATPI: {
- Asynchronous policy iteration for greedy search
- Time-dependency & Monte-Carlo sampling
- Local policies and values via confidence functions

- Asynchronous PI: local improvements / partial evaluation.
- t-dependent Monte-Carlo sampling: loopless — finite — total criterion.
- Confidence functions: alternative to heuristic-based approaches.

iATPI

Given the current policy π_n,
the current process state (s, t),
the current estimate $\tilde{V}^{\pi_n}$

Compute the best action a^* with respect to $\tilde{V}^{\pi_n}$

Use $C^{\tilde{V}^{\pi_n}}$ to check if $\tilde{V}^{\pi_n}$ can be used
Sample more evaluation trajectories for π_n if not
Refine $\tilde{V}^{\pi_n}$ and $C^{\tilde{V}^{\pi_n}}$

Pick (s', t') according to a^*

Until $t' > T$

Compute $\tilde{V}^{\pi_{n+1}}, C^{\tilde{V}^{\pi_{n+1}}}, \pi_{n+1}, C^{\pi_{n+1}}$ for the last(s) episode(s)

Output

A pile $\Pi_n = \{(\pi_0, C^{\pi_0}), (\pi_1, C^{\pi_1}), \ldots, (\pi_n, C^{\pi_n}) | C^{\pi_0}(s,t) = \top\}$ of partial policies.

Preliminary results with *iATPI*

Preliminary results on ATPI and the subway problem:

Subway problem

4 trains, 6 stations

$\longrightarrow$ 22 hybrid state variables, 9 actions

episodes of 12 hours with around 2000 steps.

Preliminary results with *iATPI*

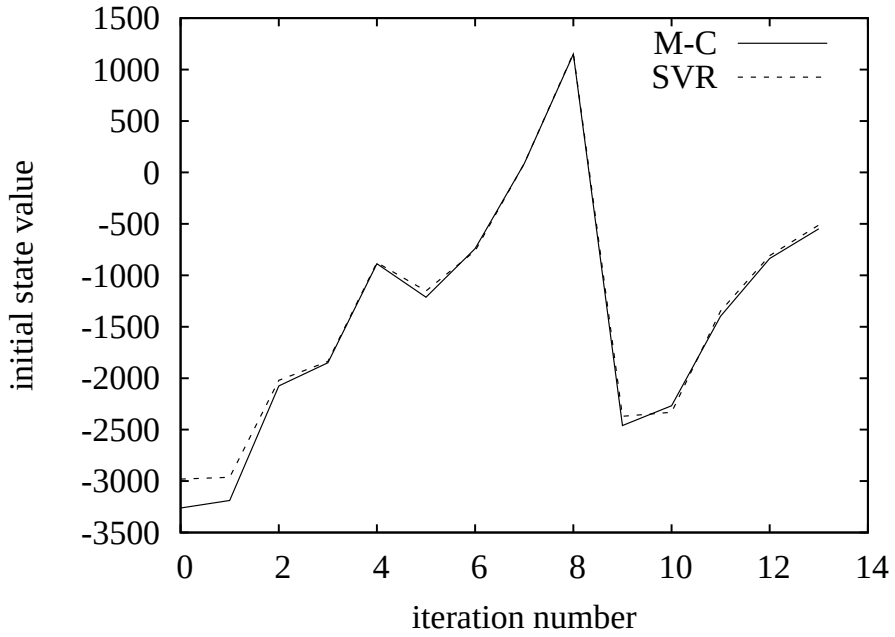
Preliminary results on ATPI and the subway problem:

With proper initialization, *naive* ATPI finds good policies.

[Rachelson et al., 2008b, Rachelson et al., 2008c]

Preliminary results with *iATPI*

Preliminary results on ATPI and the subway problem:



Value functions, policies and confidence functions

How do we write $\tilde{V}$, C^V, π and C^π?

$\longrightarrow$ Statistical learning problem

We implemented and tried several options:

$\tilde{V}$ incremental, local regression problem.

SVR, LWPR, Nearest-neighbours.

π local classification problem.

SVC, Nearest-neighbours.

C incremental, local statistical sufficiency test.

OC-SVM, central-limit theorem.

Value functions, policies and confidence functions

Since they are **local** and able to answer "I don't know",
the value functions and policies of *iATPI*
can be considered as KWIK learners.

Li, L., Littman, M. L., and Walsh, T. J. (2008).
Knows What It Knows: A Framework for Self-Aware Learning.
In *International Conference on Machine Learning*.

Learning rate and convergence guarantees?

Perspectives for *iATPI*

iATPI is ongoing work
$\longrightarrow$ no hasty conclusions

Needed work: extensive testing of the algorithm's full version.

Still lots of open questions:

- How to avoid local maxima in value function space?
- Test on a fully discrete and observable problem?

. . . and many ideas for improvement:

- Use V_{n-k} functions as lower bounds on V_n
- Utility functions for stopping sampling in *episode*.bestAction()

Contributions and experience feedback

- Modeling framework for stochastic decision processes: GSMDPs + continuous time.
- *iATPI*

Modeling claim

Describing **concurrent**, **exogenous** contributions to the system's dynamics **separately**.

Concurrent observable-time SMDPs affecting the same state space
$\longrightarrow$ observable-time GSMDPs.

Natural framework for describing temporal problems.

Integration in the VLE platform for DEVS multi-model simulation.

Contributions and experience feedback

- Modeling framework for stochastic decision processes: GSMDPs + continuous time.
- *iATPI*

iATPI

iATPI: $\begin{cases} \text{Asynchronous policy iteration} \\ \text{Time-dependency \& Monte-Carlo sampling} \\ \text{Confidence functions} \end{cases}$

- Asynchronous PI: local improvements / partial evaluation.
- t-dependent Monte-Carlo sampling: loopless — finite — total criterion.
- Confidence functions: alternative to heuristic-based approaches.

Thank you for your attention!

Boyan, J. A. and Littman, M. L. (2001).
Exact Solutions to Time Dependent MDPs.
Advances in Neural Information Processing Systems, 13:1026–1032.

Rachelson, E., Garcia, F., and Fabiani, P. (2008a).
Extending the Bellman Equation for MDP to Continuous Actions and Continuous Time in the Discounted Case.
In *International Symposium on Artificial Intelligence and Mathematics*.

Feng, Z., Dearden, R., Meuleau, N., and Washington, R.
Dynamic Programming for Structured Continuous Markov Decision Problems (2004).
In *Conference on Uncertainty in Artificial Intelligence*.

Li, L., and Littman, M. L.
Lazy Approximation for Solving Continuous Finite-Horizon MDPs (2005).
In *National Conference on Artificial Intelligence*.

Rachelson, E., Garcia, F., and Fabiani, P. (2009).
TiMDPpoly : an Improved Method for Solving Time-dependent MDPs.
Submitted to *International Conference on Automated Planning and Scheduling*.

Glynn, P. (1989).
A GSMP Formalism for Discrete Event Systems.
Proc. of the IEEE, 77.

Younes, H. L. S. and Simmons, R. G. (2004).
Solving Generalized Semi-Markov Decision Processes using Continuous Phase-Type Distributions.
In *AAAI Conference on Artificial Intelligence*.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996).
Neuro-Dynamic Programming.
Athena Scientific.

Rachelson, E., Quesnel, G., Garcia, F., and Fabiani, P. (2008b).
A Simulation-based Approach for Solving Generalized Semi-Markov Decision Processes.
In *European Conference on Artificial Intelligence*.

Rachelson, E., Quesnel, G., Garcia, F., and Fabiani, P. (2008c).
Approximate Policy Iteration for Generalized Semi-Markov Decision Processes: an Improved Algorithm.
In *European Workshop on Reinforcement Learning*.

Li, L., Littman, M. L., and Walsh, T. J. (2008).
Knows What It Knows: A Framework for Self-Aware Learning.
In *International Conference on Machine Learning*.

Quesnel, G., Duboz, R., Ramat, E., and Traore, M. K. (2007).
VLE - A Multi-Modeling and Simulation Environment.
In *Summer Simulation Conf.*, pages 367–374.