Simulation-based Approximate Policy Iteration for Generalized Semi-Markov Decision Processes

Emmanuel Rachelson ¹
Patrick Fabiani ¹
Frédérick Garcia ²

¹ONERA-DCSD
²INRA-BIA
Toulouse, France

ECAI08, July 23rd, 2008
Plan

1. Time and MDP: motivation and modeling
   - Examples
   - Problem features
   - GSMDP

2. Focusing Policy search in Policy Iteration
   - Policy Iteration algorithms
   - Asynchronous Dynamic Programming
   - Real-Time Policy Iteration

3. Dealing with large dimension, continuous state spaces
   - RL, Monte-Carlo sampling and Statistical Learning
   - The ATPI algorithm (naive version)
   - The ATPI algorithm (complete version)
Plan

1. Time and MDP: motivation and modeling
   - Examples
   - Problem features
   - GSMDP

2. Focusing Policy search in Policy Iteration
   - Policy Iteration algorithms
   - Asynchronous Dynamic Programming
   - Real-Time Policy Iteration

3. Dealing with large dimension, continuous state spaces
   - RL, Monte-Carlo sampling and Statistical Learning
   - The ATPI algorithm (naive version)
   - The ATPI algorithm (complete version)
Planning under uncertainty with time dependency.

→ planning to coordinate with an uncertain and unstationnary environment.
Planning under uncertainty with time dependency. → planning to coordinate with an uncertain and unstationary environment.

Should we open more lines?
Planning under uncertainty with time dependency.

→ planning to coordinate with an uncertain and unstationary environment.

Airplanes taxiing management
Planning under uncertainty with time dependency.
→ planning to coordinate with an uncertain and unstationnary environment.

Onboard planning for coordination
Planning under uncertainty with time dependency.
→ planning to coordinate with an uncertain and unstationary environment.

Adding or removing trains?
Main idea

Why is writing an MDP for the previous problems such a difficult task?

“Lots of things occur in parallel”

- concurrent phenomena
- partially controlable dynamics
Main idea

Why is writing an MDP for the previous problems such a difficult task?

“Lots of things occur in parallel”

- concurrent phenomena
- partially controllable dynamics
Main idea

Why is writing an MDP for the previous problems such a difficult task?

“Lots of things occur in parallel”

- concurrent phenomena
- partially controlable dynamics
Typical features

- Continuous time
- Hybrid state spaces
- Large state spaces
- Total reward criteria
- Long trajectories / long episodes
How do we model all this?
Time and MDP: motivation and modeling

Focusing Policy search in Policy Iteration

Dealing with large dimension, continuous state spaces

GSMDP, (Younes et al., 04)
GSMDP, (Younes et al., 04)

GSMP, (Glynn, 89)
Several semi-Markov processes affecting the same state space

→ \( \langle S, E, A, P, F, r \rangle \)

One process conditionned by the choice of the action undertaken
GSMDP, (Younes et al., 04)

GSMP, (Glynn, 89)
Several semi-Markov processes affecting the same state space

One process conditionned by the choice of the action undertaken

→ \langle S, E, A, P, F, r \rangle

\[ P(s'|s_1, e_4) \quad P(s'|s_2, a) \]

\[ E_{s_1} : \begin{aligned} & e_2 \quad e_4 \quad e_5 \quad a \\ & \end{aligned} \quad E_{s_2} : \begin{aligned} & e_2 \quad e_3 \quad a \\ & \end{aligned} \]
Controlling GSMDP

non-Markov behaviour!

→ no guarantee of an optimal Markov policy

(Younes et al., 04): approximate your model with phase-type (exponential) distributions.

Supplementary variables technique (Nilsen, 98).

Large dimension state spaces.

Our approach: no hypothesis, simulation-based API.
Controlling GSMDP

non-Markov behaviour!

→ no guarantee of an optimal Markov policy

(Younes et al., 04): approximate your model with phase-type (exponential) distributions.

Supplementary variables technique (Nilsen, 98).
Large dimension state spaces.

Our approach: no hypothesis, simulation-based API.
Controlling GSMDP

non-Markov behaviour!

→ no guarantee of an optimal Markov policy

(Younes et al., 04): approximate your model with phase-type (exponential) distributions.

Supplementary variables technique (Nilsen, 98).
Large dimension state spaces.

Our approach: no hypothesis, simulation-based API.
## Plan

1. **Time and MDP: motivation and modeling**
   - Examples
   - Problem features
   - GSMDP

2. **Focusing Policy search in Policy Iteration**
   - Policy Iteration algorithms
   - Asynchronous Dynamic Programming
   - Real-Time Policy Iteration

3. **Dealing with large dimension, continuous state spaces**
   - RL, Monte-Carlo sampling and Statistical Learning
   - The ATPI algorithm (naive version)
   - The ATPI algorithm (complete version)
Policy Iteration

Policy evaluation: $V^{\pi_n}$

One-step improvement: $\pi_{n+1}$
Policy Iteration

- performs search in policy space
- converges in less iterations than VI
- takes longer than VI
Policy Iteration

Approximate evaluation: $V^{\pi_n}$

One-step improvement: $\pi_{n+1}$
Asynchronous Dynamic Programming

Bellman backups can be performed in any order, the algorithm eventually reaches the optimal policy.

Example

Asynchronous Value Iteration

\[ V_{n+1}(s) \leftarrow \max_{a \in A} r(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V_n(s') \]

Some states can be updated several times before some others are updated for the first time.
Asynchronous Dynamic Programming

Bellman backups can be performed in any order, the algorithm eventually reaches the optimal policy.

Example

Asynchronous Policy Iteration

$$\pi_{n+1}(s) \leftarrow \arg \max_{a \in A} r(s, a) + \gamma \sum_{s' \in S} P(s'|s, a) V^{\pi_n}(s')$$

We can choose to update only some states before entering a new evaluation of $\pi$ step.
Learning to act using real-time dynamic programming.

**RTDP**

Asynchronous VI with heuristic guidance. Updated states at step $n+1 = \text{states visited by the one-step lookahead greedy policy w.r.t } V_n$.

Is there an equivalent for policy iteration? We introduce:

**RTPI**

At iteration $n+1$, updated states are states visited by the one-step lookahead greedy policy w.r.t $V^{\pi_n}$. ie. states visited by the application of $\pi_{n+1}$.
Practical motivation for RTPI

Motivation: don’t want / can’t improve the policy everywhere

- too time/resource consuming
- not useful with regard to ‘relevant’ information gathered

Useful ? Interesting ? Relevant ?

→ “Improving the policy in the situations I am likely to encounter today”

In other words . . .

Which subset of states for Asynchronous PI ?
The ones visited by policy simulation.
Practical motivation for RTPI

Motivation: don’t want / can’t improve the policy everywhere

- too time/resource consuming
- not useful with regard to ‘relevant’ information gathered

Useful ? Interesting ? Relevant ?

→ “Improving the policy in the situations I am likely to encounter today”

In other words . . .

Which subset of states for Asynchronous PI ?

The ones visited by policy simulation.
Practical motivation for RTPI

Motivation: don’t want / can’t improve the policy everywhere

- too time/resource consuming
- not useful with regard to ‘relevant’ information gathered

Useful ? Interesting ? Relevant ?

→ “Improving the policy in the situations I am likely to encounter today”

In other words . . .

Which subset of states for Asynchronous PI ?

The ones visited by policy simulation.
Practical motivation for RTPI

Motivation: don’t want / can’t improve the policy everywhere

- too time/resource consuming
- not useful with regard to ‘relevant’ information gathered

Useful ? Interesting ? Relevant ?

→ “Improving the policy in the situations I am likely to encounter today”

In other words . . .

Which subset of states for Asynchronous PI?

The ones visited by policy simulation.
Plan

1. Time and MDP: motivation and modeling
   - Examples
   - Problem features
   - GSMDP

2. Focusing Policy search in Policy Iteration
   - Policy Iteration algorithms
   - Asynchronous Dynamic Programming
   - Real-Time Policy Iteration

3. Dealing with large dimension, continuous state spaces
   - RL, Monte-Carlo sampling and Statistical Learning
   - The ATPI algorithm (naive version)
   - The ATPI algorithm (complete version)
Simulation-based policy evaluation

Our hypothesis: we have a generative model of the process.
\[ \rightarrow \text{(Monte-Carlo) simulation-based policy evaluation.} \]

Statistical learning

Simulating the policy
\[ \Leftrightarrow \text{Drawing a set of } \textit{trajectories} \]
\[ \Leftrightarrow \text{Finite set of realisations of r.v. } R^\pi(s) \]

We need to
1. abstract (generalize) local information from samples
2. compactly store previous knowledge of \[ V^\pi(s) = E(R^\pi(s)) \]
Simulation-based policy evaluation

Our hypothesis: we have a generative model of the process.

\[ \rightarrow \text{(Monte-Carlo) simulation-based policy evaluation.} \]

Statistical learning

Simulating the policy

\[ \Leftrightarrow \text{Drawing a set of trajectories} \]
\[ \Leftrightarrow \text{Finite set of realisations of r.v. } R^{\pi}(s) \]

We need to

- abstract (generalize) local information from samples
- compactly store previous knowledge of \( V^{\pi}(s) = E(R^{\pi}(s)) \)
Regression for RL

Reminder:

Approximate evaluation: \( V^{\pi_n} \)

One-step improvement: \( \pi_{n+1} \)

(nearest neighbours, SVR, kLASSO, LWPR)
**ATPI**

RTPI algorithm on continuous variables with simulation-based policy evaluation + regression.

**main:**
Input: \( \pi_0 \) or \( \tilde{V}_0, s_0 \)

**loop**

\[ \text{TrainingSet} \leftarrow \emptyset \]

\[ \text{for } i = 1 \text{ to } N_{\text{sim}} \text{ do} \]

\[ \{(s, v)\} \leftarrow \text{simulate}(\tilde{V}, s_0) \]

\[ \text{TrainingSet} \leftarrow \text{TrainingSet} \cup \{(s, v)\} \]

end for

\[ \tilde{V} \leftarrow \text{TrainApproximator}(\text{TrainingSet}) \]

end loop
ATPI

RTPI algorithm on continuous variables with simulation-based policy evaluation + regression.

\[
\text{simulate} (\tilde{V}, s_0):
\]

\[
\text{ExecutionPath} \leftarrow \emptyset
\]

\[
s \leftarrow s_0
\]

\[
\text{while} \quad \text{horizon not reached do}
\]

\[
\text{action} \leftarrow \text{ComputePolicy}(s, \tilde{V})
\]

\[
(s', r) \leftarrow \text{GSMDPstep}(s, \text{action})
\]

\[
\text{ExecutionPath} \leftarrow \text{ExecutionPath} \cup (s', r)
\]

\[
\text{end while}
\]

convert execution path to \{(s, v)\}

\[
\text{return} \{(s, v)\}
\]
ATPI
RTPI algorithm on continuous variables with simulation-based policy evaluation + regression.

\begin{align*}
\text{ComputePolicy}(s, \tilde{V}):
\quad & \text{for } a \in A \text{ do} \\
\quad & \quad \tilde{Q}(s, a) = 0 \\
\quad & \quad \text{for } j = 1 \text{ to } N_{\text{samples}} \text{ do} \\
\quad & \quad \quad (s', r) \leftarrow \text{GSMDPstep}(s, a) \\
\quad & \quad \quad \tilde{Q}(s, a) \leftarrow \tilde{Q}(s, a) + r + \gamma^{t' - t} \tilde{V}(s') \\
\quad & \quad \text{end for} \\
\quad & \quad \tilde{Q}(s, a) \leftarrow \frac{1}{N_{\text{samples}}} \tilde{Q}(s, a) \\
\quad & \text{end for} \\
\quad & \text{action} \leftarrow \arg \max_{a \in A} \tilde{Q}(s, a) \\
\quad & \text{return action}
\end{align*}
Subway problem results

Initial version of online-ATPI with SVR.
Initial policy sets trains to run all day long.

![Graph showing initial state value versus iteration number for ATPI and SVR methods.](image-url)
The ATPI algorithm (naive version)

Is there anybody out there?
Is there anybody out there?

\[ Q(s_0, a_1) =? \]

\[ P(s', t'|s_0, t_0, a_1) \]
Is there anybody out there?

Should I trust my regression? → what if it overestimates the true $V^\pi(s)$?
The ATPI algorithm (naive version)

Is there anybody out there?

Define a notion of *confidence*
Introducing confidence

- “confidence” ⇔ having enough points around $s$
  ⇔ approaching the sufficient statistics for $V^\pi(s)$
  → approx. measure: pdf of the underlying process.

- What should we do if we are not confident?
  → generate data – increase the samples’ density – simulate

- Storing the policy?

Same problem for policy storage than for value function:

*Lagoudakis et al., 03* RL as Classification.

Full statistical learning problem:

(local incremental) regression ($V^\pi$), classification ($\pi$),
density estimation (conf)
Introducing confidence

- “confidence” ⇔ having enough points around $s$
  ⇔ approaching the sufficient statistics for $V^\pi(s)$
  → approx. measure: pdf of the underlying process.
- What should we do if we are not confident?
  → generate data – increase the samples’ density – simulate

Storing the policy?

Same problem for policy storage than for value function:

(Lagoudakis et al., 03) RL as Classification.

Full statistical learning problem:

(local incremental) regression ($V^\pi$), classification ($\pi$),
density estimation ($conf$)
Introducing confidence

- “confidence” ⇔ having enough points around $s$
  ⇔ approaching the sufficient statistics for $V_\pi(s)$
    → approx. measure: pdf of the underlying process.
- What should we do if we are not confident ?
  → generate data – increase the samples’ density – simulate
- Storing the policy ?

Same problem for policy storage than for value function:
(Lagoudakis et al., 03) RL as Classification.

Full statistical learning problem:
(local incremental) regression ($V_\pi$), classification ($\pi$),
density estimation (conf)
Introducing confidence

- “confidence” ⇔ having enough points around \( s \)
  ⇔ approaching the sufficient statistics for \( V^\pi(s) \)
  → approx. measure: pdf of the underlying process.
- What should we do if we are not confident?
  → generate data – increase the samples’ density – simulate
- Storing the policy?

Same problem for policy storage than for value function:

(\cite{Lagoudakis et al., 03}) RL as Classification.

Full statistical learning problem:

(local incremental) regression (\( V^\pi \)), classification (\( \pi \)),
density estimation (conf)
The ATPI algorithm (complete version)

**ATPI - complete version**

```plaintext
ATPI
samples ← ∅
for i = 1 to N_{sim} do
    while t < horizon do
        estimate Q-values
        s' ← apply best action
        store (s, a, r, s') in samples
    end while
end for
train\tilde{V}^{\pi}(samples)
train\tilde{\pi}(samples)
```

Emmanuel Rachelson  Patrick Fabiani  Frédérick Garcia

Simulation-based Approximate Policy Iteration for Generalized Semi-Markov Decision Processes
ATPI - complete version

estimate $Q(s,a)$

$\tilde{Q}(s,a) \leftarrow 0$

for $i = 1$ to $N_a$ do

$(r, s') \leftarrow$ pick next state

if $\text{confidence}(s') = \text{true}$ then

$\tilde{Q}(s,a) \leftarrow \tilde{Q}(s,a) + \frac{r + \tilde{V}_\pi(s')}{N_a}$

else

$\text{data} = \text{simulate}(\pi, s')$

retrain $\tilde{V}_\pi(\text{data})$

$\tilde{Q}(s,a) \leftarrow \tilde{Q}(s,a) + \frac{r + \tilde{V}_\pi(s')}{N_a}$

end if

end for

return $\tilde{Q}(s,a)$
Conclusion

**GSMDP**  Modeling of large scale temporal problems of decision under uncertainty.

**RTPI**  Introduction of a new asynchronous PI method performing partial and incremental state space exploration guided by simulation / local policy improvement.

**ATPI**  Design of a RTPI algorithm for continuous, high dimensional state spaces, exploiting the properties of the time variable and bringing together results from:
- discrete events simulation
- simulation-based policy evaluation
- approximate asynchronous policy iteration
- statistical learning

GiSMoP C++ library
→ http://emmanuel.rachelson.free.fr/fr/gismop.html
Future work

RTPI  Independent algorithm
   • study convergence
   • compare with RTDP

ATPI  Algorithm improvement and testing
   • Even non-parametric methods need some tuning! (currently: LWPR / MC-SVM / OC-SVM)
   • error bounds for API
   • other benchmarks
Thank you for your attention!