Temporal Markov Decision Problems

—

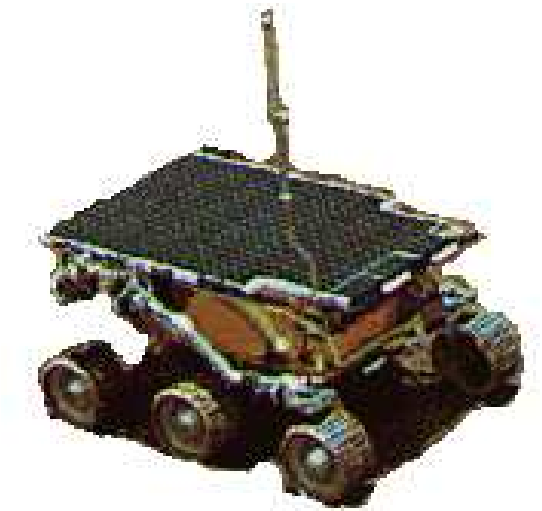
Formalization and Resolution

Emmanuel Rachelson

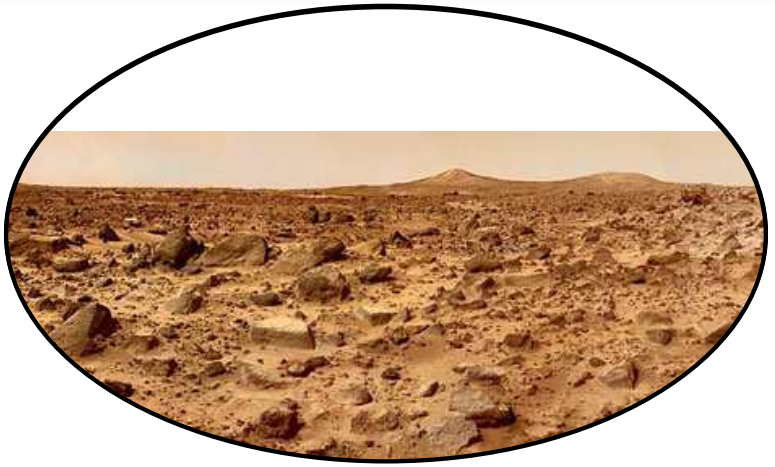
Ecole doctorale : Systèmes
Etablissement d'inscription : ISAE-SUPAERO
Laboratoire d'accueil : ONERA-DCSD

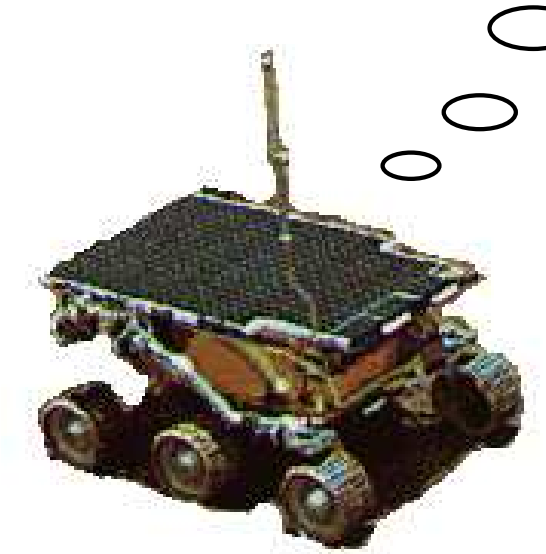
March 23rd, 2009

Motivation



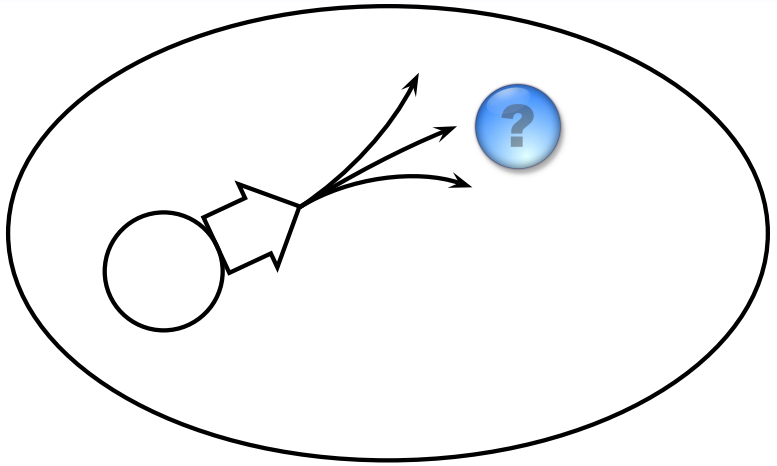
Motivation

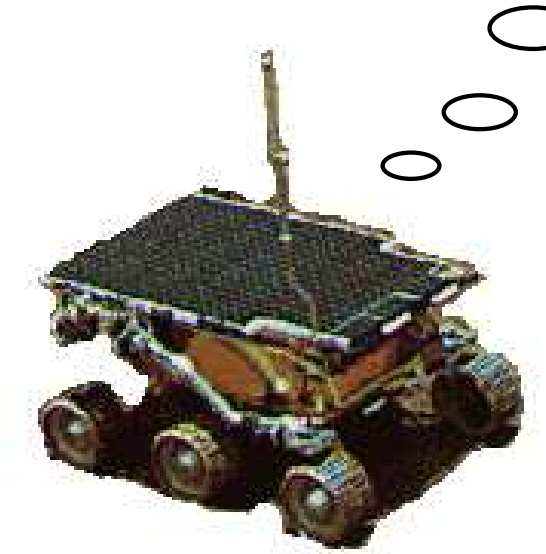




Performing
"as well as possible"

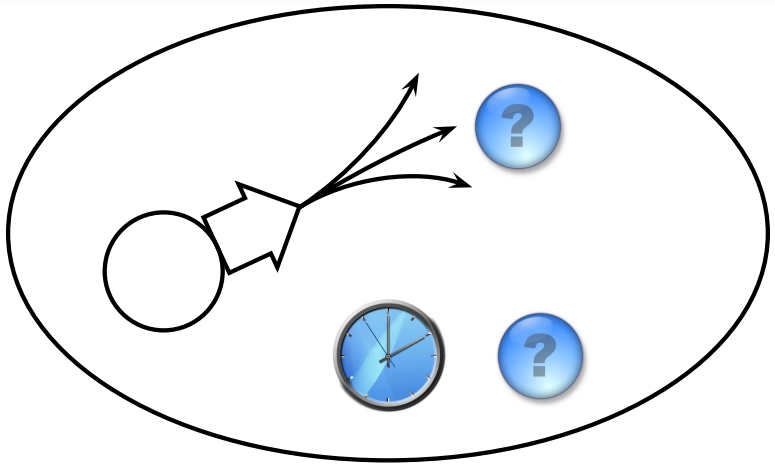
Motivation

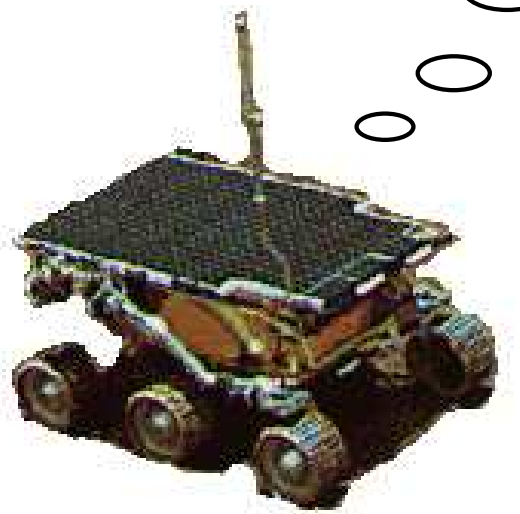




Uncertain outcomes

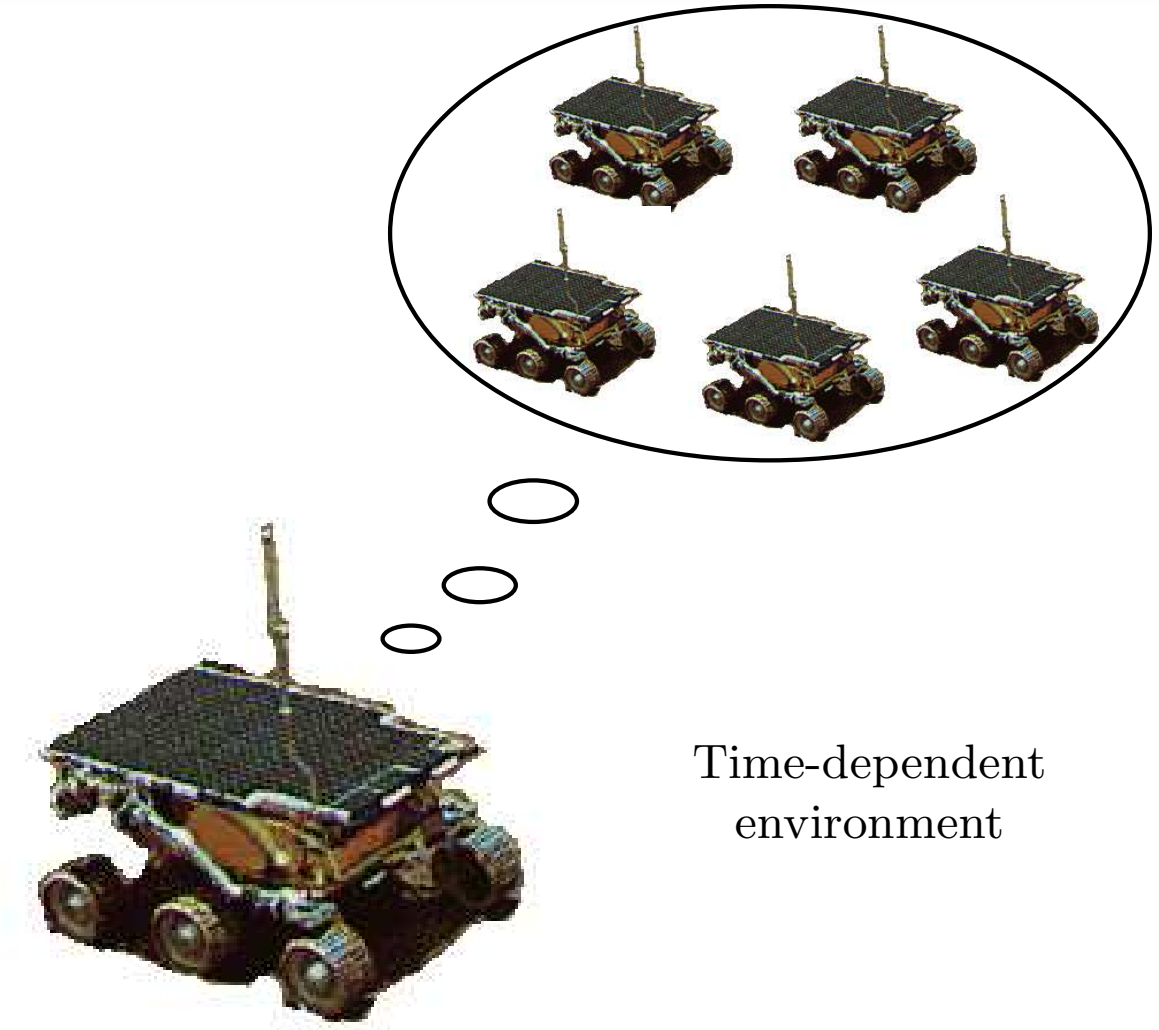
Motivation





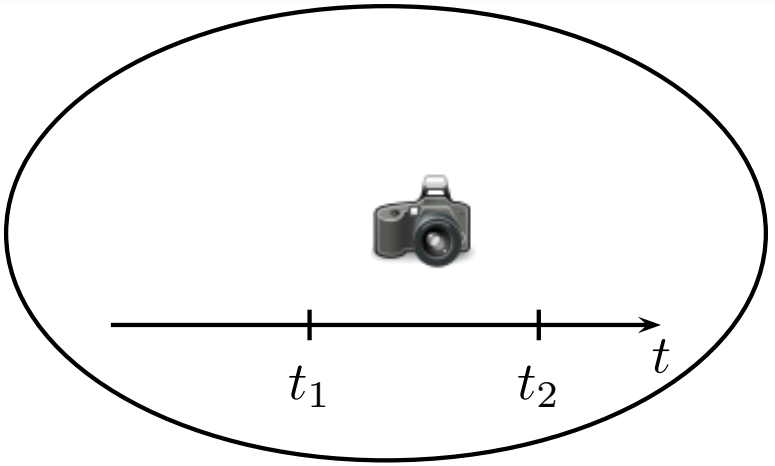
Uncertain durations

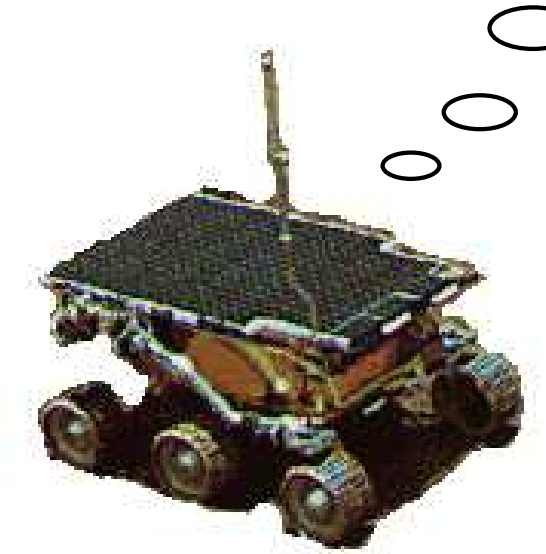
Motivation



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

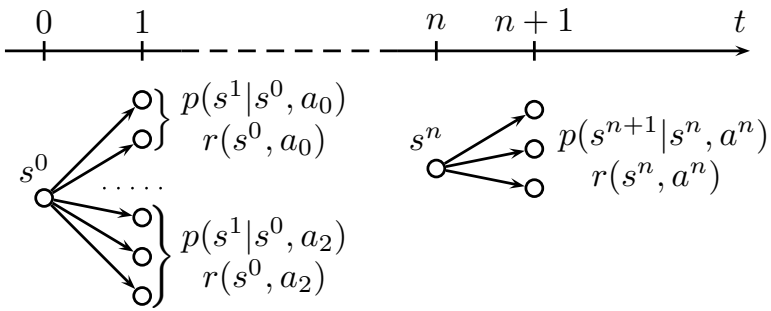
Modeling background

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

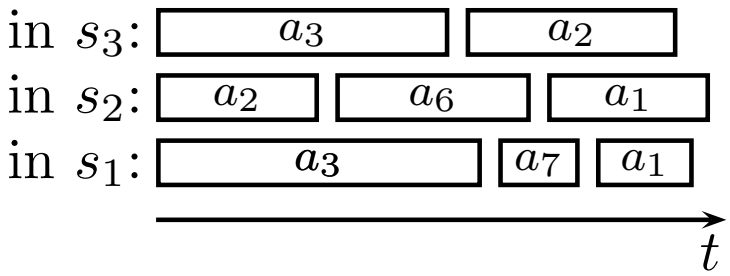
$$\mathscr{D} = \left\{\pi \ : \ \begin{array}{ccc} S & \rightarrow & A \\ s & \mapsto & \pi(s) = a \end{array}\right\}$$

Optimality equation

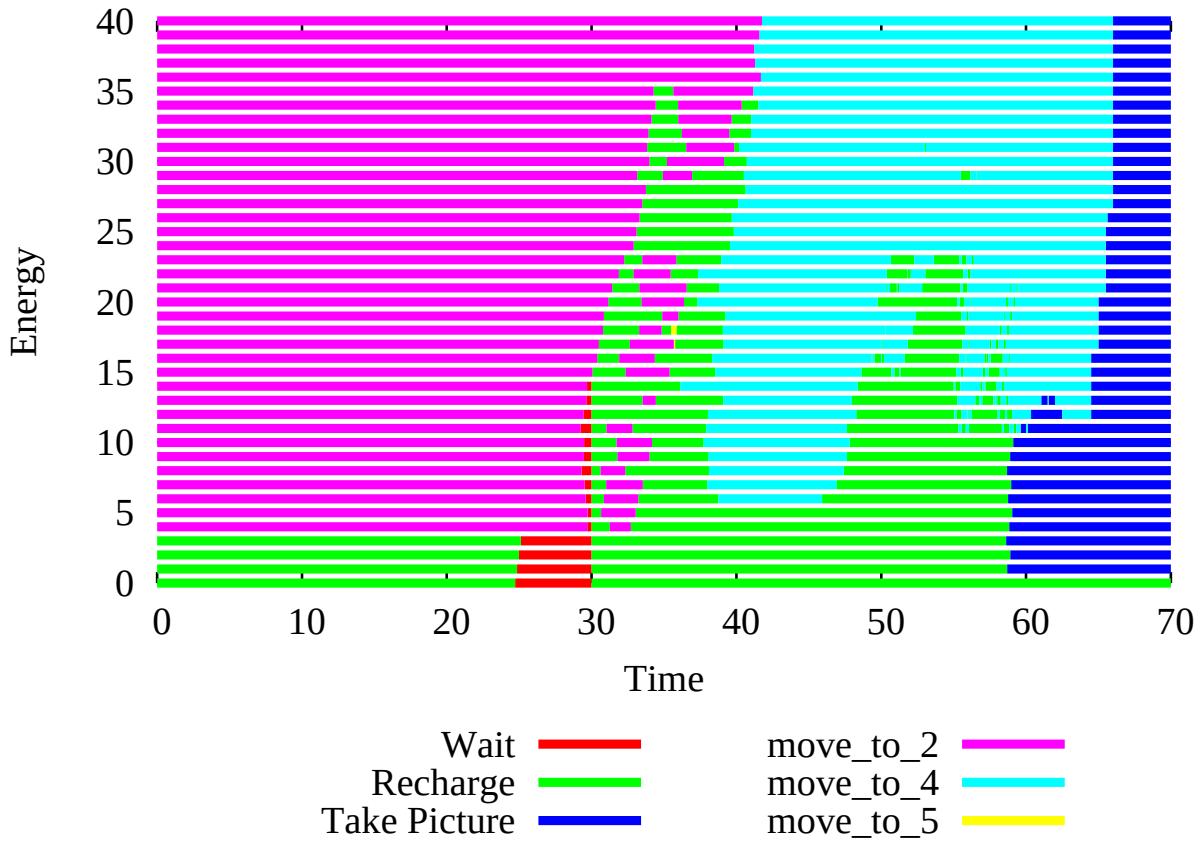
$$V^*(s) = \max_{\pi \in \mathscr{D}} V^{\pi}(s) = \max_{a \in A}\left\{r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s')\right\}$$

What are we looking for?

Time-dependent policies



What are we looking for?



Continuous durations in stochastic processes

MDPs: the set T contains integer-valued dates.

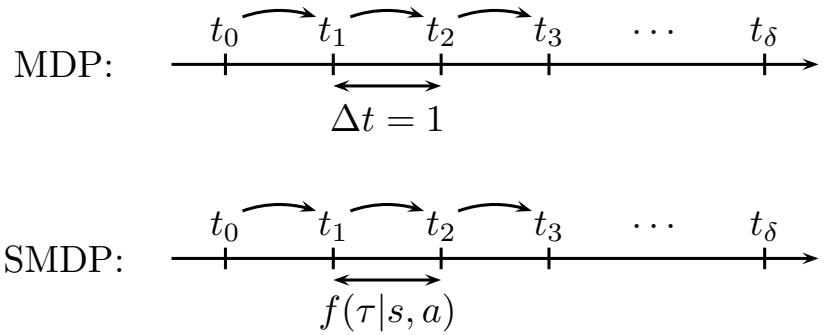
$\longrightarrow$ more flexible durations?

Semi-Markov Decision Process

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

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

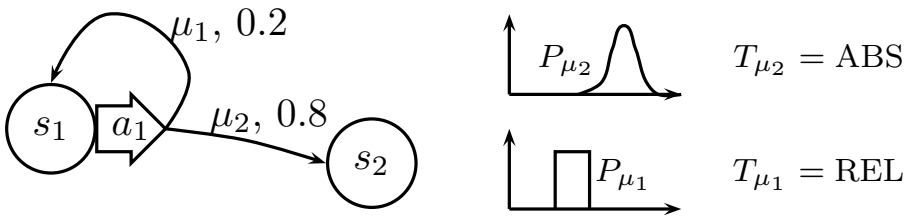
Transition model $p(s'|s,a)$ or $p(s'|s,a,\tau)$



Time-dependent MDPs

Definition (TMDP, [Boyan and Littman, 2001])

Tuple $\langle S, A, M, L, R, K \rangle$
M Set of outcomes $\mu = (s'_\mu, T_\mu, P_\mu)$
$L(\mu|s,t,a)$ Probability of triggering outcome μ
$R(\mu,t,t') = r_{\mu,t}(t) + r_{\mu,\tau}(t'-t) + r_{\mu,t'}(t')$

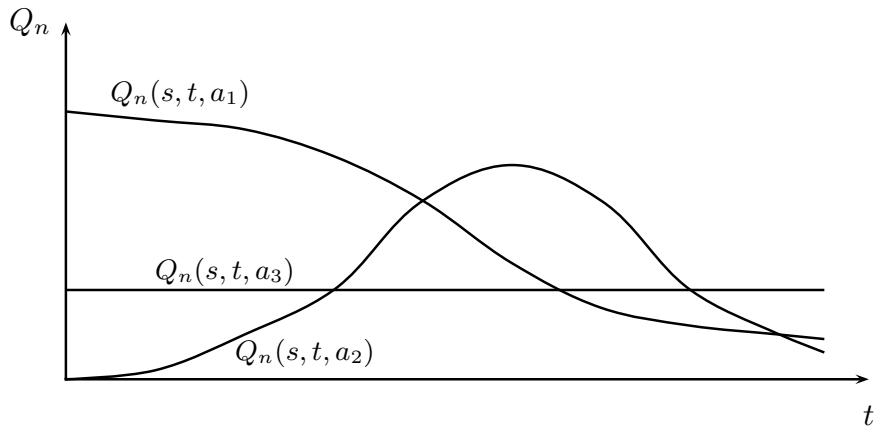


Boyan, J. A. and Littman, M. L. (2001).
Exact Solutions to Time Dependent MDPs.
Advances in Neural Information Processing Systems, 13:1026–1032.

TMDP optimality equation

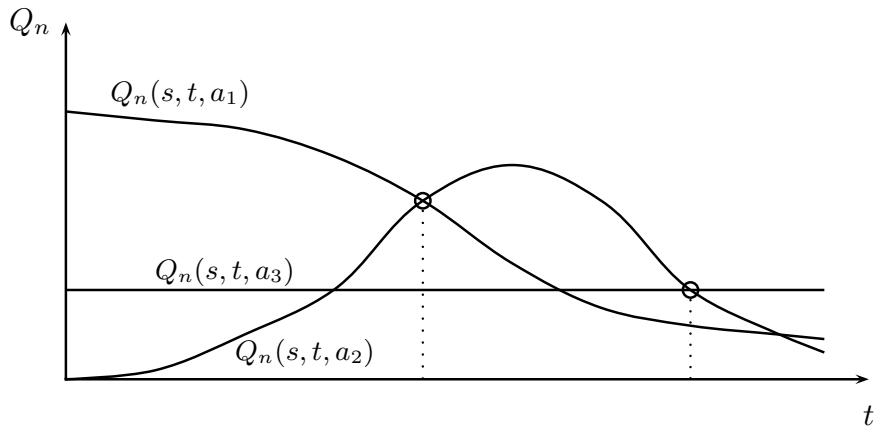
$$Q(s,t,a) = \sum_{\mu \in M} L(\mu|s,t,a) \cdot U(\mu,t)$$

$$U(\mu,t) = \begin{cases} \int_{-\infty}^{\infty} P_\mu(t')[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{ABS} \\ \int_{-\infty}^{\infty} P_\mu(t'-t)[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{REL} \end{cases}$$



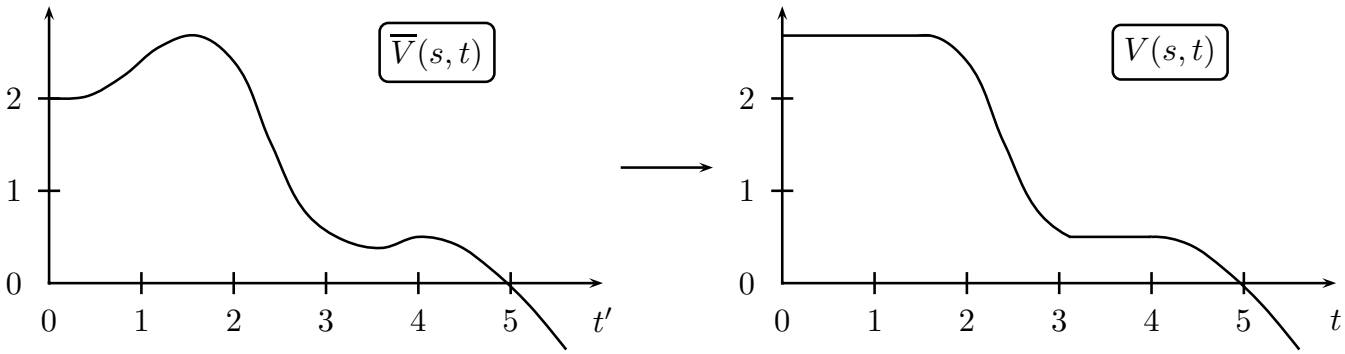
TMDP optimality equation

$$
\begin{aligned}
\overline{V}(s,t) &= \max_{a \in A} Q(s,t,a) \\
Q(s,t,a) &= \sum_{\mu \in M} L(\mu | s,t,a) \cdot U(\mu,t) \\
U(\mu,t) &= \begin{cases} \int_{-\infty}^{\infty} P_\mu(t')[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{ABS} \\ \int_{-\infty}^{\infty} P_\mu(t'-t)[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{REL} \end{cases}
\end{aligned}
$$



TMDP optimality equation

$$
\begin{aligned}
V(s,t) &= \sup_{t' \geq t} \left(\int_t^{t'} K(s,\theta) d\theta + \overline{V}(s,t') \right) \\
\overline{V}(s,t) &= \max_{a \in A} Q(s,t,a) \\
Q(s,t,a) &= \sum_{\mu \in M} L(\mu | s,t,a) \cdot U(\mu,t) \\
U(\mu,t) &= \begin{cases} \int_{-\infty}^{\infty} P_\mu(t')[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{ABS} \\ \int_{-\infty}^{\infty} P_\mu(t'-t)[R(\mu,t,t') + V(s'_\mu,t')]dt' & \text{if } T_\mu = \text{REL} \end{cases}
\end{aligned}
$$



An MDP with continuous observable time?

SMDPs no explicit time-dependency

TMDPs time-dependent but { no explicit criterion / no theoretical guarantees / restrictions on the model

$\Rightarrow$ Can we provide a sound and more general framework for representing time in MDPs?

Including observable time in MDPs

Can an MDP represent its own process' time as a state variable?

XMDP

Tuple $\langle \Sigma, A(X), p, r \rangle$

Σ	$\sigma = (s,t) \in \mathscr{B}(S \times \mathbb{R})$
$A(X)$	compact set of parametric actions $a_i(x)$
$p(\sigma' \mid \sigma, a(x))$	upper semi-continuous w.r.t. x
$r(\sigma, a(x))$	positive, upper semi-continuous w.r.t. x

Steady time advance

$$\forall (\sigma, a(x)) \in \Sigma \times A(X),\ \exists \alpha > 0 /\ t' < t + \alpha \Rightarrow p(\sigma' \mid \sigma, a(x)) = 0$$

$$\text{“}t_{\delta+1} \geq t_\delta + \alpha\text{”}$$

Theorem (XMDP optimality equation, [Rachelson et al., 2008a])

The optimal value function V^* is the unique solution of:

$$\forall (s,t) \in S \times \mathbb{R},\ V(s,t) =$$
$$\sup_{a(x) \in A(X)} \left\{ r(s,t,a(x)) + \int_{\substack{t' \in \mathbb{R} \\ s' \in S}} \gamma^{t'-t} p(s',t'|s,t,a(x)) V(s',t') ds' dt' \right\}$$

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*.

Theorem (XMDP optimal policy)

Under the previous assumptions, there exists a deterministic, Markovian policy such that $V^\pi = V^*$.

TMDPs and XMDPs

Optimality equation and conditions

TMDP optimality equation $\equiv$ XMDP equation with specific assumptions.

- total reward criterion
- t-deterministic and s-static, implicit *wait* action
- interleaving of wait/action
- no lump sum reward for *wait* action
- assumptions on r, L, P_μ so that the optimal policy exists
- assumptions on r, L, P_μ so that the systems retains physical meaning

TMDPs and XMDPs

Optimality equation and conditions

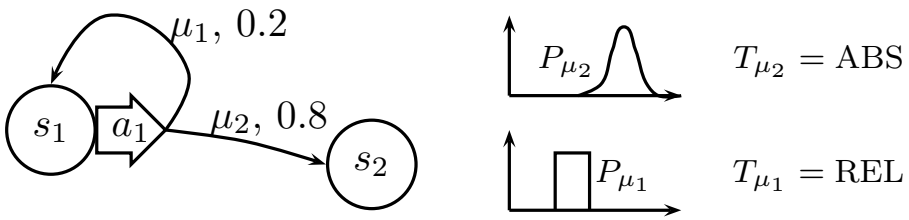
TMDP optimality equation $\equiv$ XMDP equation with specific assumptions.

XMDPs provide proven optimality conditions and equation.

But solving the general case of XMDPs is too complex.

$\longrightarrow$ In practice, we turn back to solving TMDPs

Solving TMDPs

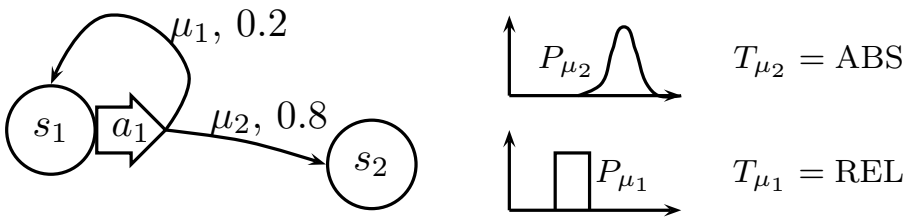


Value iteration Bellman backups for TMDPs can be performed exactly if:

- $L(\mu|s,t,a)$ piecewise constant
- $R(\mu,t,t') = r_{\mu,t}(t) + r_{\mu,\tau}(t'-t) + r_{\mu,t'}(t')$
- $r_{\mu,t}(t), r_{\mu,\tau}(\tau), r_{\mu,t'}(t')$ piecewise linear
- $P_\mu(t')$, $P_\mu(t'-t)$ discrete distributions

Then $V^*(s,t)$ is piecewise linear.

Solving TMDPs



- What about other, more expressive functions?
- How does this theoretical result scale to practical resolution?

Extending exact resolution

Piecewise polynomial models: L, P_μ, $r_i \in \mathscr{P}_n$.

Degree evolution

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i, V_0 \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow d^\circ(V_n) = B + n(A + C + 1)$$

Extending exact resolution

Piecewise polynomial models: L, P_μ, $r_i \in \mathscr{P}_n$.

Degree evolution

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i, V_0 \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow d^\circ(V_n) = B + n(A + C + 1)$$

$$\text{Stability} \Leftrightarrow A + C = -1.$$

Extending exact resolution

Piecewise polynomial models: L, P_μ, $r_i \in \mathscr{P}_n$.

Degree evolution

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i, V_0 \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow d^\circ(V_n) = B + n(A + C + 1)$$

$$\text{Stability} \Leftrightarrow A + C = -1.$$

Exact resolution conditions

Degree stability + exact analytical computations: $\left\{\begin{array}{l} P_\mu \in \mathscr{DP}_{-1} \\ r_i \in \mathscr{P}_4 \\ L \in \mathscr{P}_0 \end{array}\right.$

Extending exact resolution

Piecewise polynomial models: L, P_μ, $r_i \in \mathscr{P}_n$.

Degree evolution

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i, V_0 \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow d^\circ(V_n) = B + n(A + C + 1)$$

$$\text{Stability} \Leftrightarrow A + C = -1.$$

Exact resolution conditions

Degree stability + exact analytical computations: $\left\{\begin{array}{l} P_\mu \in \mathscr{DP}_{-1} \\ r_i \in \mathscr{P}_4 \\ L \in \mathscr{P}_0 \end{array}\right.$

If $B > 4$: approximate root finding.

Extending exact resolution

Piecewise polynomial models: L, P_μ, $r_i \in \mathscr{P}_n$.

Degree evolution

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i, V_0 \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow d^\circ(V_n) = B + n(A + C + 1)$$

$$\text{Stability} \Leftrightarrow A + C = -1.$$

Exact resolution conditions

Degree stability + exact analytical computations: $\left\{\begin{array}{l} P_\mu \in \mathscr{DP}_{-1} \\ r_i \in \mathscr{P}_4 \\ L \in \mathscr{P}_0 \end{array}\right.$

If $A + C > 0$: projection scheme of V_n on $\mathscr{P}_B$.

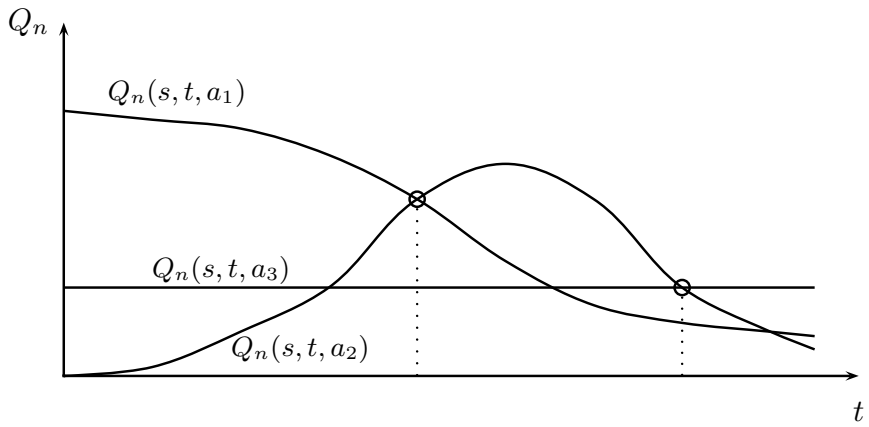
And in practice?

Fact (Admitted)

The number of definition intervals in V_n grows with n and does not necessarily converge.

$\Rightarrow$ numerical problems occur before $\| V_n - V_{n-1} \| < \varepsilon$.

e.g. $\overline{V}$ calculation:



And in practice?

Fact (Admitted)

The number of definition intervals in V_n grows with n and does not necessarily converge.

$\Rightarrow$ numerical problems occur before $\|V_n - V_{n-1}\| < \varepsilon$.

$\longrightarrow$ general case: approximate resolution by piecewise polynomial **interval simplification** for the value function.

Approximation $\nearrow$ degree reduction

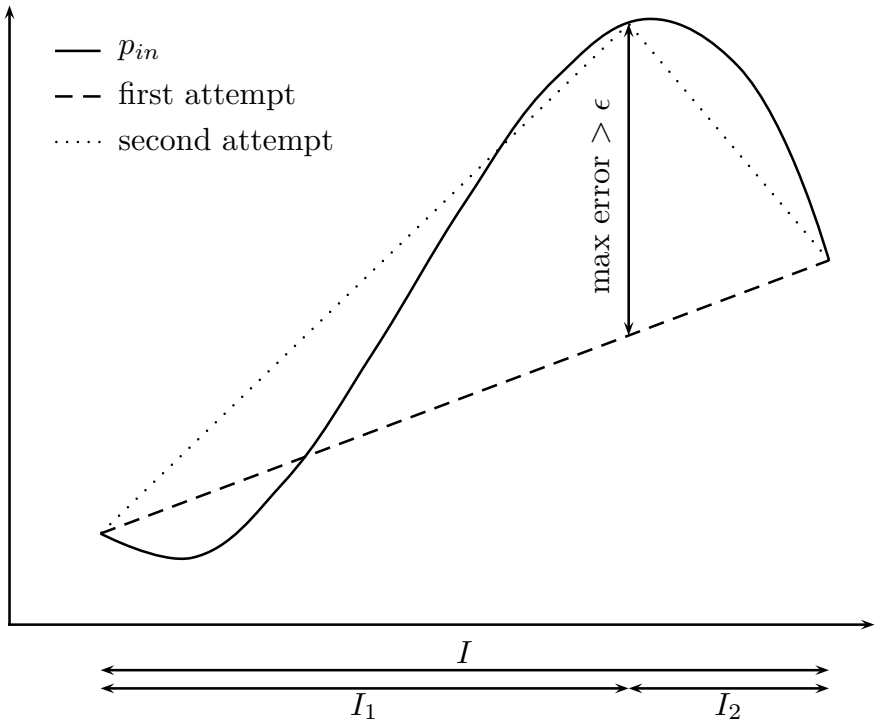
Approximation $\searrow$ interval simplification

$TMDP_{poly}$: Approximate Value Iteration on TMDPs

$TMDP_{poly}$ polynomial approximation

$p_{out} = \text{poly\_approx}(p_{in}, [l, u], \varepsilon, B)$

Two phases: incremental refinement and simplification.

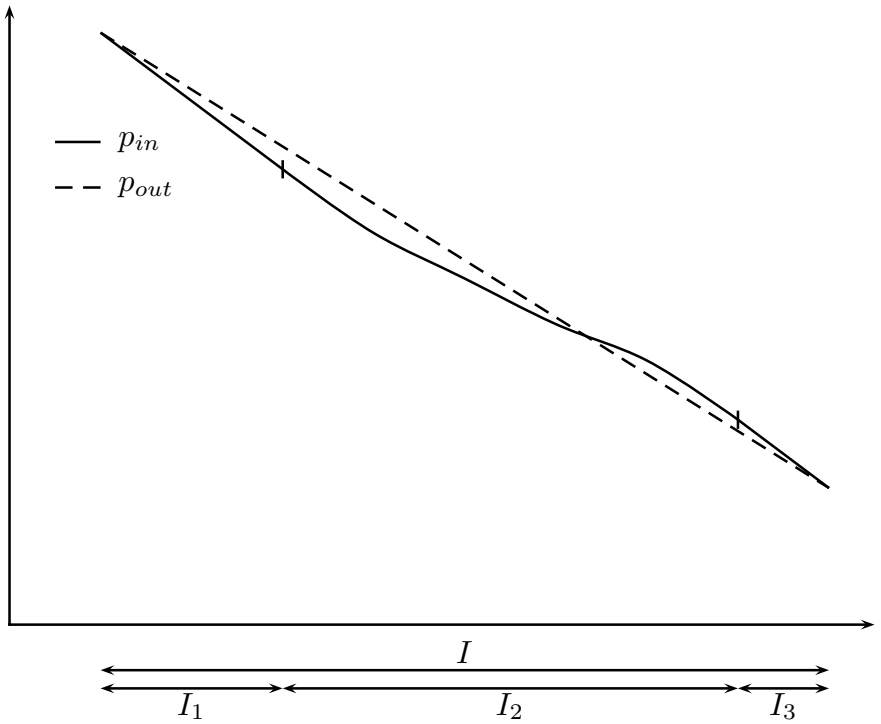


$TMDP_{poly}$: Approximate Value Iteration on TMDPs

$TMDP_{poly}$ polynomial approximation

$p_{out} = \text{poly\_approx}(p_{in}, [l, u], \varepsilon, B)$

Two phases: incremental refinement and simplification.



$TMDP_{poly}$: Approximate Value Iteration on TMDPs

$TMDP_{poly}$ polynomial approximation

$p_{out} = \text{poly\_approx}(p_{in}, [l, u], \varepsilon, B)$
Two phases: incremental refinement and simplification.

Properties

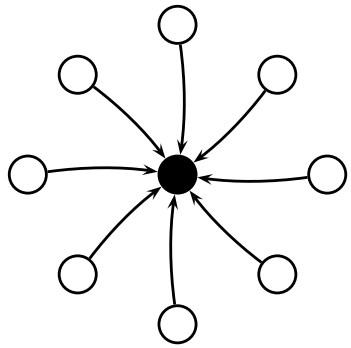
- $p_{out} \in \mathscr{P}_B$
- $\|p_{in} - p_{out}\|_\infty \leq \varepsilon$
- suboptimal number of intervals
- good complexity compromise

$TMDP_{poly}$: Approximate Value Iteration on TMDPs

Prioritized Sweeping.

Leveraging the computational effort by
ordering Bellman backups

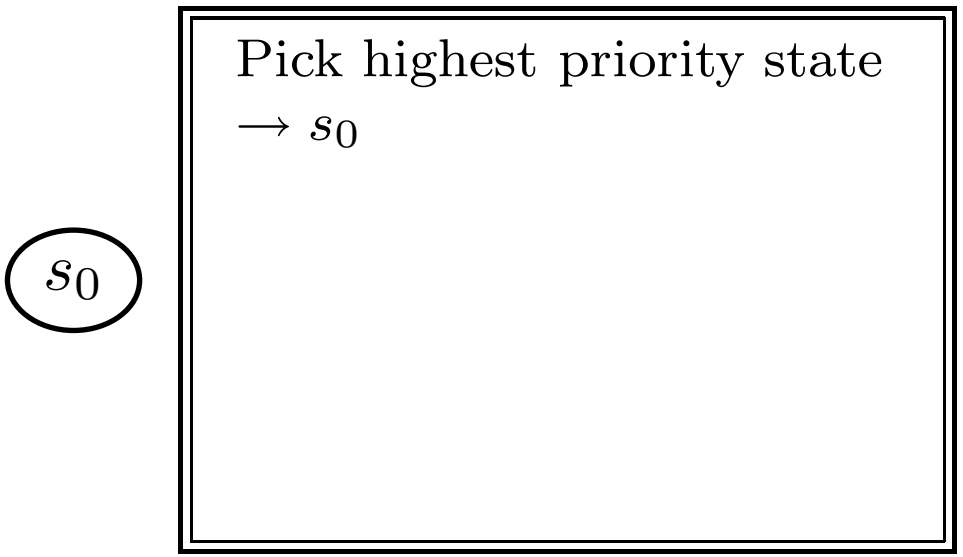
Perform Bellman backups in states with the largest value function change.



Moore, A. W. and Atkeson, C. G. (1993).
Prioritized Sweeping: Reinforcement Learning with Less Data and Less Real Time.
Machine Learning Journal, 13(1):103–105.

$TMDP_{poly}$: Approximate Value Iteration on TMDPs

Adapting Prioritized Sweeping to TMDPs.



$TMDP_{poly}$: Approximate Value Iteration on TMDPs

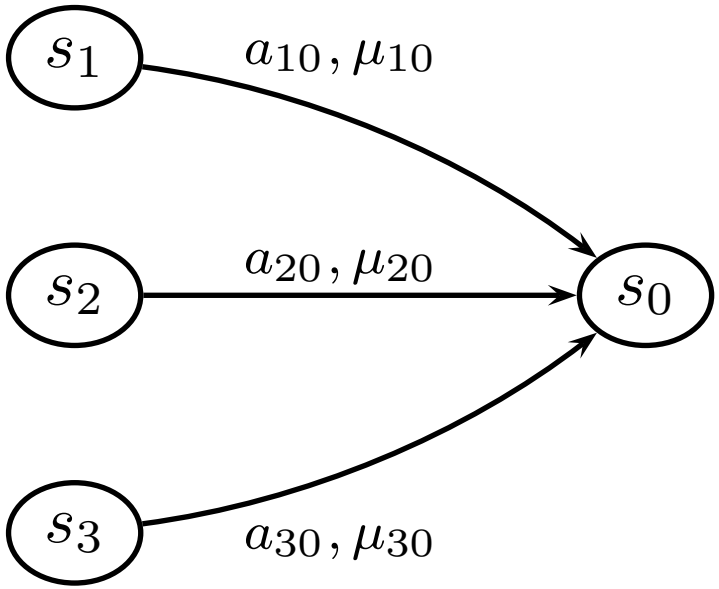
Adapting Prioritized Sweeping to TMDPs.

`update` $\overline{V}(s_0, t)$
`update` $V(s_0, t)$
$\text{poly\_approx}\,(V(s_0, t))$

Pick highest priority state
$\rightarrow s_0$
Bellman backup
$\rightarrow V(s_0, t)$

$TMDP_{poly}$: Approximate Value Iteration on TMDPs

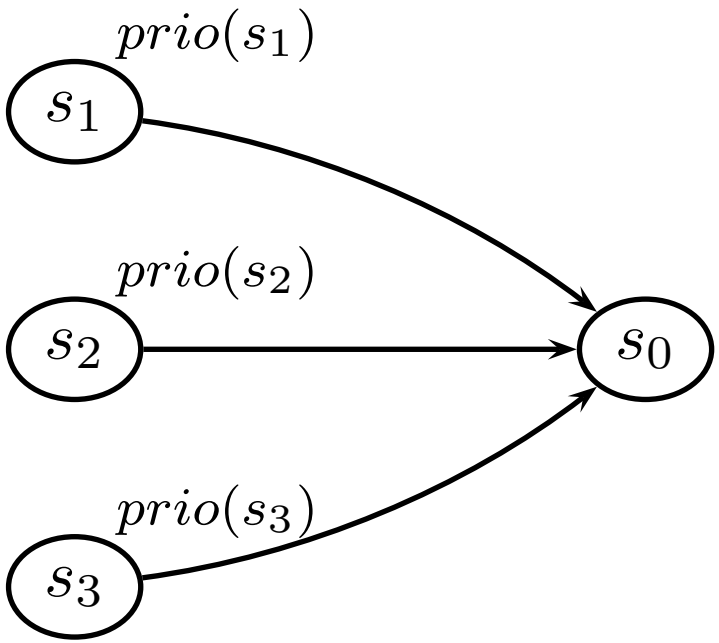
Adapting Prioritized Sweeping to TMDPs.



Pick highest priority state
$\rightarrow s_0$
Bellman backup
$\rightarrow V(s_0, t)$
Update Q values
$\rightarrow Q(s, t, a)$

$TMDP_{poly}$: Approximate Value Iteration on TMDPs

Adapting Prioritized Sweeping to TMDPs.



Pick highest priority state
$\rightarrow s_0$
Bellman backup
$\rightarrow V(s_0, t)$
Update Q values
$\rightarrow Q(s, t, a)$
Update priorities
$\rightarrow prio(s) = \|Q - Q_{old}\|_\infty$

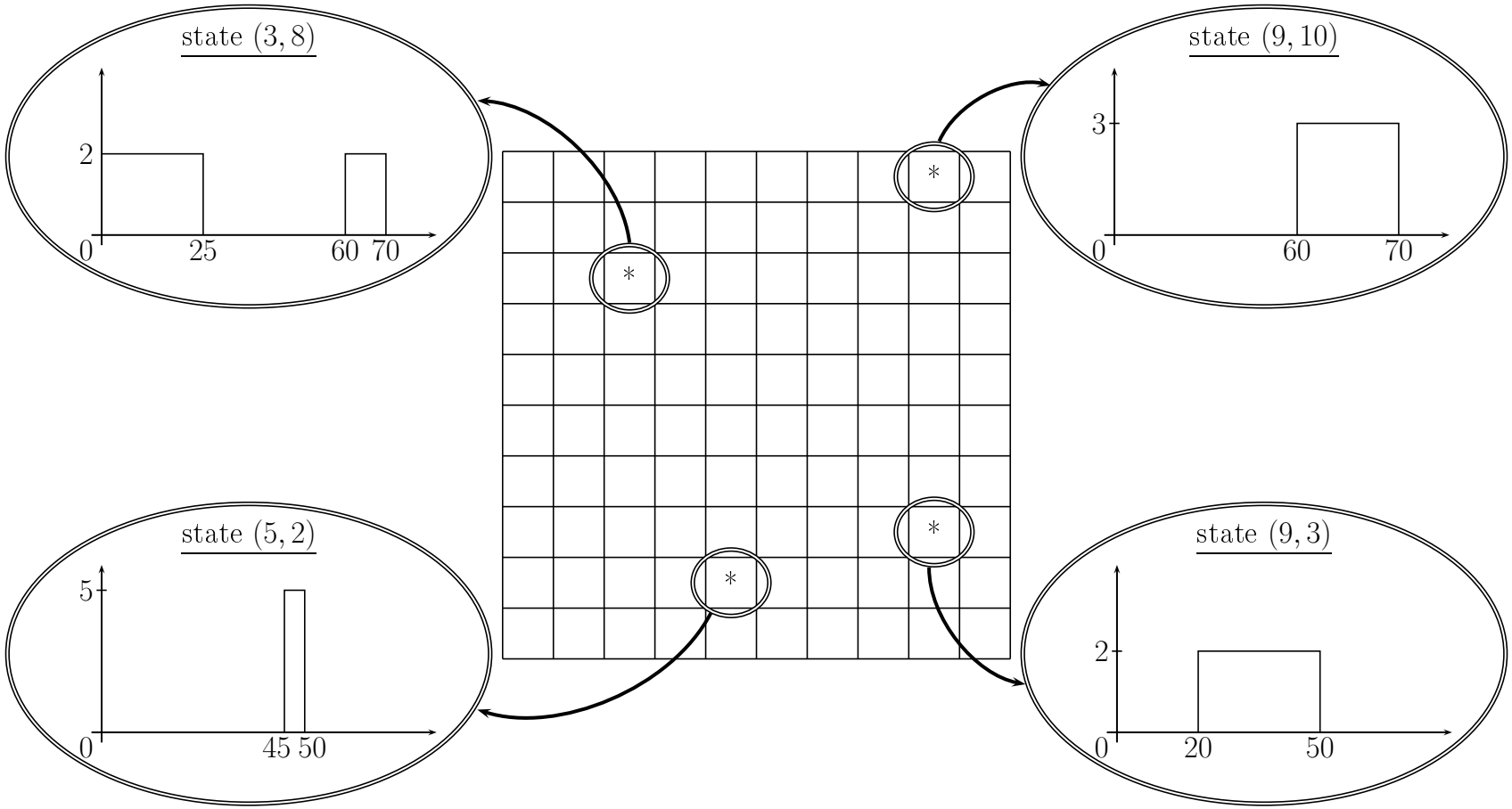
$TMDP_{poly}$

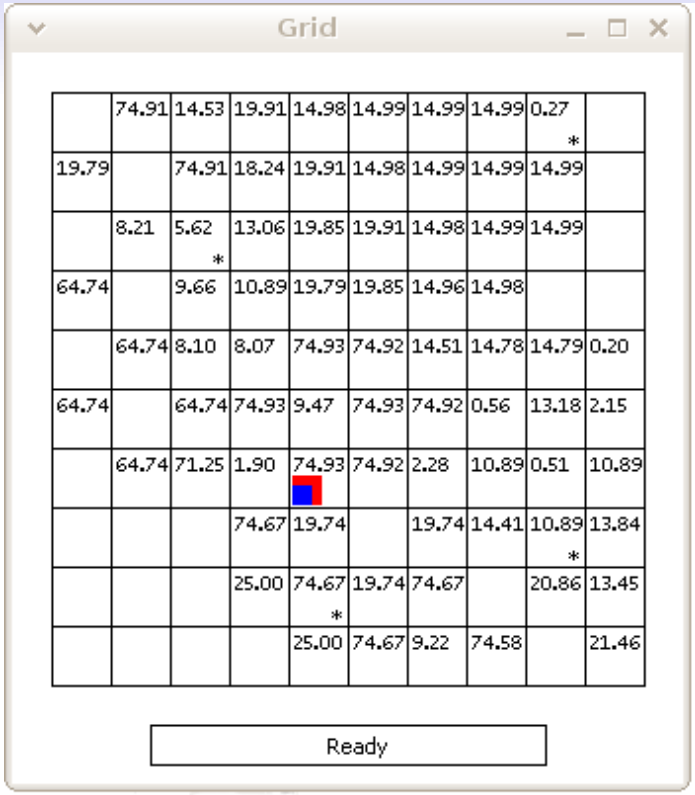
$TMDP_{poly}$ in a nutshell

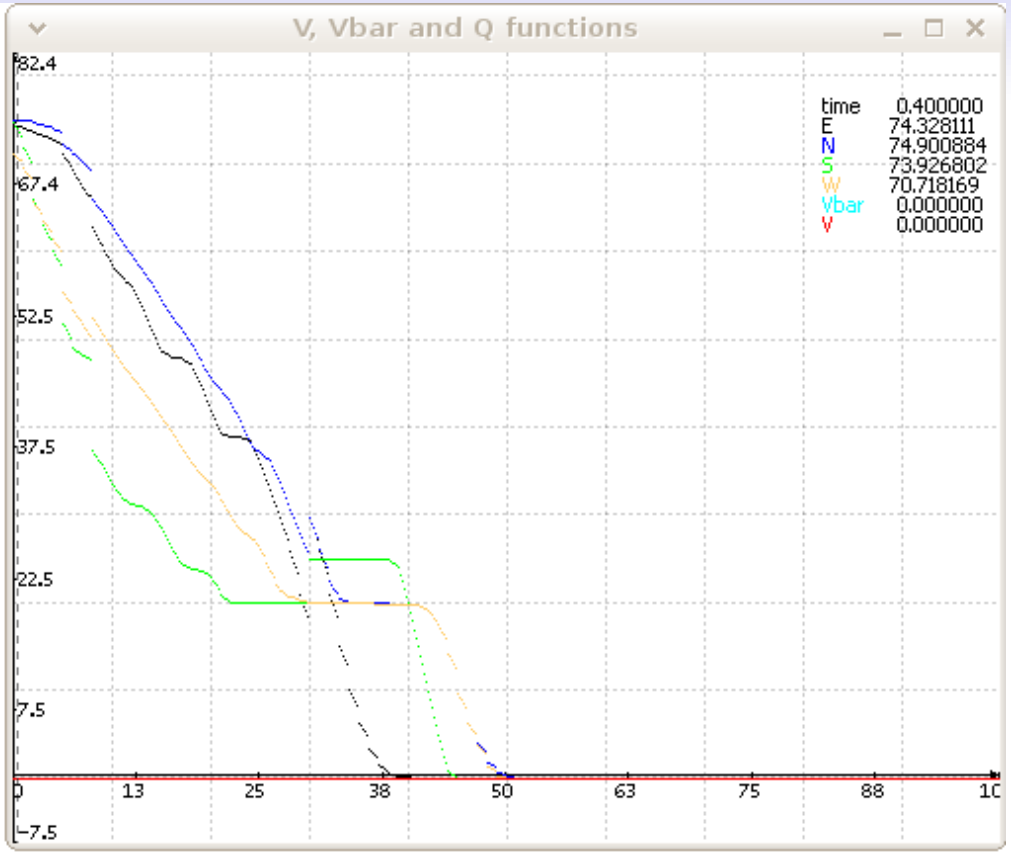
$TMDP_{poly}$: { Analytical polynomial calculations
L_∞-bounded error projection
Prioritized Sweeping for TMDPs

- Analytical operations: option for representing continuous quantities.
- Approximation makes resolution possible.
- Asynchronous VI makes it faster.

Illustration — UAV patrol problem



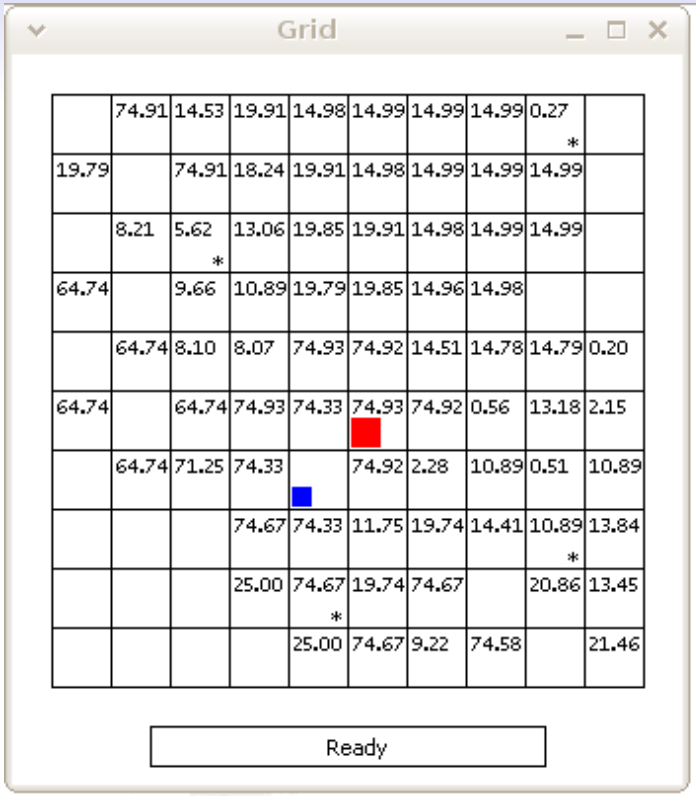


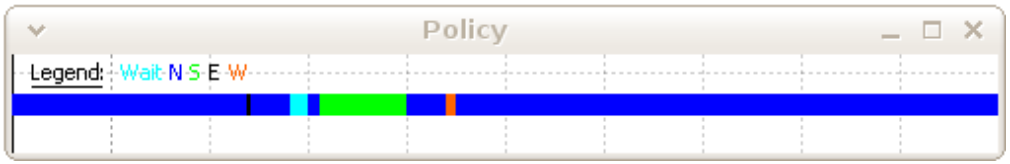


Backup #158

—

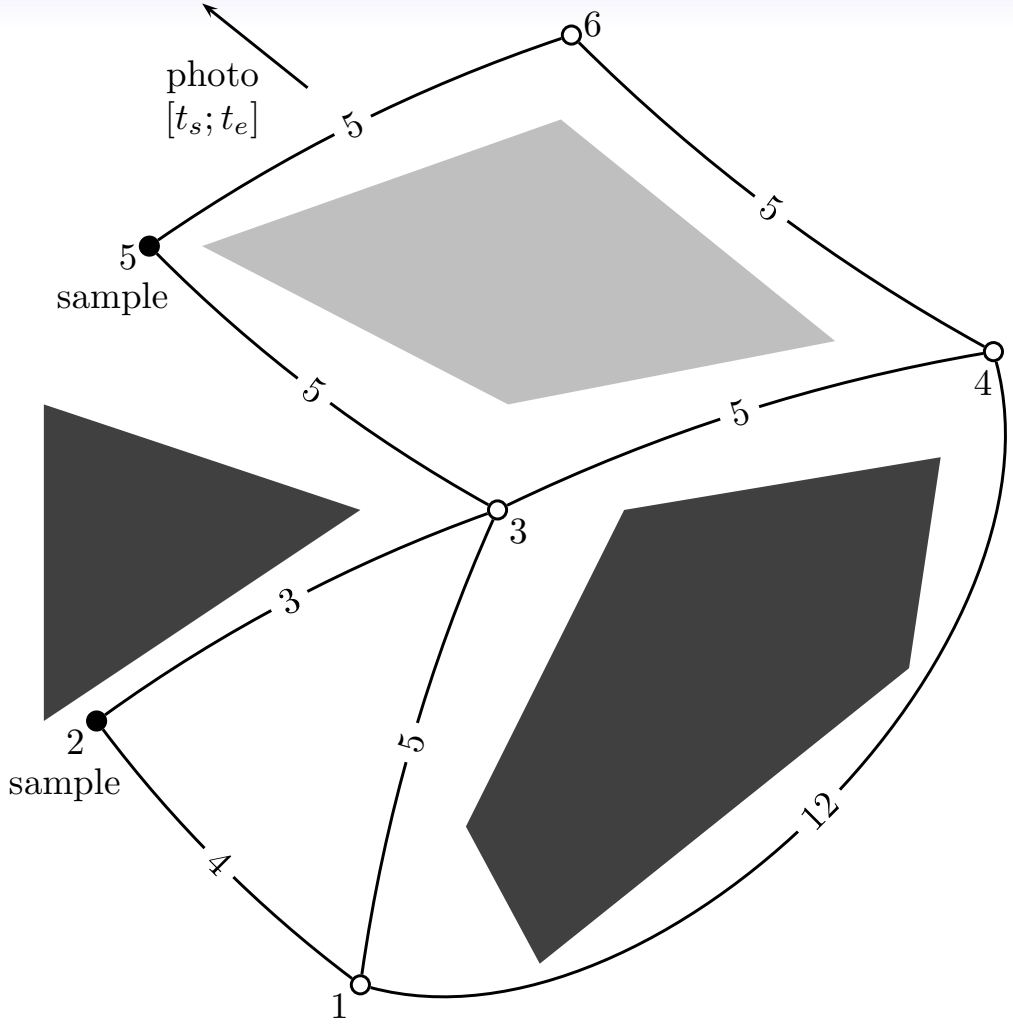
Compute $\overline{V}(s,t)$, $V(s,t)$ and poly_approx$(V(s,t))$





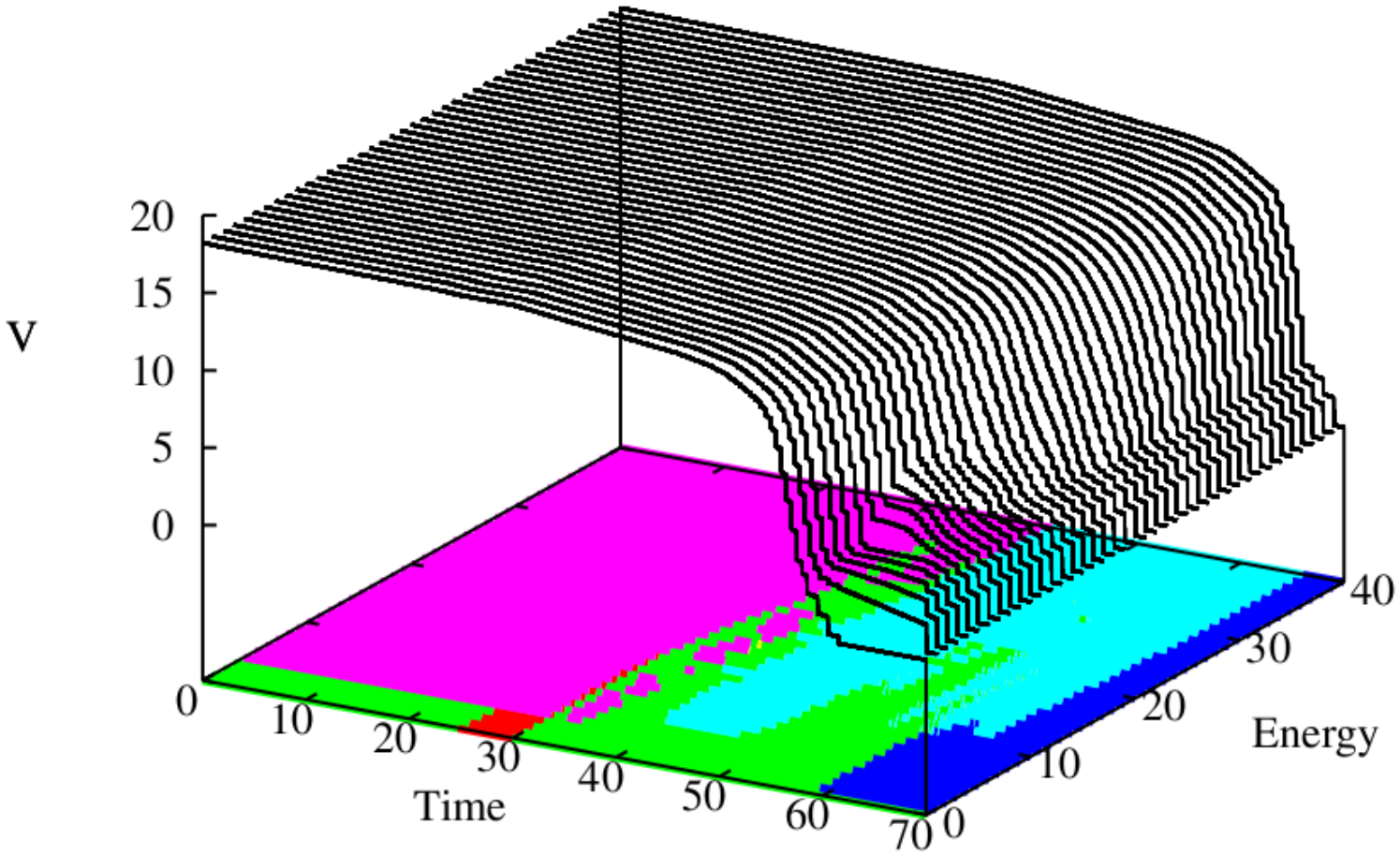
Compute $U(\mu, t)$, $Q(s, a, t)$ and $prio(s)$

Mars Rover



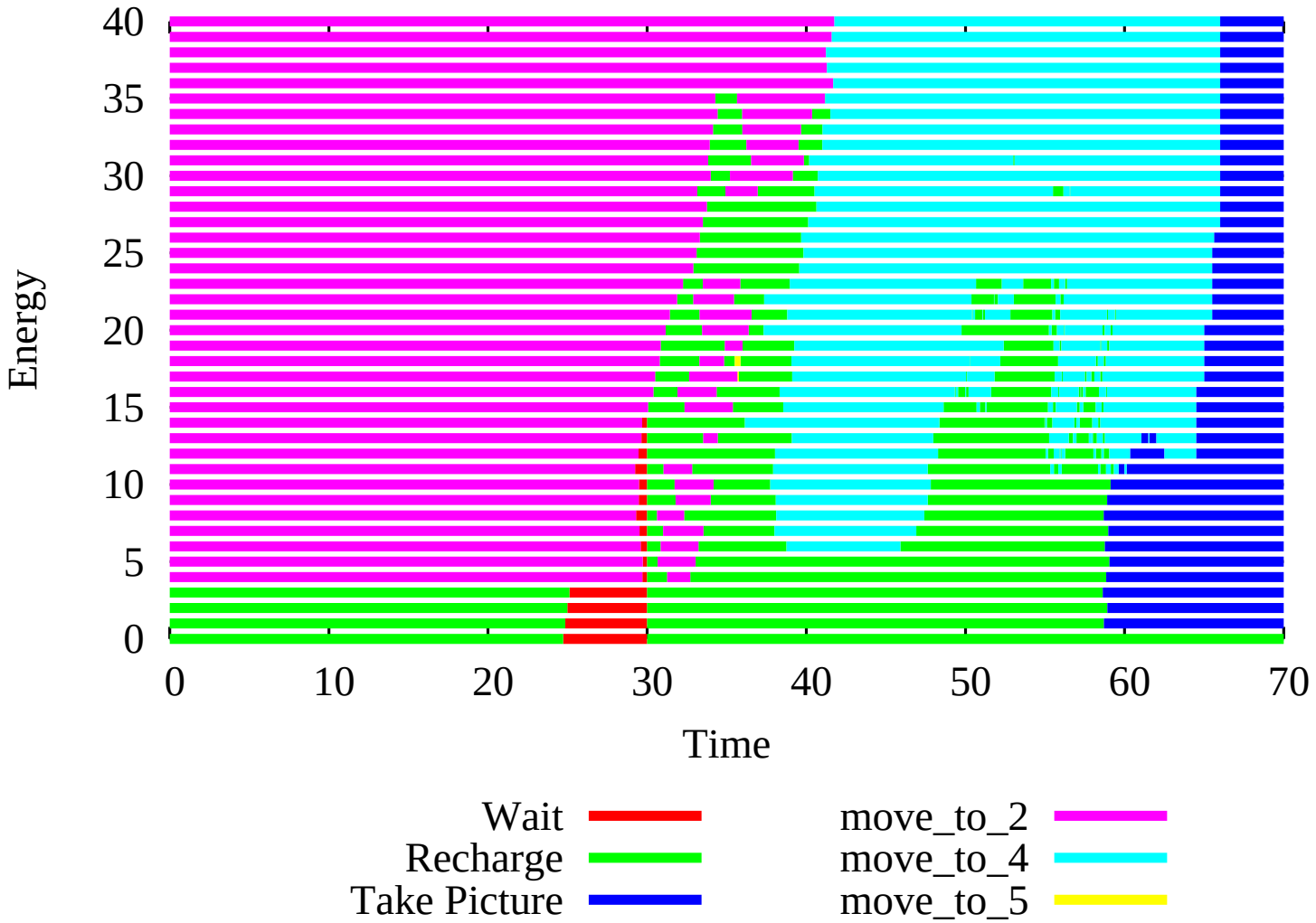
Mars rover policy

V and π in $p = 3$ when no goals have been completed yet.



Mars rover policy

π in $p = 3$ when no goals have been completed yet — 2D view.



Contributions

- XMDP optimality conditions and equations.
- Specific case of TMDPs.
- Extending exact resolution of TMDPs.
- $TMDP_{poly}$ allows better resolution of generalized piecewise polynomial TMDPs (including the exact case).

Optimal value function and policy

Existence of optimality conditions and an optimality equation on V and π for continuous observable time, discrete event stochastic processes.

$$V^* = LV^*$$

$$\pi^* = \underset{a(x)\in A(X)}{\operatorname{argmax}} \left\{ r(s,t,a(x)) + \int_{\substack{t'\in\mathbb{R} \\ s'\in S}} \gamma^{t'-t} p(s',t'|s,t,a(x)) V^*(s',t') ds' dt' \right\}$$

Contributions

- XMDP optimality conditions and equations.
- Specific case of TMDPs.
- Extending exact resolution of TMDPs.
- $TMDP_{poly}$ allows better resolution of generalized piecewise polynomial TMDPs (including the exact case).

TMDP hypothesis

TMDPs are XMDPs with specific hypothesis and a total reward criterion.

Contributions

- XMDP optimality conditions and equations.
- Specific case of TMDPs.
- Extending exact resolution of TMDPs.
- $TMDP_{poly}$ allows better resolution of generalized piecewise polynomial TMDPs (including the exact case).

Exact resolution conditions

Conditions for exact resolution of TMDPs can be slightly extended.

$$\left.\begin{array}{l} P_\mu \in \mathscr{DP}_A \\ r_i \in \mathscr{P}_B \\ L \in \mathscr{P}_C \end{array}\right\} \Rightarrow \left\{\begin{array}{l} P_\mu \in \mathscr{DP}_{-1} \\ r_i \in \mathscr{P}_4 \\ L \in \mathscr{P}_0 \end{array}\right.$$

But practical resolution call for approximation.

Contributions

- XMDP optimality conditions and equations.
- Specific case of TMDPs.
- Extending exact resolution of TMDPs.
- $TMDP_{poly}$ allows better resolution of generalized piecewise polynomial TMDPs (including the exact case).

$TMDP_{poly}$ in a nutshell

$TMDP_{poly}$: { Analytical polynomial calculations; L_∞-bounded error projection; Prioritized Sweeping for TMDPs }

- Analytical operations: option for representing continuous quantities.
- Approximation makes resolution possible.
- Asynchronous VI makes it faster.

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?

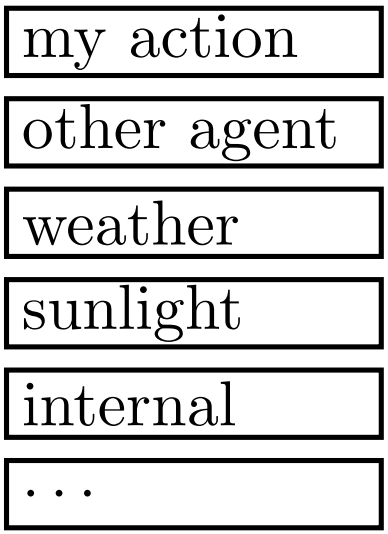
Concurrent exogeneous events

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

Concurrent exogeneous events

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

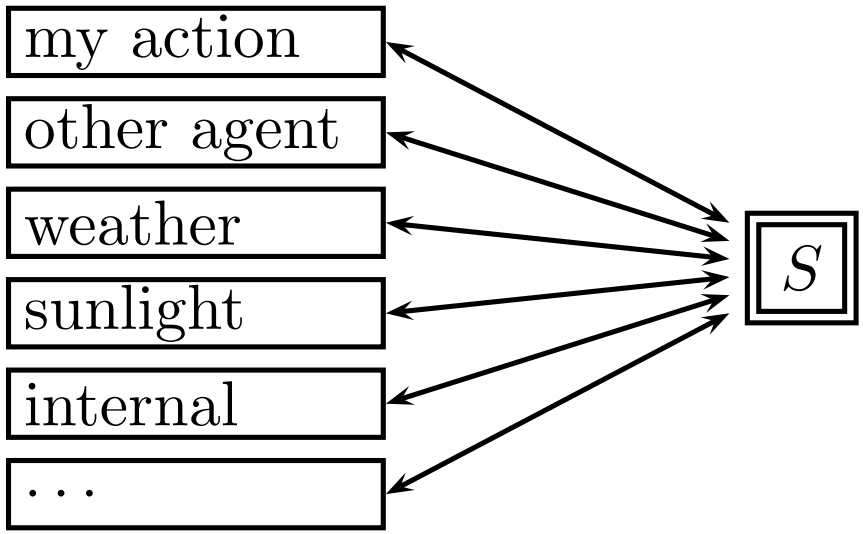
Aggregating the contribution of concurrent temporal processes. . .



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

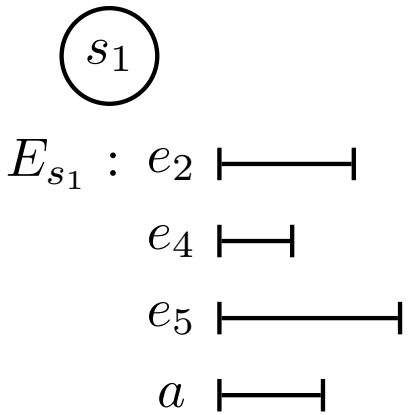
E	Set of events.
$A \subset E$	Subset of controlable events (actions).
$f(c_e \mid s, e)$	Duration model of event e.
$p(s' \mid s, e, c_e)$	Transition model of event e.

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.

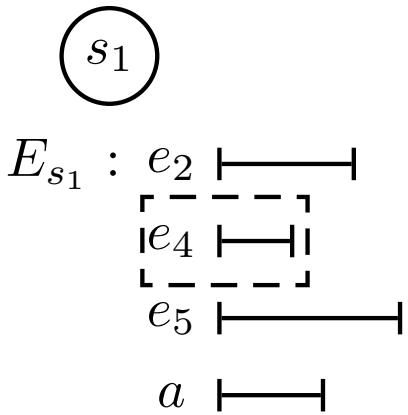


GSMDPs

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.

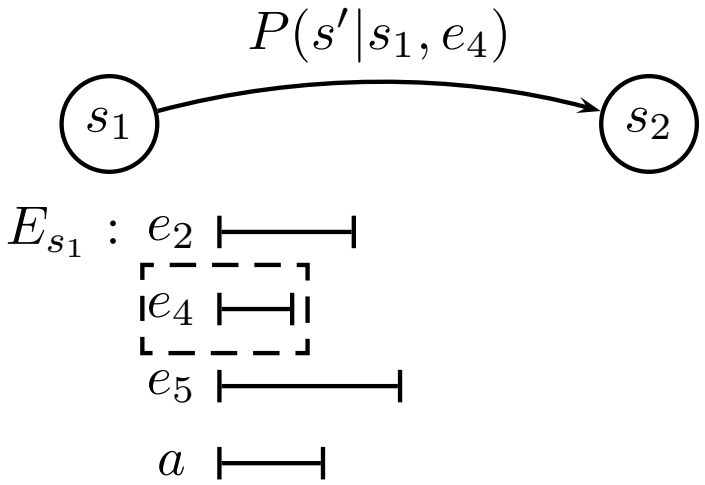


GSMDPs

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.

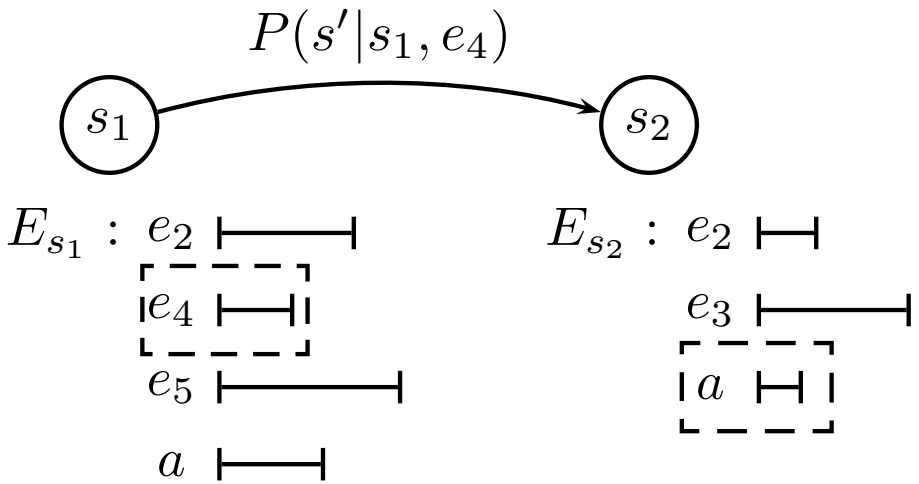


GSMDPs

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.

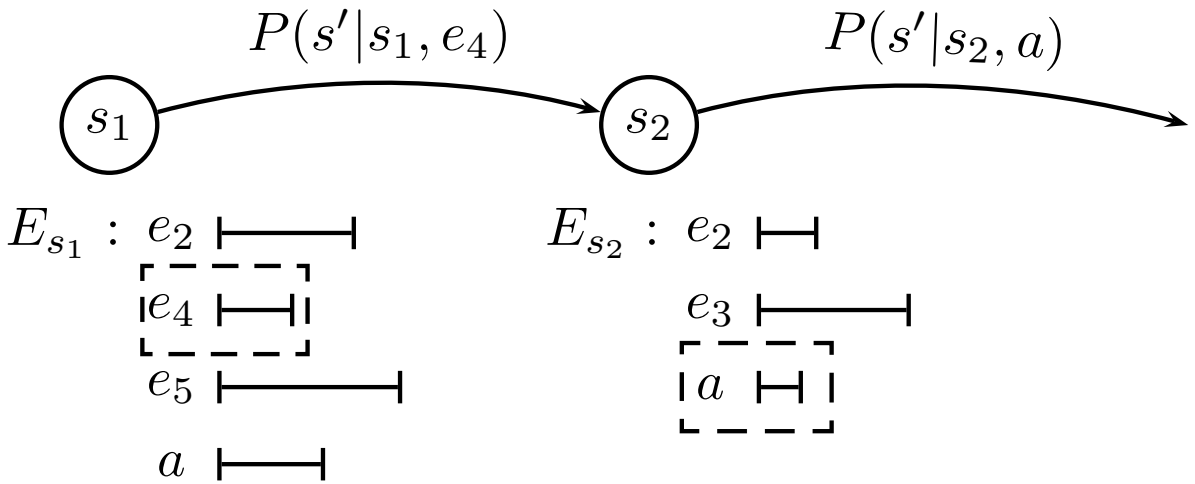


GSMDPs

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).

Properties

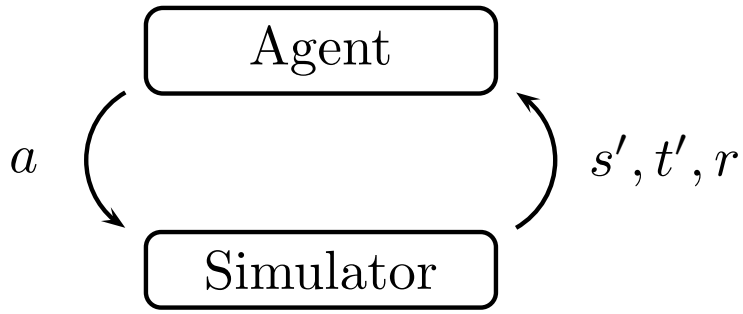
Remark

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?

Illustration

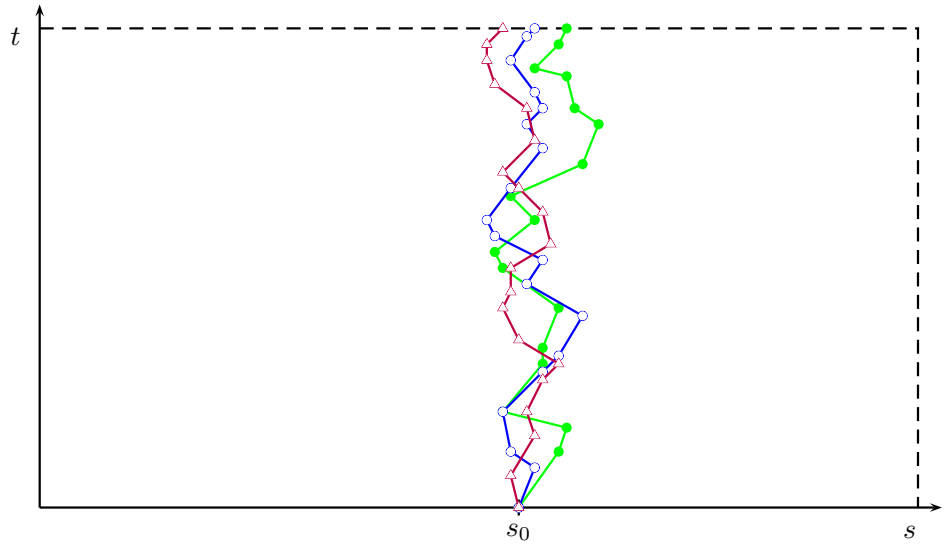
This approach is motivated by
problems such as the “subway problem” with
large, hybrid state spaces, many concurrent events,
for which a global model is not available.

Illustration

This approach is motivated by
problems such as the “subway problem” with
large, hybrid state spaces, many concurrent events,
for which a global model is not available.

Exploiting info from
episodes?

episode = observed
simulated trajectory through
the state space.



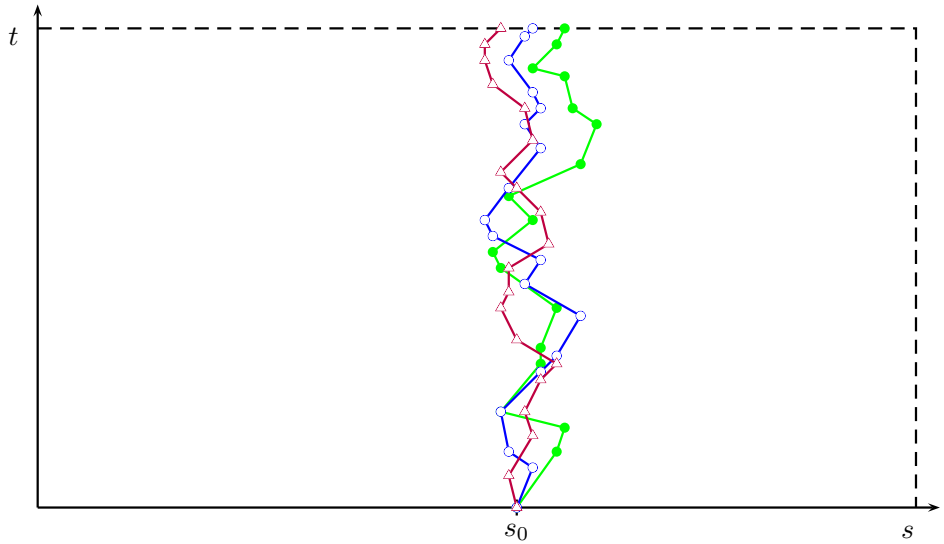
Illustration

Our 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

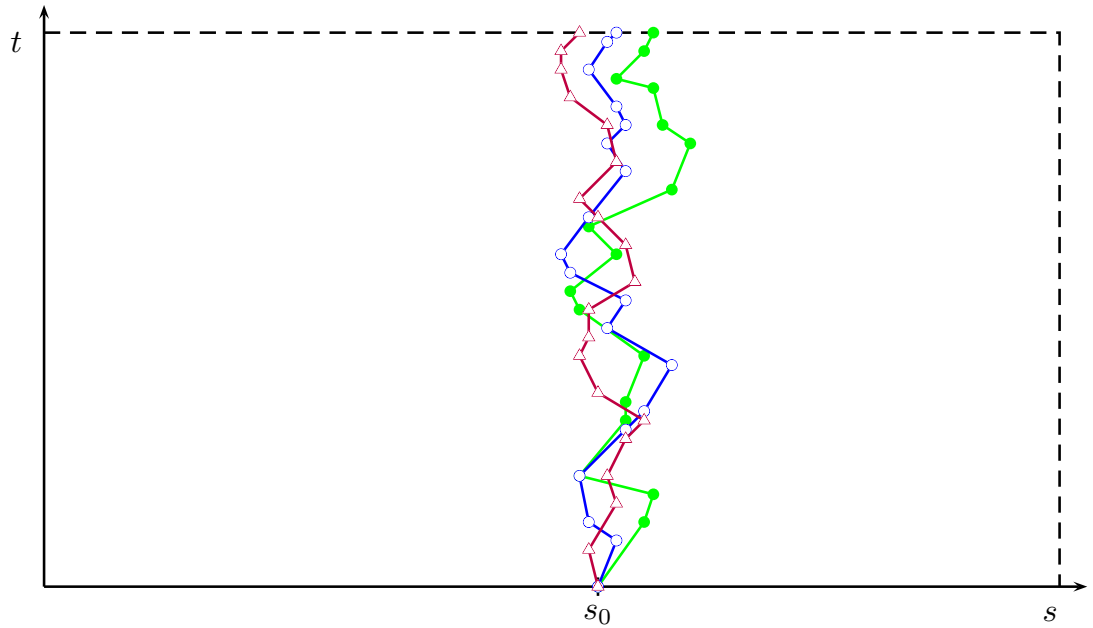
Once an improving action a is found in (s,t), the next state (s',t') picked for Bellman backup is chosen by applying a.
Observable time $\Rightarrow$ this (s',t') is picked according to $P(s',t'|s,t,\pi_n)$.

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

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

$$\Downarrow$$

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

$$\Downarrow$$

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

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 $t' > T$

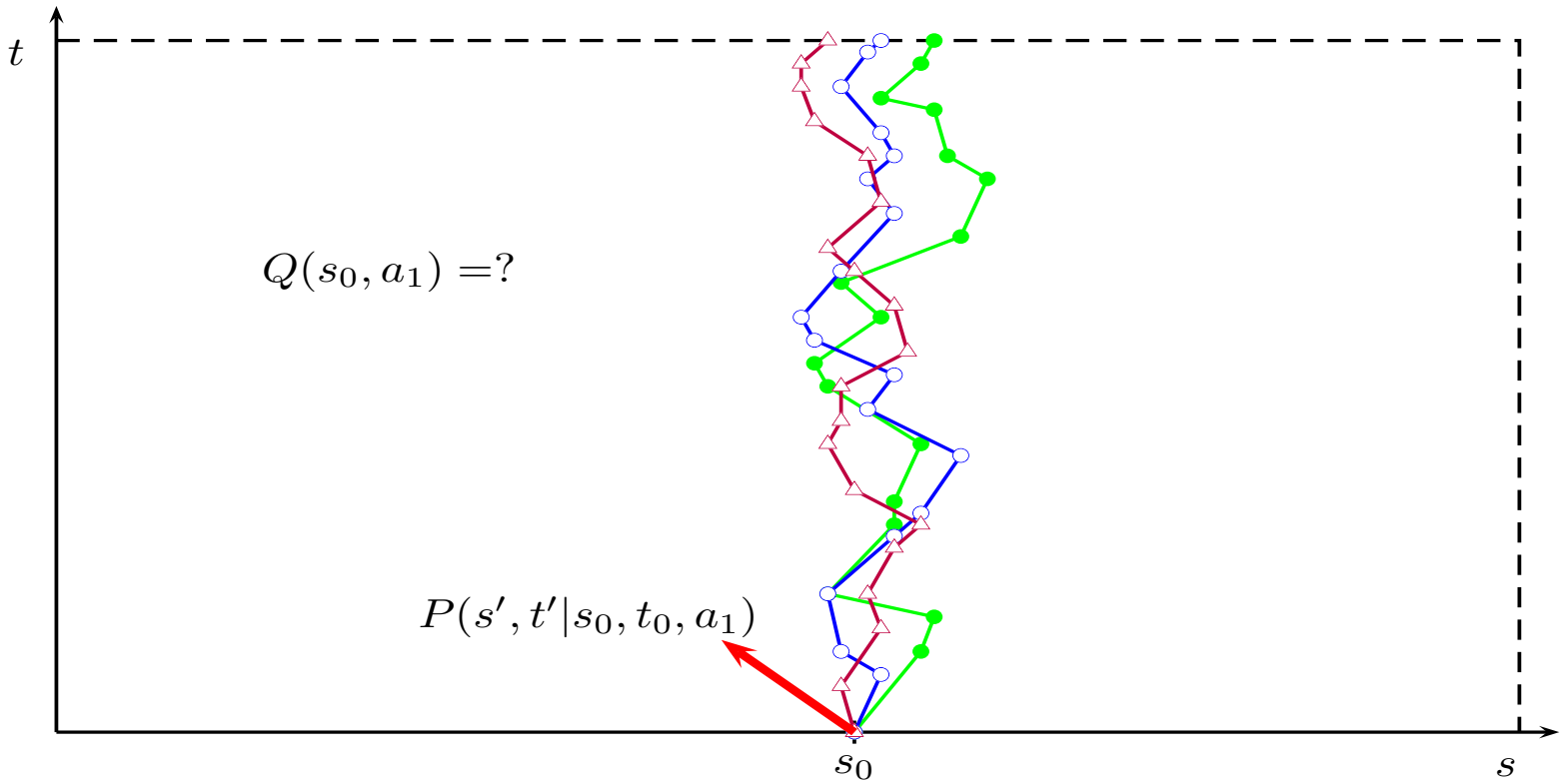
Compute $\tilde{V}^{\pi_{n+1}}$ for the last(s) episode(s)

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)?

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

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

iATPI

iATPI

iATPI: $\begin{cases} \text{Asynchronous policy iteration for greedy search} \\ \text{Time-dependency \& Monte-Carlo sampling} \\ \text{Local policies and values via confidence functions} \end{cases}$

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

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.

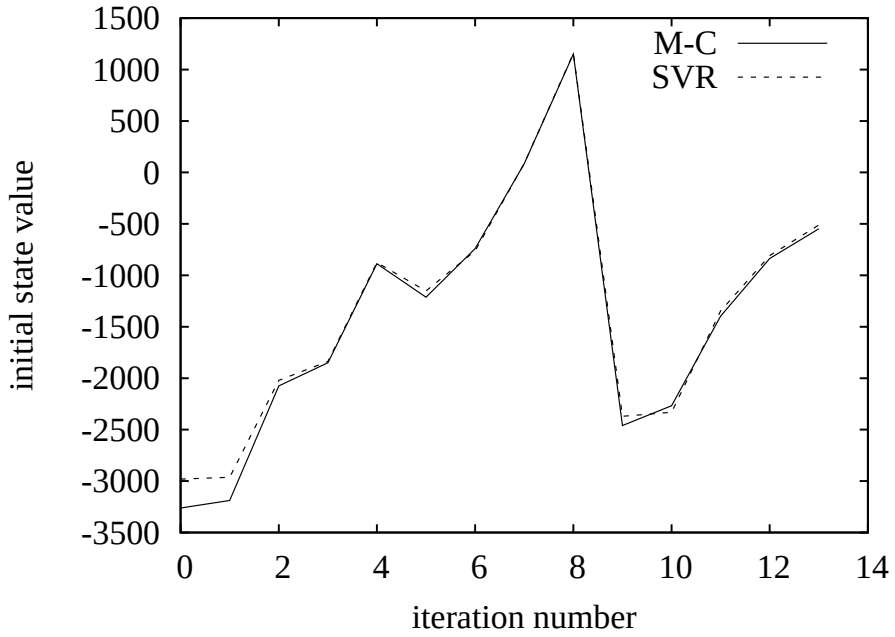
Preliminary results with *iATPI*

Preliminary results on ATPI and the subway problem:

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

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.

Perspectives for *iATPI*

iATPI is ongoing work
$\longrightarrow$ no hasty conclusions

Current work: extensive testing of the algorithm 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

- 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.

Contributions

- 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.

Summarizing the work done

Three ways of reading the thesis:

Modeling of temporal stochastic decision processes:

implicit-event (extended TMDP)
and
explicit-event (observable time GSMDP)

Theory General framework of XMDPs, optimality conditions and equations.

Algorithms for time-dependent policy search:

model-based asynchronous value iteration ($TMDP_{poly}$)
and
model-free local search for policy iteration (*iATPI*).

Thank you for your attention!

International Conferences

Rachelson, E., Teichteil, F., and Garcia, F. (2007a).
Temporal coordination under uncertainty: initial results for the two agents case.
In *ICAPS Doctoral Consortium*.

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*.

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*.

French-speaking Conferences

Rachelson, E., Fabiani, P., Farges, J.-L., Teichteil, F., and Garcia, F. (2006).
Une approche du traitement du temps dans le cadre MDP : trois méthodes de découpage de la droite temporelle.
In *Journées Françaises Planification Décision Apprentissage*.

Rachelson, E., Teichteil, F., and Garcia, F. (2007b).
XMDP : un modèle de planification temporelle dans l'incertain à actions paramétriques.
In *Journées Françaises Planification Décision Apprentissage*.

Rachelson, E., Fabiani, P., and Garcia, F. (2008a).
Un Algorithme Amélioré d'Itération de la Politique Approchée pour les Processus Décisionnels Semi-Markoviens Généralisés.
In *Journées Françaises Planification Décision Apprentissage*.

Rachelson, E., Fabiani, P., Garcia, F., and Quesnel, G. (2008b).
Une Approche basée sur la Simulation pour l'Optimisation des Processus Décisionnels Semi-Markoviens Généralisés (english version).
In *Conférence Francophone sur l'Apprentissage Automatique*.
Best student paper, awarded by AFIA.

Talks and presentations

ONERA DCSD, UR-CD, Toulouse (April 2006).
Planification dans l'incertain — Introduire une variable temporalle continue.

INRA-BIA, Toulouse (May 25th, 2007).
Planifier en fonction du temps dans le cadre MDP.

ONERA DCSD, UR-CD, Toulouse (February 3rd, 2008)
Formalisation et résolution de problèmes de Markov temporels par couplage avec VLE.
Coupled with "Multi-modélisation et simulation : la plate-forme VLE" by G. Quesnel.

Intelligent Systems Laboratory, Technical University of Crete (July 29th, 2008)
Simulation-based Approximate Policy Iteration for Generalized Semi-Markov Decision Processes.

Teaching activities

Non-linear optimization.
lecturing (2007, 2008), tutoring (2006) — ENAC

Probabilities and Harmonic analysis, introduction module.
lecturing (2006) — SUPAERO

Reinforcement Learning and Dynamic Programming
tutoring (2008) — ISAE-SUPAERO

Stochastic Processes
tutoring (2007, 2008) — SUPAERO then ISAE-SUPAERO

Optimization and numeric computation
tutoring (2006, 2007, 2008) — SUPAERO then ISAE-SUPAERO

MatLab introduction
tutoring (2006, 2007) — SUPAERO

Harmonic analysis
tutoring (2006) — SUPAERO

Algorithmic perspectives

Model based approaches:

- Biasing PS in $TMDP_{poly}$ to obtain better convergence speed.
- Better algorithms (and implementation) for *POLYTOOLS* .
- $XMDP_{poly}$?
- Policy Iteration for XMDPs? TMDPs?
- ...

The *iATPI* perspective:

- Discounted criteria?
- Statistical learning for *iATPI*, sound algorithms and efficient implementations.
- Avoiding local minima with *iATPI*.
- ...

Perspectives: models and foundations

Time and stochastic processes:

- Foundations of time-explicit decision processes: lifting the mathematical assumptions in the XMDP model
- Relation between GSMDP and POMDP: defining a belief state from the (s, c) state

Exploration vs exploitation?

How does *iATPI* compare to other methods concerning the exploration vs. exploitation trade-off?

- Automated balancing through “optimism”:
  - “Optimism in the face of uncertainty”
  - Rmax
  - Admissible heuristics

  Encourages early exploration.
  Automatically balances the trade-off. $\Rightarrow$ Very good for online learning.
- *iATPI* suggests an “offline/online” alternative:
  - abandon global exploration for incremental, episode-based exploration.
  - explore what we need locally for evaluation, use it for local improvement, then look outside.

  No exploration “enc/discouragement”.
  Local search idea $\Rightarrow$ Good for “cautious” search?

Other illustrations of GSMDPs

Should we open more lines ?

Other illustrations of GSMDPs

Airplanes taxiing management

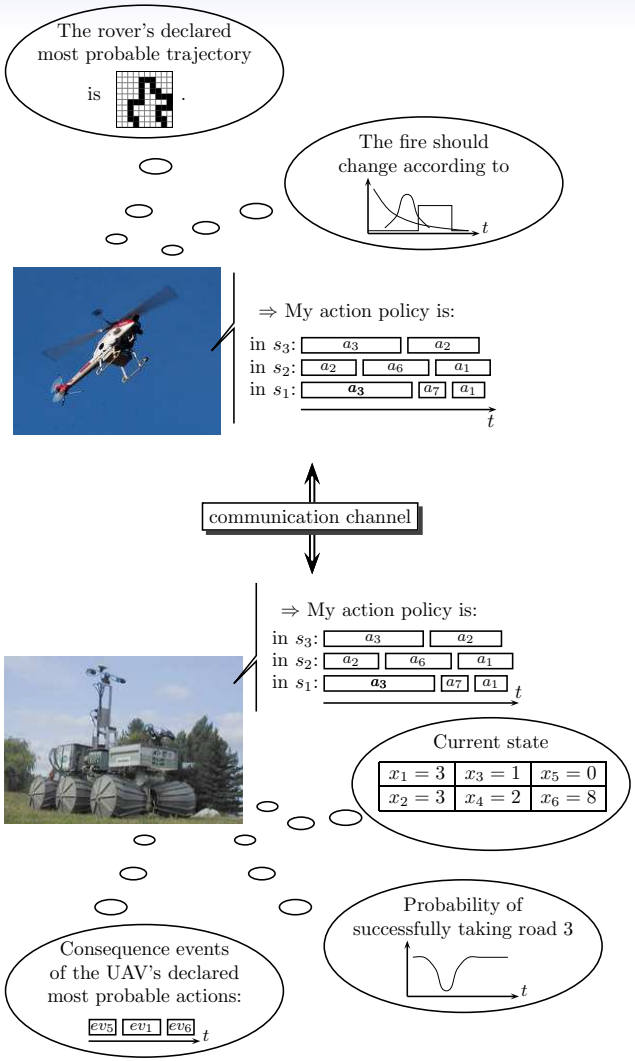
Other illustrations of GSMDPs

Adding or removing trains ?

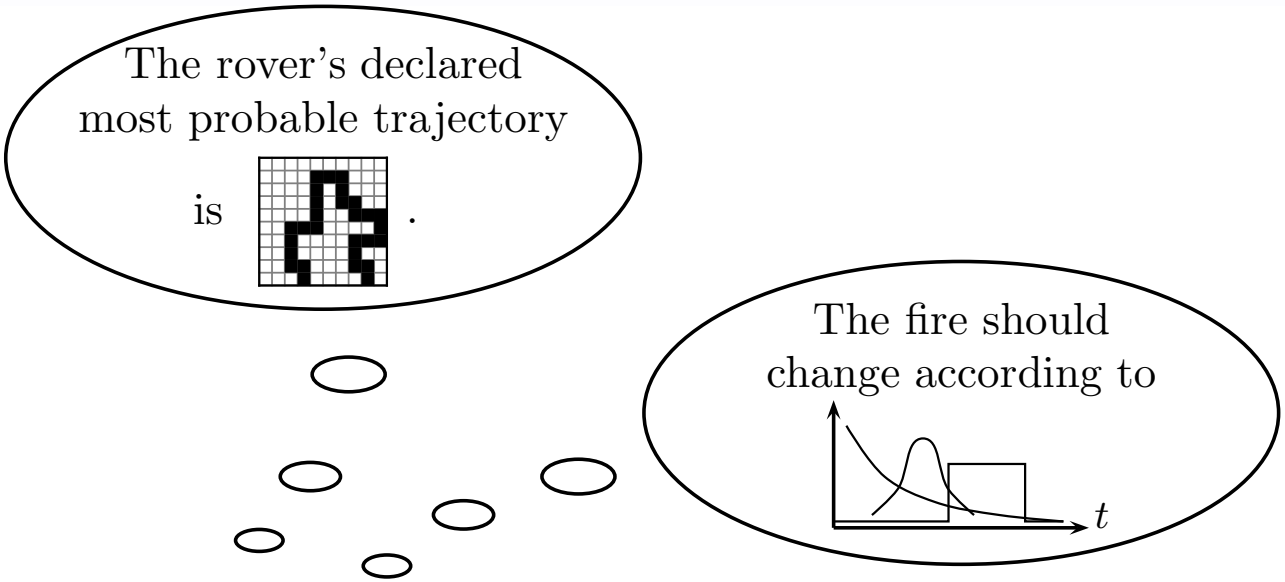
Other illustrations of GSMDPs

Onboard planning for coordination

Other illustrations of GSMDPs



Other illustrations of GSMDPs



$\Rightarrow$ My action policy is:

in s_3: a_3 a_2

in s_2: a_2 a_6 a_1

in s_1: a_3 a_7 a_1

t

Other illustrations of GSMDPs

$\Rightarrow$ My action policy is:

in s_3: a_3 a_2

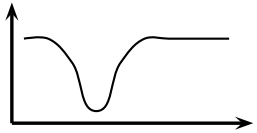
in s_2: a_2 a_6 a_1

in s_1: a_3 a_7 a_1 t

Current state

$x_1 = 3$	$x_3 = 1$	$x_5 = 0$
$x_2 = 3$	$x_4 = 2$	$x_6 = 8$

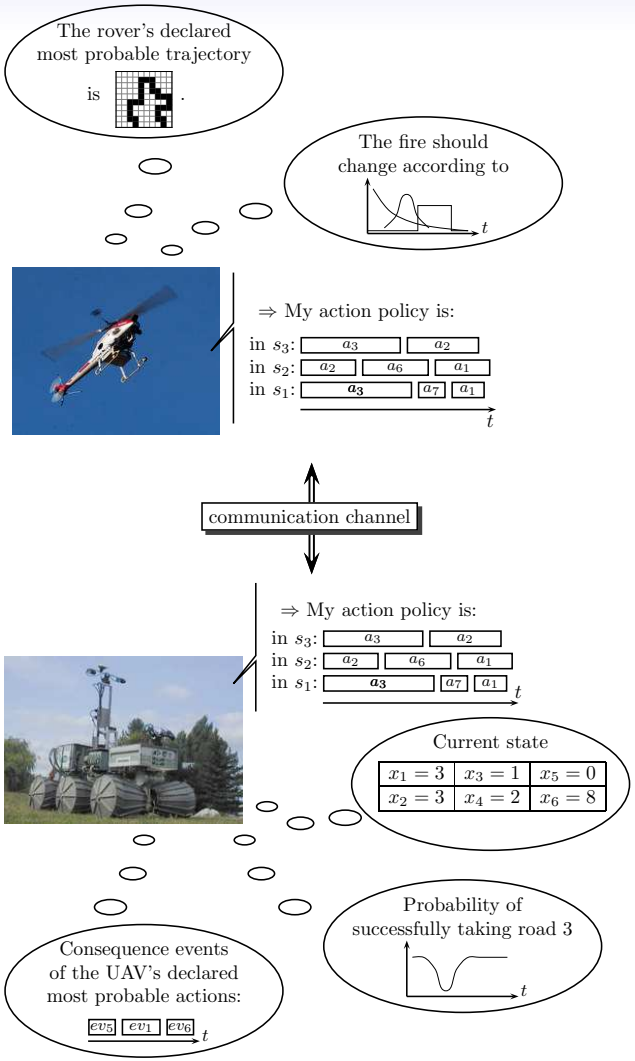
Probability of successfully taking road 3



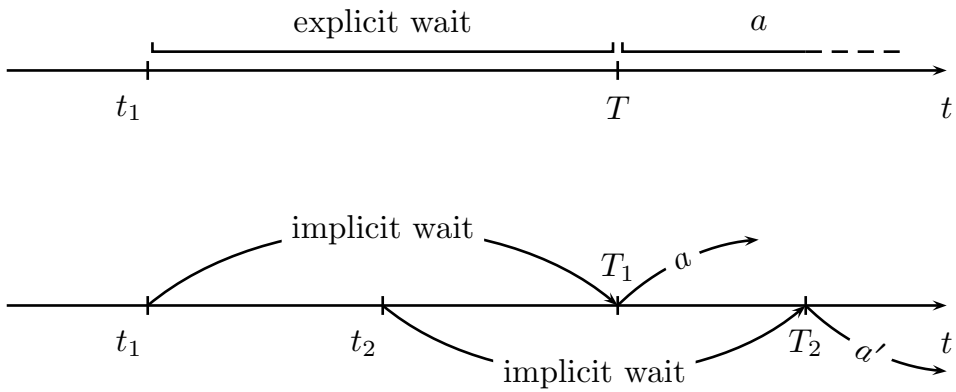
Consequence events of the UAV's declared most probable actions:

ev_5 ev_1 ev_6 t

Other illustrations of GSMDPs



Waiting or being idle?



being idle $\longrightarrow$ let the system change **continuously**

discrete event process $\longrightarrow$ **stepwise** changes in the system

From the execution point of view:

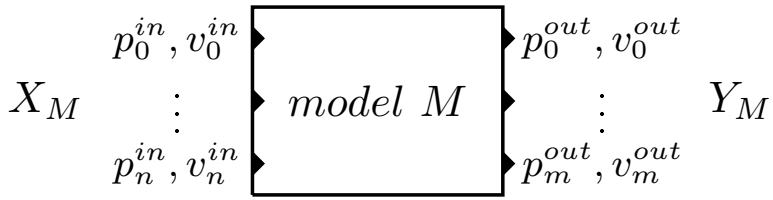
being idle $\longrightarrow$ let the system change by itself

$\Rightarrow$ interest of W function or explicit-event representations (a_∞).

But this is different from the TMDP's *wait*.

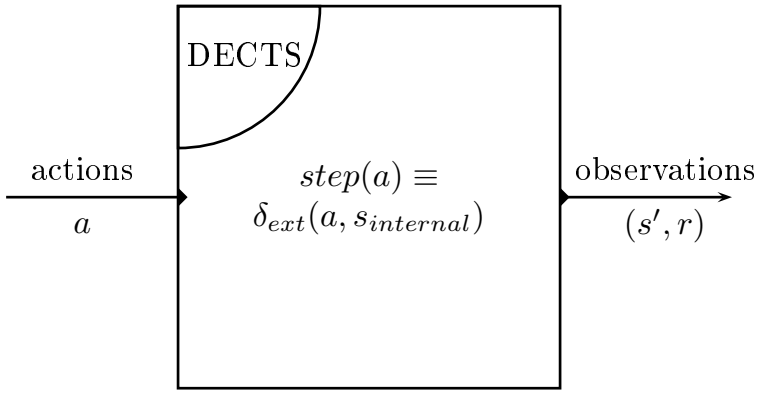
DECTS

GSMPs = concurrent temporal stochastic processes
DEVS = generic description of discrete events systems



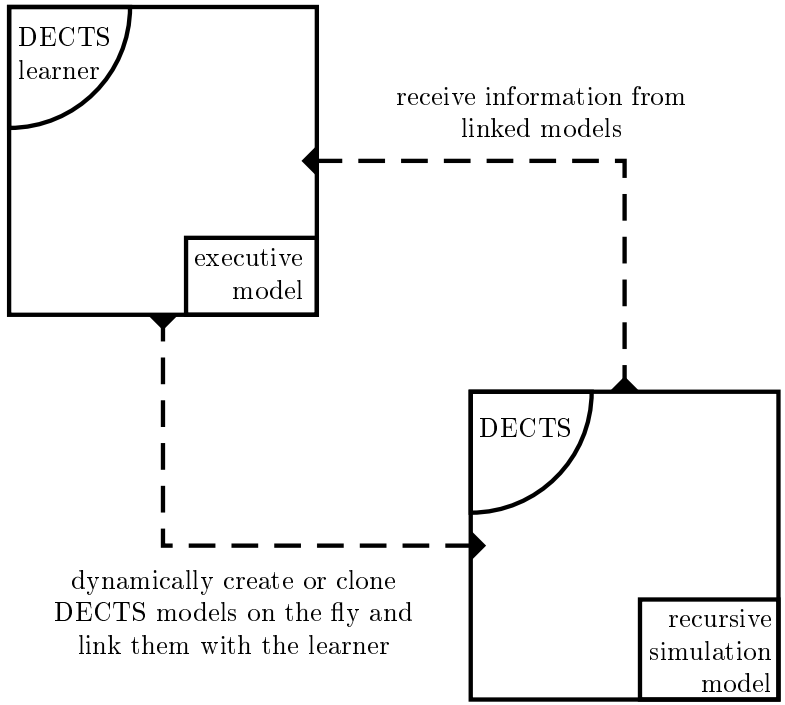
DECTS

Temporal decision process $\equiv$ input port *a*



DECTS

An optimization process $\equiv$ sequence of operations involving experiments with a DECTS model.



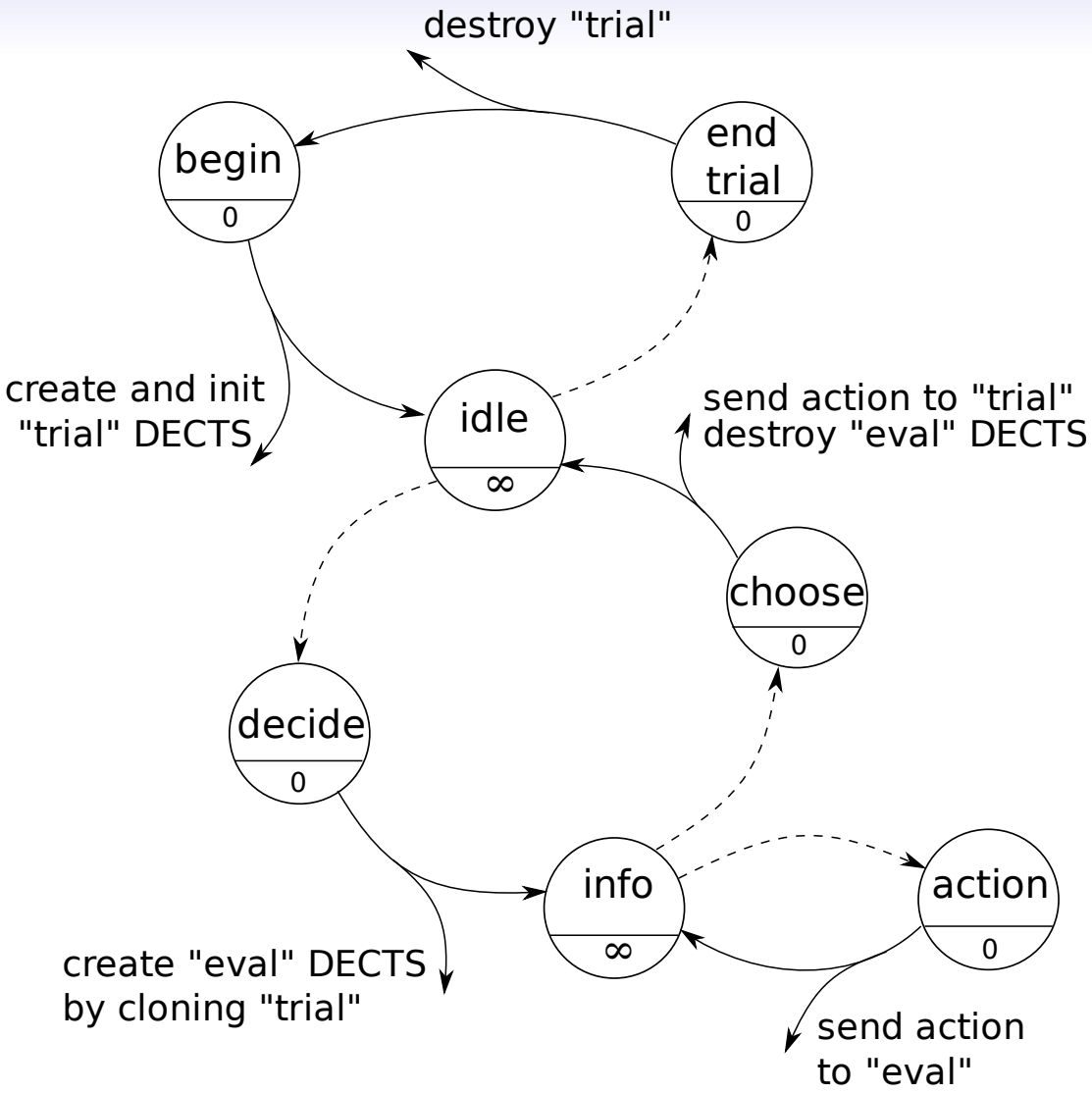
DECTS

A DECTS learner is an executive (high-level) discrete events system, creating and controling a set of DECTS experiments.

It has internal **decision objects** (policies, values, etc.)

Nota: Actor-Critic vs. DECTS? Actor-Critic is the architecture of the DECTS learner's decision objects.

iATPI as a DECTS



Database *iATPI*

H_0 hypothesis

The asymptotical convergence of $\tilde{Q}_n(s,a)$ towards a distribution $\mathcal{N}(Q(s,a),\sigma)$ is quick.

Theorem (PAC-bound guarantee)

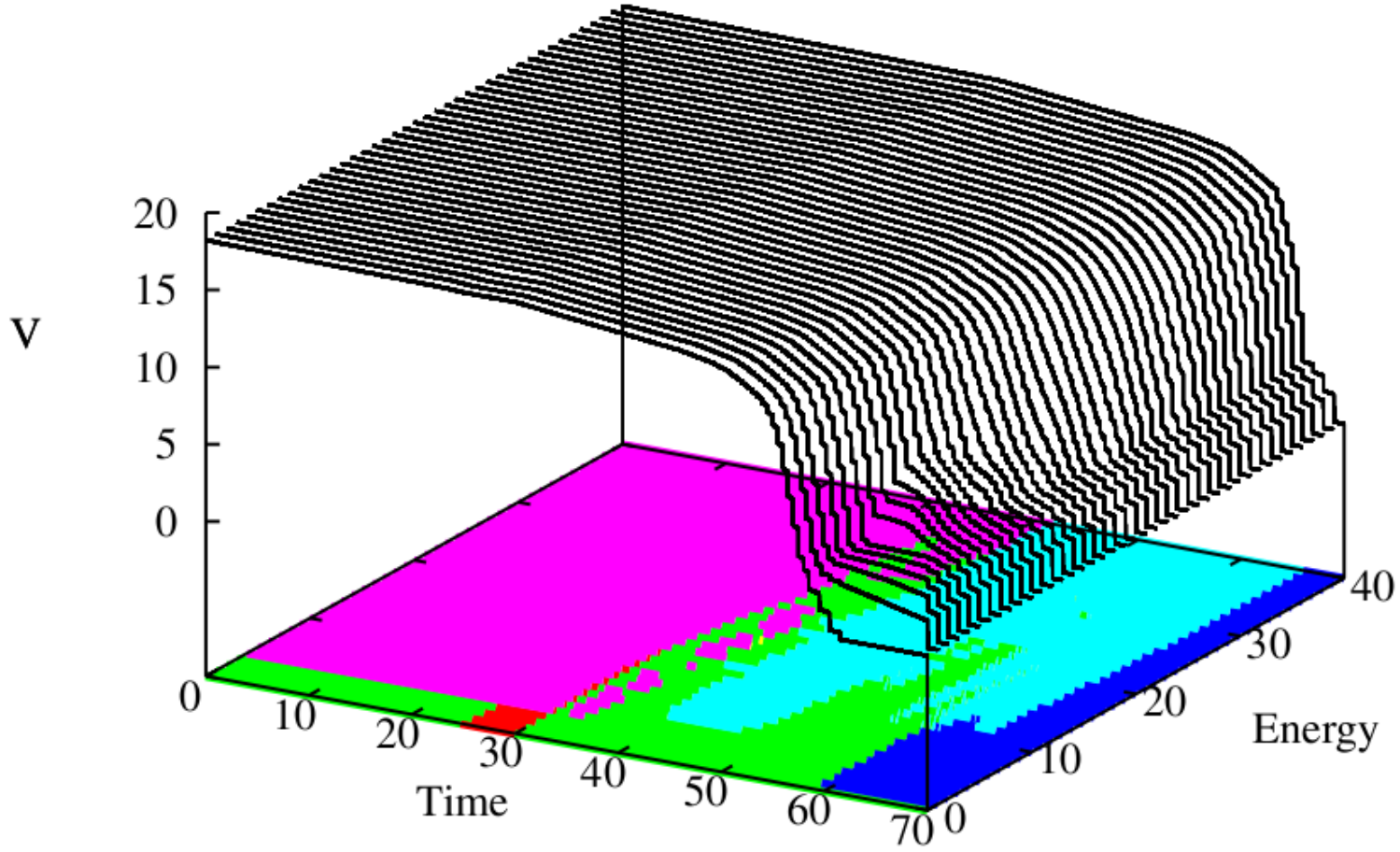
$Q_n(s,a)$ is an ε-estimate of $Q(s,a)$ with probability $p = \mathrm{erf}\left(\frac{\varepsilon\sqrt{n}}{\sigma_n^Q\sqrt{2}}\right)$

In practice

N_a	Stop the rollouts in (s,a) whenever $\sigma_n^Q \leq \frac{\varepsilon\sqrt{n}}{\mathrm{erf}^{-1}(p)\sqrt{2}}$.
$N_{episodes}$	Stop running episodes for the current policy when the $Q(s_0,a^*)$ has σ_n^Q lower than the bound.
rollouts	Early stopping if a state with $\sigma_n^M \leq \frac{\varepsilon}{\mathrm{erf}^{-1}(p)\sqrt{2}}$ is encountered.

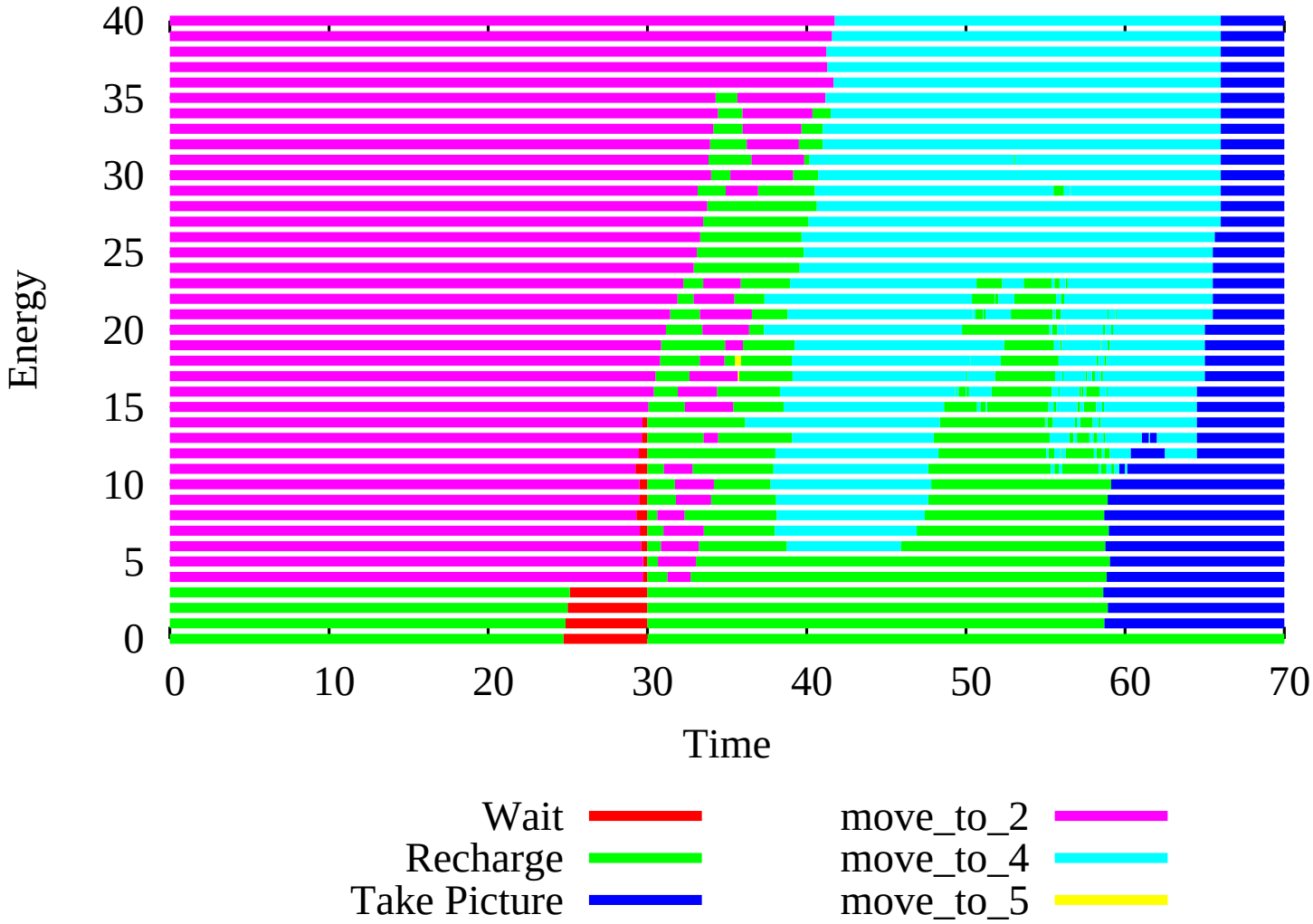
Mars rover

V and π in $p = 3$ when no goals have been completed yet.



Mars rover

π in $p = 3$ when no goals have been completed yet — 2D view.



Analytical resolution of GSMDPs

[Younes and Simmons, 2004] $\longrightarrow$ approximate all duration models $f(\tau|s,e)$ by chains of exponential distributions.

Phase-type distributions.
Introduce abstract states for the nodes in phase-type distr.

Memoryless exponential distributions turn the GSMDP into a CTMDP.

Resolution by **uniformization**.

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 and POMDPs

Observations and hidden process

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

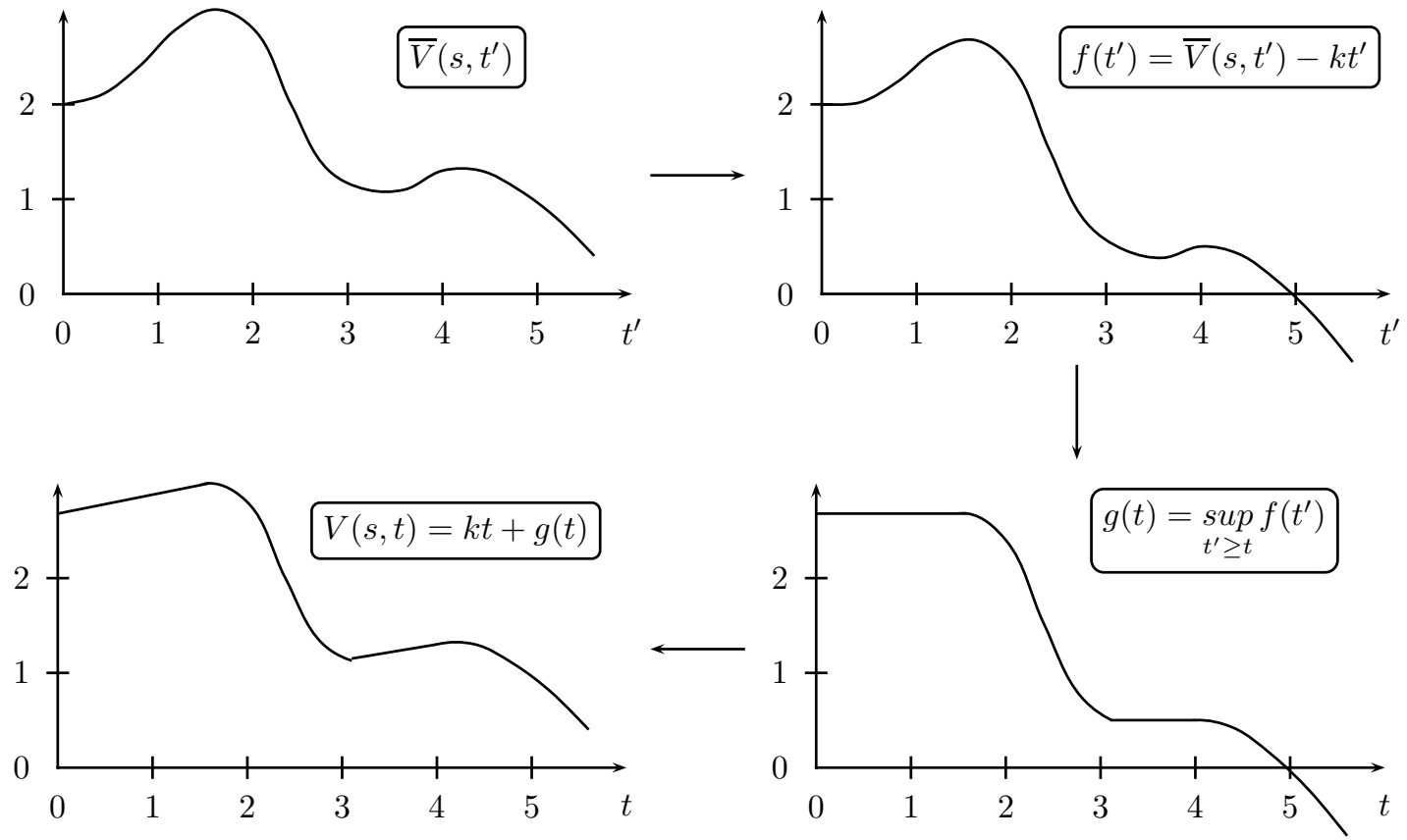
$$\begin{cases} (s,c) & \leftrightarrow & \text{hidden state} \\ s & \leftrightarrow & \text{observations} \end{cases}$$

Working hypothesis

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

iATPI $\longrightarrow$ simulation-based, asynchronous policy iteration for stochastic shortest path POMDPs.

Computing V from $\overline{V}$



Asynchronous Policy Iteration

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^*.

Examples

Unordered V-backups (alternate π-backups)	VI
Asynchonous V-backups (alternate π-backups)	Async VI, Prio. Sweeping, RTDP, ...
Unordered, alternate 1 π-backup / m V-backups	(Modified) PI

iATPI

Main loop(π_0 or $\tilde{V}_0$, s_0, t_0, T, $N_{episodes}$)

iATPI

Main loop(π_0 or $\tilde{V}_0$, s_0, t_0, T, $N_{episodes}$)

loop

ValueSet.reset(), *ActionSet*.reset()

for $i = 1$ to $N_{episodes}$ **do**

$\tilde{V}_n, C^{\tilde{V}_n}, \pi_n, C^{\pi_n} \leftarrow \text{train}(\mathit{ValueSet}, \mathit{ActionSet})$

iATPI

Main loop(π_0 or $\tilde{V}_0$, s_0, t_0, T, $N_{episodes}$)

loop
  $ValueSet$.reset(), $ActionSet$.reset()
  for $i = 1$ to $N_{episodes}$ **do**
    σ.reset()
    $episode$.reset(s_0, t_0)
    while $t < T$ **do**
      $a = episode$.bestAction()
      $episode$.activateEvent(a)
      $((s', t'), r) \leftarrow episode$.step()
      σ.add($(s, t), a, r$)
      $t \leftarrow t'$

  $\tilde{V}_n, C^{\tilde{V}_n}, \pi_n, C^{\pi_n} \leftarrow$ train($ValueSet, ActionSet$)

iATPI

Main loop(π_0 or $\tilde{V}_0$, s_0, t_0, T, $N_{episodes}$)

loop
  $ValueSet$.reset(), $ActionSet$.reset()
  for $i = 1$ to $N_{episodes}$ **do**
    σ.reset()
    $episode$.reset(s_0, t_0)
    while $t < T$ **do**
      $a = episode$.bestAction()
      $episode$.activateEvent(a)
      $((s', t'), r) \leftarrow episode$.step()
      σ.add($(s, t), a, r$)
      $t \leftarrow t'$
    ($ValueSet, ActionSet$).merge(convert(σ))
  $\tilde{V}_n, C^{\tilde{V}_n}, \pi_n, C^{\pi_n} \leftarrow$ train($ValueSet, ActionSet$)

iATPI

episode.**bestAction**()

for $a \in A_s$ **do**
  $\tilde{Q}(a) = 0$, $n = 0$
  while not enough samples for $\tilde{Q}(a)$ **do**
    $\tilde{Q}(a) \leftarrow \tilde{Q}(a) + \frac{1}{n}(episode.\text{rollout}(a) - \tilde{Q}(a))$
return $\arg\max_{a \in A} \tilde{Q}(a)$

iATPI

episode.**rollout**(a)

rolloutEpisode(*episode*)
rolloutEpisode.activateEvent(a)
$((s', t'), r) \leftarrow$ *rolloutEpisode*.step()

iATPI

episode.**rollout**(a)

rolloutEpisode(*episode*)
rolloutEpisode.activateEvent(a)
$((s', t'), r) \leftarrow$ *rolloutEpisode*.step()
if $C^{\tilde{V}_{n-1}}(s', t') = \top$ **then**
    return $r + \tilde{V}_{n-1}(s', t')$

iATPI

episode.**rollout**(a)

rolloutEpisode(*episode*)
rolloutEpisode.activateEvent(a)
$((s', t'), r) \leftarrow$ *rolloutEpisode*.step()
if $C^{\tilde{V}_{n-1}}(s', t') = \top$ **then**
    return $r + \tilde{V}_{n-1}(s', t')$
else
    $Q \leftarrow r,\ s \leftarrow s',\ \sigma_r \leftarrow \emptyset$
    while rollout unfinished **do**
        $a = \pi_{n-1}(s)$
        rolloutEpisode.activateEvent(a)
        $((s', t'), r) \leftarrow$ *rolloutEpisode*.step()
        $Q \leftarrow Q + r$
        σ_r.add($(s, t), r$)

iATPI

episode.**rollout**(a)

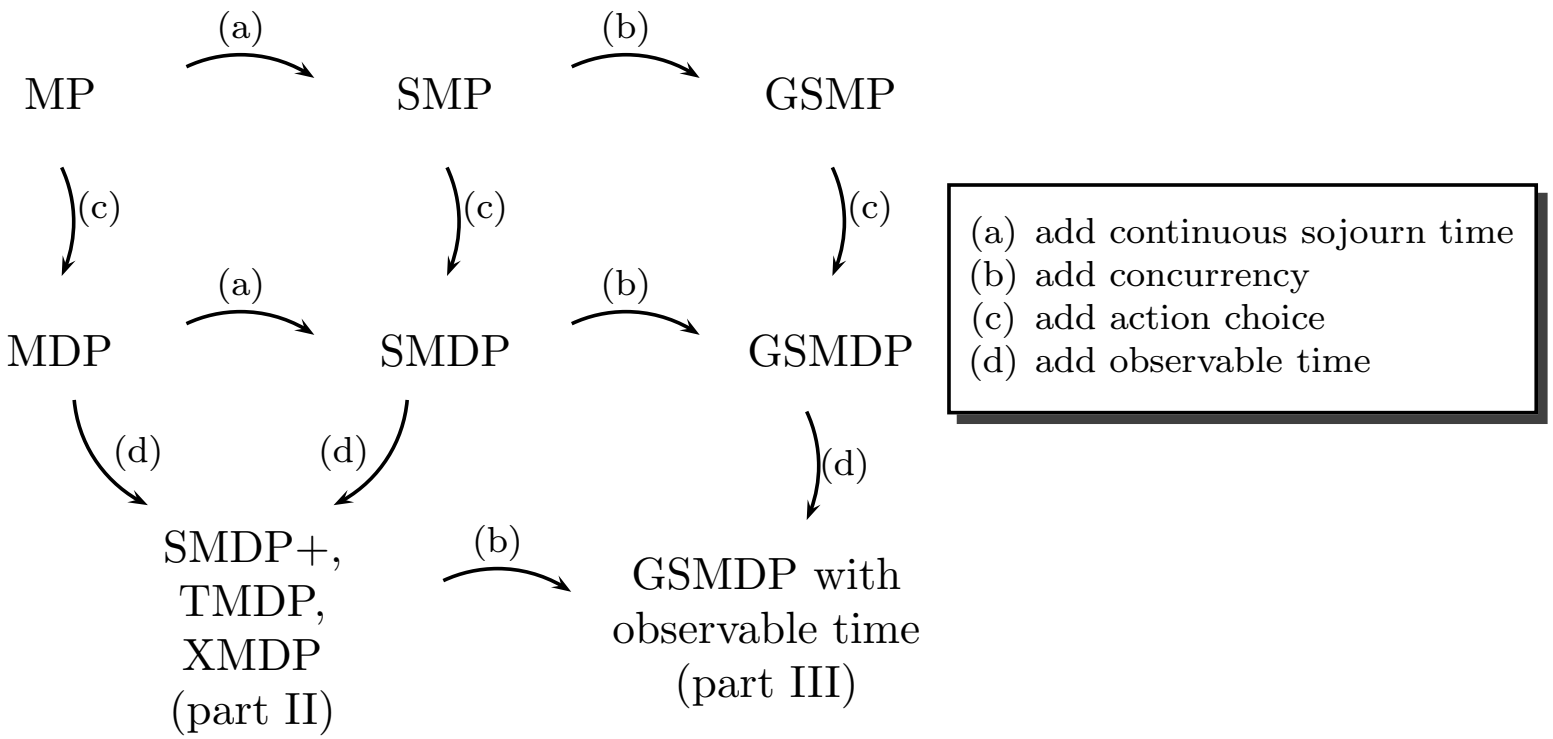
rolloutEpisode(*episode*)
rolloutEpisode.activateEvent(a)
$((s', t'), r) \leftarrow$ *rolloutEpisode*.step()
if $C^{\tilde{V}_{n-1}}(s', t') = \top$ **then**
  return $r + \tilde{V}_{n-1}(s', t')$
else
  $Q \leftarrow r,\ s \leftarrow s',\ \sigma_r \leftarrow \emptyset$
  while rollout unfinished **do**
    $a = \pi_{n-1}(s)$
    rolloutEpisode.activateEvent(a)
    $((s', t'), r) \leftarrow$ *rolloutEpisode*.step()
    $Q \leftarrow Q + r$
    σ_r.add($(s, t), r$)
  $\tilde{V}_{n-1}, C^{\tilde{V}_{n-1}} \leftarrow$ incTrain(convert(σ_r))
  return Q

iATPI

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.

Models map


MP
SMP
GSMP
MDP
SMDP
GSMDP
SMDP+,
TMDP,
XMDP
(part II)
GSMDP with
observable time
(part III)
(a)
(b)
(c)
(d)
(a) add continuous sojourn time
(b) add concurrency
(c) add action choice
(d) add observable time