Regret Minimization in MDPs with Options


Ronan Fruit[†]


Matteo Pirotta[†]


Alessandro Lazaric[†*]


Emma Brunskill[‡]

[†] SequeL – INRIA Lille
[*] FAIR – Facebook Paris
[‡] Stanford University

Stanford, July 2017

Why Options?

Why Options?

- Most real-world RL tasks involve solving many **different subtasks**

Why Options?

- Most real-world RL tasks involve solving many **different subtasks**
- As the size of the state-action space grows, it becomes difficult to learn **complex behaviours** with "flat" RL methods

Why Options?

- Most real-world RL tasks involve solving many **different subtasks**
- As the size of the state-action space grows, it becomes difficult to learn **complex behaviours** with "flat" RL methods
- **Hierarchical RL** approaches decompose large problems into smaller ones by exploiting the structure of the problem

Why Options?

- Most real-world RL tasks involve solving many **different subtasks**
- As the size of the state-action space grows, it becomes difficult to learn **complex behaviours** with “flat” RL methods
- **Hierarchical RL** approaches decompose large problems into smaller ones by exploiting the structure of the problem
- **Options** are a possible implementation of temporally extended actions (skills) for hierarchical RL

Why Options?

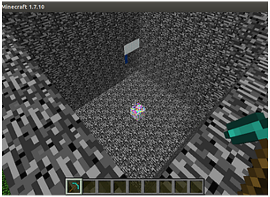
- Most real-world RL tasks involve solving many **different subtasks**
- As the size of the state-action space grows, it becomes difficult to learn **complex behaviours** with “flat” RL methods
- **Hierarchical RL** approaches decompose large problems into smaller ones by exploiting the structure of the problem
- **Options** are a possible implementation of temporally extended actions (skills) for hierarchical RL

In this talk: **on-line learning** with options

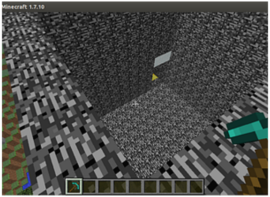
Example: Minecraft [?]

- Three subtasks

Navigate



Pick-up

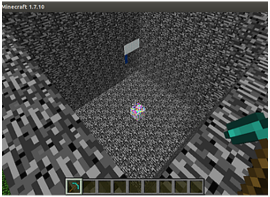


Place

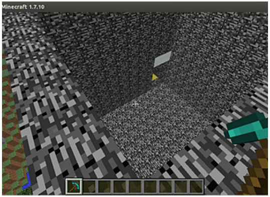
Example: Minecraft [?]

- Three subtasks

Navigate

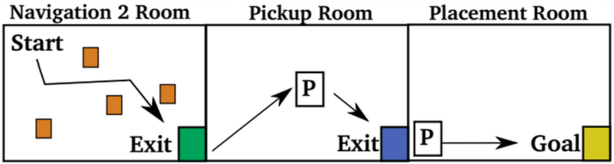


Pick-up



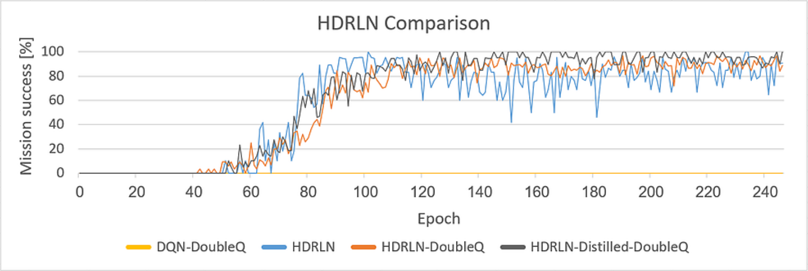
Place

- One macro-task combining all three subtasks



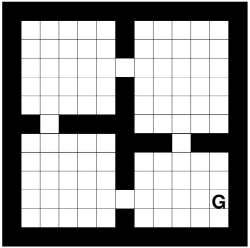
Example: Minecraft [?]

- Simulations



Options Limitations

Four-rooms maze [?]



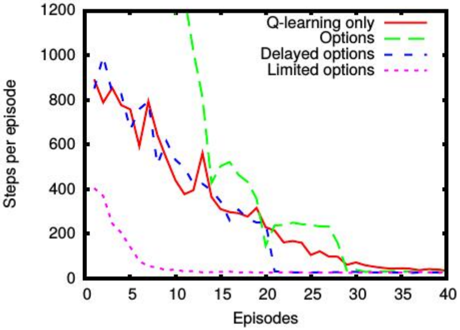


Figure 4: Comparison of learning agents with varying access to correct temporal abstractions. The Q-learning agent never uses options. The "options" agent gains immediate access to the correct options everywhere. The "delayed options" agent gains access to these options after 20 episodes. The "limited options" agent gains immediate access to these options except in the lower-right room. Each learning curve is the average of 50 independent runs.

Research questions

Empirical observations: introducing options in an MDP can *speed up* learning [**?**] but can also be *harmful* [**?**].

$\rightsquigarrow$ Is there a **theoretical explanation** for this?

Research questions

Empirical observations: introducing options in an MDP can *speed up* learning [**?**] but can also be *harmful* [**?**].

$\rightsquigarrow$ Is there a **theoretical explanation** for this?

Option Design: a challenging problem that has been always empirically tackled:

- Leveraging on MDP properties (e.g., bottleneck discovery [**?**], Laplacian analysis [**?**])
- Direct option optimization (e.g., Option-Critic [**?**])
- and many other concepts

$\rightsquigarrow$ Can we exploit theoretical *properties to design options*?

Research questions

Empirical observations: introducing options in an MDP can *speed up* learning [**?**] but can also be *harmful* [**?**].

$\rightsquigarrow$ Is there a **theoretical explanation** for this?

Option Design: a challenging problem that has been always empirically tackled:

- Leveraging on MDP properties (e.g., bottleneck discovery [**?**], Laplacian analysis [**?**])
- Direct option optimization (e.g., Option-Critic [**?**])
- and many other concepts

$\rightsquigarrow$ Can we exploit theoretical *properties to design options*?

Disclaimer:

this talk is not about option design

options are **assumed** to be given as input

Problem addressed in this talk

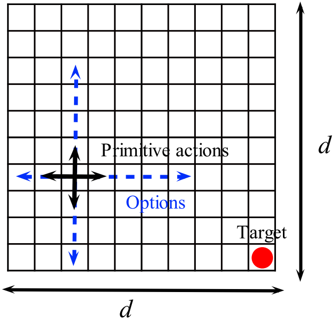
- Introducing options enables to **reduce the size of the state-action space** hence speed up learning

Problem addressed in this talk

- Introducing options enables to **reduce the size of the state-action space** hence speed up learning
- Is there another advantage in using options?

Problem addressed in this talk

- Introducing options enables to **reduce the size of the state-action space** hence speed up learning
- Is there another advantage in using options?



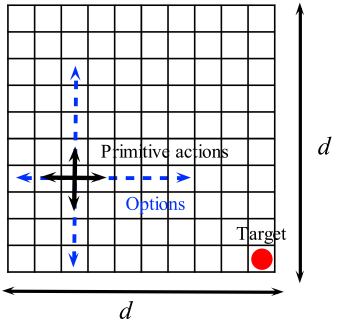
GRID WORLD

Replace all 4 cardinal actions by *cardinal options*:

- Same number of states
- Same number of actions

Problem addressed in this talk

- Introducing options enables to **reduce the size of the state-action space** hence speed up learning
- Is there another advantage in using options?



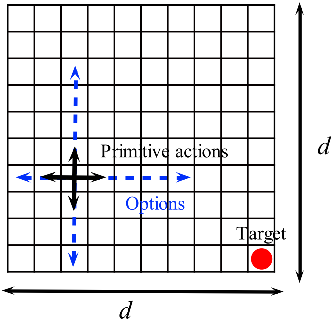
GRID WORLD

Replace all 4 cardinal actions by *cardinal options*:

- Same number of states
- Same number of actions

Problem addressed in this talk

- Introducing options enables to **reduce the size of the state-action space** hence speed up learning
- Is there another advantage in using options?



<u>Grid world</u>

Replace all 4 cardinal actions by *cardinal options*:

- Same number of states
- Same number of actions

Question: What is the impact of options in the above example? How is exploration affected by options?

Markov Decision Processes

$$\mathcal{M} = \{\mathcal{S}, \mathcal{A}, p, r\}$$

- $\mathcal{S}$ is the state space
- $\mathcal{A} = (\mathcal{A}_s)_{s \in \mathcal{S}}$ is the set of actions
- when choosing action a in state s:
  - random reward with mean $r(s, a) \in [0, 1]$
  - transition to the next state $(s, a) \to s'$ according to transition probability distribution $p(\cdot|s, a)$

[?]

Average Reward MDP

Policies, gain and optimality

The average expected reward (or **gain**) of a policy π is

$$g(\mathcal{M},\pi) = \lim_{N\to\infty} \frac{1}{N}\mathbb{E}\left[\sum_{t=1}^{N} r(s_t,a_t)|\mathcal{M},\pi\right]$$

where $a_t \sim \pi(\cdot|s_t)$.

Average Reward MDP

Policies, gain and optimality

The average expected reward (or **gain**) of a policy π is

$$g(\mathcal{M}, \pi) = \lim_{N \to \infty} \frac{1}{N} \mathbb{E}\left[\sum_{t=1}^{N} r(s_t, a_t) | \mathcal{M}, \pi\right]$$

where $a_t \sim \pi(\cdot | s_t)$.

Learner's Goals:

1. Find the *optimal* policy $\pi^* = \arg\max_{\pi} g(\mathcal{M}, \pi)$

OPTIMALITY EQUATION

$$g^* = \max_{a} \left\{ r(s,a) + p^T u^* - u^*(s) \right\}$$

Average Reward MDP

Policies, gain and optimality

The average expected reward (or **gain**) of a policy π is

$$g(\mathcal{M}, \pi) = \lim_{N \to \infty} \frac{1}{N} \mathbb{E}\left[\sum_{t=1}^{N} r(s_t, a_t) | \mathcal{M}, \pi\right]$$

where $a_t \sim \pi(\cdot | s_t)$.

Learner's Goals:

1. Find the *optimal* policy $\pi^* = \arg\max_{\pi} g(\mathcal{M}, \pi)$

   Optimality Equation

   $$g^* = \max_a \left\{ r(s, a) + p^T u^* - u^*(s) \right\}$$

2. Do this *online*! Don't loose too much w.r.t. $g^* := g(\mathcal{M}, \pi^*)$

Average Reward MDP

Policies, gain and optimality

The average expected reward (or **gain**) of a policy π is

$$g(\mathcal{M},\pi) = \lim_{N\to\infty} \frac{1}{N}\mathbb{E}\left[\sum_{t=1}^{N} r(s_t,a_t)|\mathcal{M},\pi\right]$$

where $a_t \sim \pi(\cdot|s_t)$.

Learner's Goals:

1. Find the *optimal* policy $\pi^* = \arg\max_\pi g(\mathcal{M},\pi)$

   OPTIMALITY EQUATION

   $$g^* = \max_a \left\{r(s,a) + p^T u^* - u^*(s)\right\}$$

2. Do this *online*! Don't loose too much w.r.t. $g^* := g(\mathcal{M},\pi^*)$

$\rightsquigarrow$ **Regret minimization!**

FREQUENTIST REGRET: *optimism in face of uncertainty (OFU)*

$$\Delta(\mathcal{M},\mathfrak{A},T) = Tg^*(\mathcal{M}) - \sum_{t=1}^{T} r_t$$

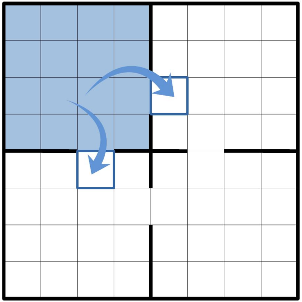
Temporal Abstraction

The Option Framework

Definition 1

A (Markov) Option o is a 3-tuple $\{s_o, \beta_o, \pi_o\}$ where:

- $s_o \in \mathcal{S}$ is the **states** where the option can be initiated,
- $\beta_o : \mathcal{S} \to [0, 1]$ is a **Markov termination condition**,
- $\pi_o \in \Pi_M^{SR}$ is a **stationary Markov policy**.



Temporal Abstraction

Semi-Markov Decision Processes

SMDP [?]

A set of options $\mathcal{O}$ defined on an MDP M induces an SMDP M' :

$$M + \mathcal{O} \implies M'$$

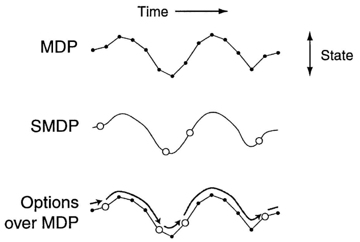
Temporal Abstraction

Semi-Markov Decision Processes

SMDP [?]

A set of options $\mathcal{O}$ defined on an MDP M induces an SMDP M' :

$$M + \mathcal{O} \implies M'$$



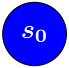
A Semi-Markov Decision Process

$$M' = \{\mathcal{S}', \mathcal{A}', p, r, \tau\}$$

is an MDP with a **random holding time** $\tau(s,a)$ associated with any state-action pair.

Learning in SMDP

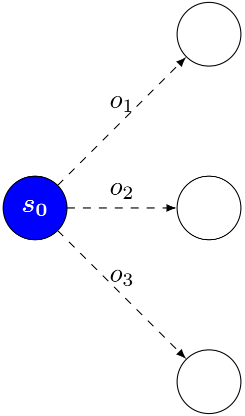
T_n: number of time steps
n: number of decision steps



$n = 0 \qquad T_0 = 0$

Learning in SMDP

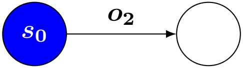
T_n: number of time steps
n: number of decision steps



$n = 1 \qquad T_0 = 0$

Learning in SMDP

T_n: number of time steps
n: number of decision steps



$n = 1 \qquad T_0 = 0$

Learning in SMDP

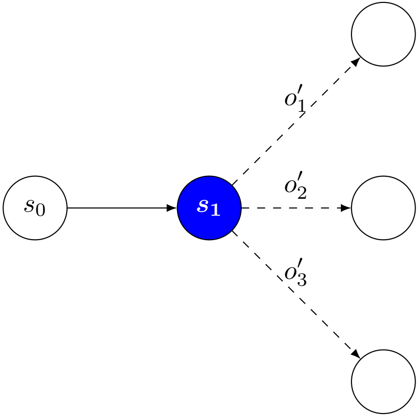
T_n: number of time steps
n: number of decision steps

$$s_0 \xrightarrow{\boldsymbol{\tau_1},\ \boldsymbol{R_1}} \boldsymbol{s_1}$$

$$n = 1 \qquad T_1 \leftarrow T_0 + \tau_1 \qquad R_1 \leftarrow \sum_{t=1}^{\tau_1} r_t$$

Learning in SMDP

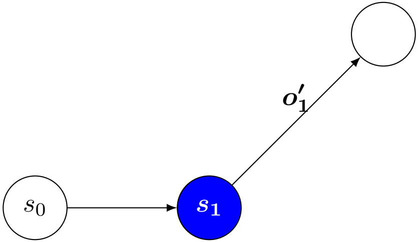
T_n: number of time steps
n: number of decision steps



$n = 2 \qquad T_1 = \tau_1$

Learning in SMDP

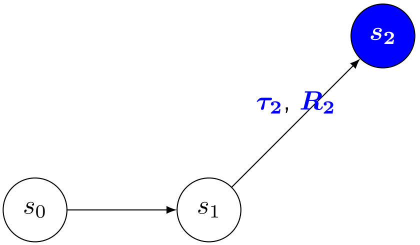
T_n: number of time steps
n: number of decision steps



$n = 2 \qquad T_1 = \tau_1$

Learning in SMDP

T_n: number of time steps
n: number of decision steps



$$n = 2 \qquad T_2 \leftarrow T_1 + \tau_2 \qquad R_2 \leftarrow \sum_{t=1}^{\tau_2} r_t$$

$$\implies T_n = \sum_{i=1}^{n} \tau_i$$

Optimal Policy and Regret

Optimal policy

$$g^*_{\mathcal{O}} = \max_{\pi} g^{\pi}_{\mathcal{O}} = \max_{\pi} \lim_{n\to\infty} \mathbb{E}^{\pi}\left[\frac{\sum_{t=1}^{n} R_t}{T_n}\right]$$

$$\pi^* : \mathcal{S} \to \mathcal{O}$$

Frequentist (SMDP) regret

$$\Delta(\mathcal{M}, \mathfrak{A}, T_n) = T_n g^*_{\mathcal{O}} - \sum_{i=1}^{n} R_i$$

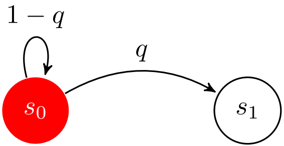
SMDP-UCRL

1. Construct set of *plausible SMDPs* $\mathcal{M}_k = \Phi(\mathcal{H}_k)$
   By exploiting confidence interval on
   - Option transition probability: $\beta_k^p(s, o)$
   - Option reward: $\beta_k^r(s, o)$
   - Option duration: $\beta_k^\tau(s, o)$

SMDP-UCRL

1. Construct set of *plausible SMDPs* $\mathcal{M}_k = \Phi(\mathcal{H}_k)$
   By exploiting confidence interval on
   - Option transition probability: $\beta_k^p(s, o)$
   - Option reward: $\beta_k^r(s, o)$
   - Option duration: $\beta_k^\tau(s, o)$ More complex than for MDPs!

EXAMPLE



$$\mathbb{E}[\tau] = \frac{1}{q}$$

$$P(\tau = k) = (1-q)^{k-1} q$$

What about a confidence interval on τ?

SMDP-UCRL

1. Construct set of *plausible SMDPs* $\mathcal{M}_k = \Phi(\mathcal{H}_k)$
   By exploiting confidence interval on
   - Option transition probability: $\beta_k^p(s, o)$
   - Option reward: $\beta_k^r(s, o)$
   - Option duration: $\beta_k^\tau(s, o)$ More complex than for MDPs!

[?, Lemma 3]

If the set of options is proper, all **holding times and rewards are sub-Exponential**. Moreover, they are sub-Gaussian if and only if they are bounded.

This property comes from its inner Markov structure
(to be continued)

SMDP-UCRL

1. Construct set of *plausible SMDPs* $\mathcal{M}_k = \Phi(\mathcal{H}_k)$
   By exploiting confidence interval on
   - Option transition probability: $\beta_k^p(s, o)$
   - Option reward: $\beta_k^r(s, o)$
   - Option duration: $\beta_k^\tau(s, o)$

$\rightsquigarrow$ concentration inequalities to sub-exponential r.v.!

[?, Lemma 3]

If the set of options is proper, all **holding times and rewards are sub-Exponential**. Moreover, they are sub-Gaussian if and only if they are bounded.

This property comes from its inner Markov structure
(to be continued)

SMDP-UCRL

1. Construct set of *plausible SMDPs* $\mathcal{M}_k = \Phi(\mathcal{H}_k)$
   By exploiting confidence interval on
   - Option transition probability: $\beta_k^p(s, o)$
   - Option reward: $\beta_k^r(s, o)$
   - Option duration: $\beta_k^\tau(s, o)$

   $\rightsquigarrow$ concentration inequalities to sub-exponential r.v.!

2. Compute $\pi_k \in \arg\max_{\pi \in \Pi_{\mathcal{O}}, M \in \mathcal{M}_k} g_{\mathcal{O}}(M, \pi)$
   - Use EVI to solve the optimality equation

$$\tilde{g}_{\mathcal{O}}^* = \max_{o \in \mathcal{O}_s} \left\{ \max_{R \in \widetilde{\boldsymbol{R}}, \tau \in \widetilde{\boldsymbol{\tau}}} \left\{ \frac{R(s,o)}{\tau(s,o)} + \frac{1}{\tau(s,o)} \left(\max_{p \in \widetilde{\boldsymbol{p}}(s,o)} \left\{ p^T u^* \right\} - u^*(s) \right) \right\} \right\}$$

Regret for MDPs

Theorem 2

In a finite MDP with diameter D, with probability at least $1-\delta$ the regret of UCRL after T_n time steps is bounded by

$$\Delta(\mathcal{M}, \mathfrak{A}, T_n) = O\left(DS\sqrt{AT_n \log\left(\frac{T_n}{\delta}\right)}\right)$$

Regret for MDPs

Theorem 2

In a finite MDP with diameter D, with probability at least $1-\delta$ the regret of UCRL after T_n time steps is bounded by

$$\Delta(\mathcal{M},\mathfrak{A},T_n) = O\left(\boxed{D} S \sqrt{AT_n \log\left(\frac{T_n}{\delta}\right)}\right)$$

Diameter

The *diameter* of an MDP $\mathcal{M}$ is the maximal expected time it takes to reach any state from any other state under an appropriate policy

$$D(\mathcal{M}) := \max_{s,s' \in \mathcal{S}, s \neq s'} \min_{\pi} \boxed{\mathbb{E}^{\pi}\left[T(s')|s_0 = s\right]}$$

Mean first passage time

Def. *Communicating MDP* $\Leftrightarrow$ *Finite Diameter*

Regret analysis for SMDPs

[?, Theorem 1]

High-probability regret bound for SMDP-UCRL in M':

$$\Delta(M', \mathfrak{A}, T_n) = O\left(\left(D'\sqrt{S'} + \boldsymbol{\mathcal{C}(M', n, \delta)}\right)\sqrt{S'A'n\log\left(\frac{n}{\delta}\right)}\right)$$

where $\mathcal{C}(M', n, \delta)$ is

h.p. bound on holding time

$$\mathcal{C}(M', n, \delta) = \tau_{\max} + C_\tau\sqrt{\log\left(\frac{n}{\delta}\right)}$$

Regret analysis for SMDPs

[?, Theorem 1]

High-probability regret bound for SMDP-UCRL in M':

$$\Delta(M', \mathfrak{A}, T_n) = O\left(\left(D'\sqrt{S'} + \boldsymbol{\mathcal{C}(M', n, \delta)}\right)\sqrt{S'A'n\log\left(\frac{n}{\delta}\right)}\right)$$

where $\mathcal{C}(M', n, \delta)$ is

h.p. bound on holding time

$$\mathcal{C}(M', n, \delta) = \tau_{\max} + C_\tau\sqrt{\log\left(\frac{n}{\delta}\right)}$$

Comparing regrets SMDP/MDP (ratio):

$$\mathcal{R} \sim \frac{D'}{D}\sqrt{\frac{On}{AT_n}}$$

Comments on SUCRL

- SUCRL requires prior knowledge about options (sub-Exponential parameters)
- this requirement can be removed by better exploiting option properties

Comments on SUCRL

- SUCRL requires prior knowledge about options (sub-Exponential parameters)
- this requirement can be removed by better exploiting option properties

$\rightsquigarrow$ **Parameter-free SUCRL**

Comments on SUCRL

- SUCRL requires prior knowledge about options (sub-Exponential parameters)
- this requirement can be removed by better exploiting option properties

$\rightsquigarrow$ **Parameter-free SUCRL**

- Avoid considering options as atomic operations

Comments on SUCRL

- SUCRL requires prior knowledge about options (sub-Exponential parameters)
- this requirement can be removed by better exploiting option properties

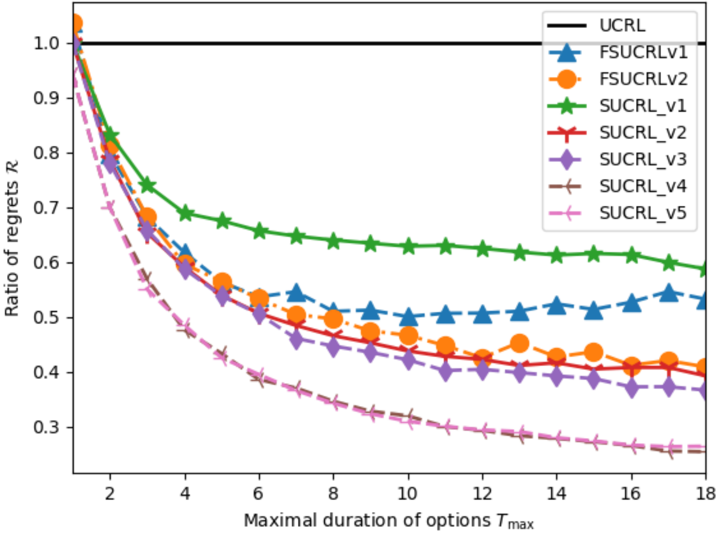
$\rightsquigarrow$ **Parameter-free SUCRL**

- Avoid considering options as atomic operations
- Take into account the inner option MDP structure

Experiments

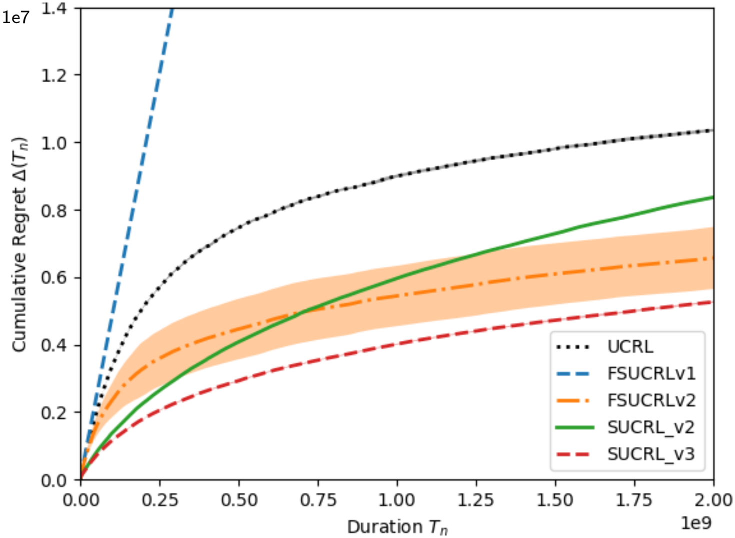
Grid World

Domain presented in the introduction



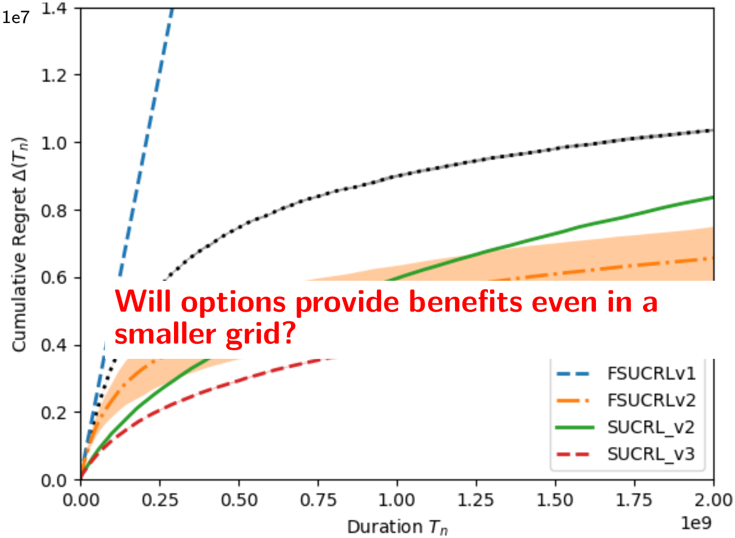
Four Rooms 14x14

The classical domain for options



Four Rooms 14x14

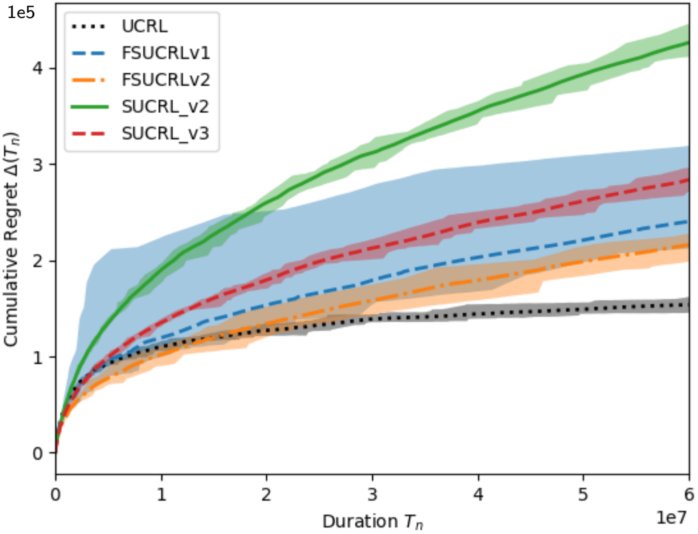
The classical domain for options



Will options provide benefits even in a smaller grid?

Four Rooms 6x6

The classical domain for options



Conclusions

- Temporal abstraction is **powerful**
  - Faster learning
  - Less regret
- But it does not come for free
  - May increase the computational complexity
  - Requires a far-sighted design of options

Thank you for your attention

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI*, pages 1726–1734, 2017.

Ronan Fruit and Alessandro Lazaric. Exploration-exploitation in mdps with options. In *AISTATS*, volume 54 of *JMLR Workshop and Conference Proceedings*, pages 576–584. JMLR.org, 2017.

Nicholas K. Jong, Todd Hester, and Peter Stone. The utility of temporal abstraction in reinforcement learning. In *The Seventh International Joint Conference on Autonomous Agents and Multiagent Systems*, May 2008.

Marlos C. Machado, Marc G. Bellemare, and Michael H. Bowling. A laplacian framework for option discovery in reinforcement learning. *CoRR*, abs/1703.00956, 2017.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994. ISBN 0471619779.

Martin Stolle and Doina Precup. Learning options in reinforcement learning. In *International Symposium on Abstraction, Reformulation, and Approximation*, pages 212–223. Springer, 2002.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181 – 211, 1999.

Chen Tessler, Shahar Givony, Tom Zahavy, Daniel J. Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in minecraft. *CoRR*, abs/1604.07255, 2016.

Chen Tessler, Shahar Givony, Tom Zahavy, Daniel J. Mankowitz, and Shie Mannor. A deep hierarchical approach to lifelong learning in minecraft. In *AAAI*, pages 1553–1561. AAAI Press, 2017.