Risk and Robustness in Reinforcement Learning: Nothing ventured nothing gained

Shie Mannor

Technion - Israel Institute of Technology & Ford

November 8th, 2019

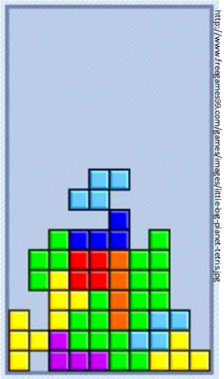
Introduction

Reinforcement learning (RL): all about sequential decision making

Reinforcement learning (RL): all about sequential decision making

Some examples:

1. Tetris:



Reinforcement learning (RL): all about sequential decision making

Some examples:

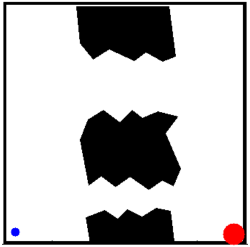
2. American put option:

A contract, giving its owner the right to sell a stock at **strike price** K, at any time until the **maturity time** T

Reinforcement learning (RL): all about sequential decision making

Some examples:

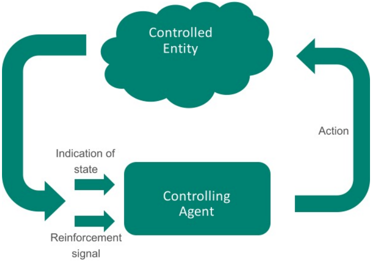
3. Pinball domain:



Stochastic shortest path

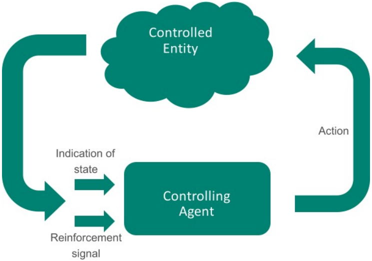
Classical Reinforcement Learning

Model = Markov Decision Process (MDP) = $\{S, P, R, A\}$



Classical Reinforcement Learning

Model = Markov Decision Process (MDP) = $\{S, P, R, A\}$

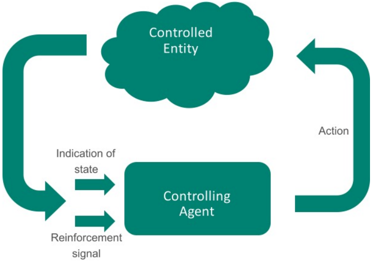


States S, actions A are known and given
Transitions P and rewards R are not known.

Classical objective: $\max_\pi \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r_t\right]$, $\gamma < 1$

Classical Reinforcement Learning

Model = Markov Decision Process (MDP) = $\{S, P, R, A\}$

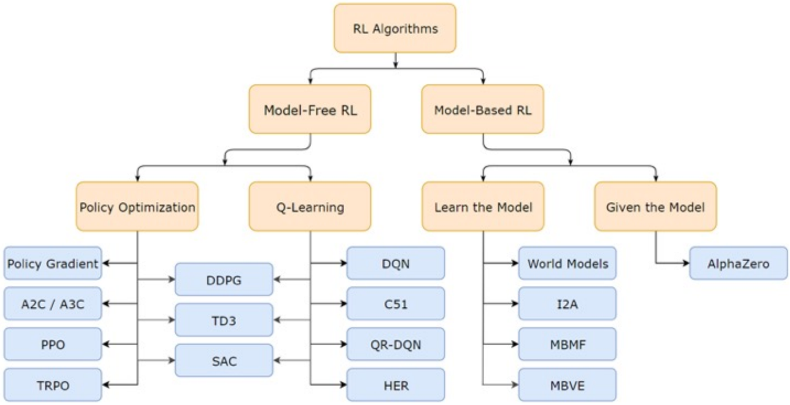


States S, actions A are known and given
Transitions P and rewards R are not known.

Classical objective: $\max_\pi \mathbb{E}\left[\sum_{t=1}^{\infty} \gamma^t r_t\right]$, $\gamma < 1$

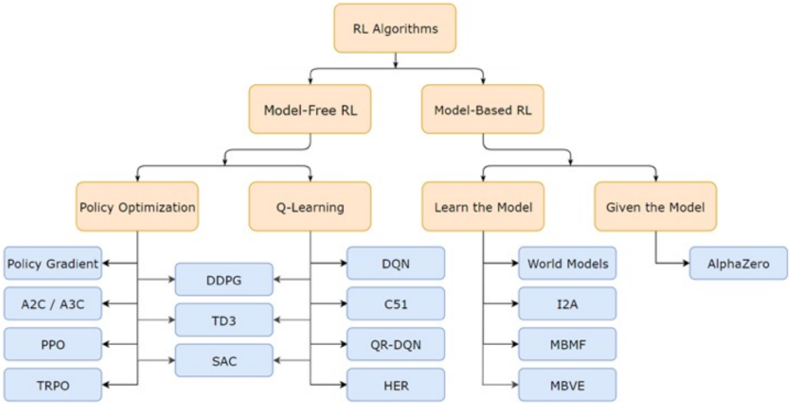
"All models are wrong, but some are useful", G. Box

Last few years



We have many algorithms!

Last few years



We have many algorithms!

But few real-world successes ....

Reductionist Fallacies

Only smart people can play Go/Chess/Shogi very well

(Almost) Everybody can tell a joke

Computers are really good in Go/Chess/Shogi

Reductionist Fallacies

Only smart people can play Go/Chess/Shogi very well

(Almost) Everybody can tell a joke

Computers are really good in Go/Chess/Shogi

$\therefore$ Computer can easily tell a joke

Reductionist Fallacies

Only smart people can play Go/Chess/Shogi very well

(Almost) Everybody can tell a joke

Computers are really good in Go/Chess/Shogi

$\therefore$ Computer can easily tell a joke

A(G)I: Do anything a human can do

Terminator

Agent Smith

Commander Data

3 Types of RL problems

Static

Dynamic

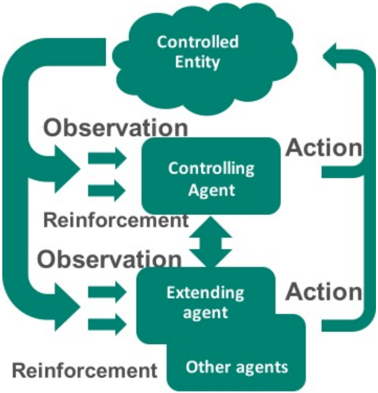
Counterfactual

An RL Problem

- Reward: $, time, energy
- State/observation: what do I see/ know/ measure
- Actions: What can I do

Extended Intelligence

Design of systems that participate as responsible, aware and robust elements of more complex systems.



The five principles of EI

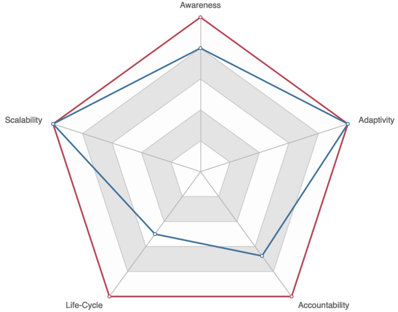
- **Awareness**: Know how well it performs, communicate its performance, identify what is happening to it, and be cognizant of other entities (other agents and humans).
- **Accountability**: Explain its actions: reasoning in words, by example, or in any other interpretable means.
- **Adaptivity**: Function properly under a variety of conditions. Some of these conditions may be expected and some harder to predict.
- **Life-Cycle consciousness**: Be aware of the life cycle, including debugging tools such as unit-testing, decomposability, and interaction with other sub-systems within a more complex system.
- **Scalability with resources**: The more resources we have in terms of data and computational power the better the policy.

Research agenda

Unleash the power of RL for EI: Develop the methodology to learn, adapt and optimize in dynamic environments

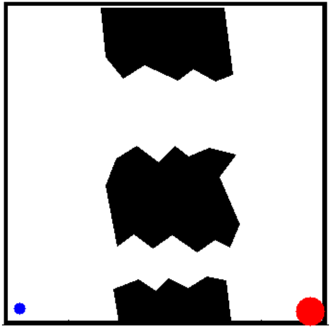
Research agenda

Unleash the power of RL for EI: Develop the methodology to learn, adapt and optimize in dynamic environments

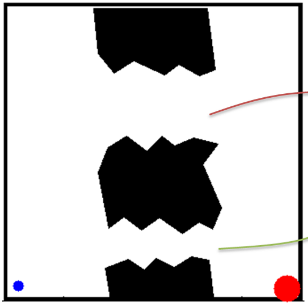


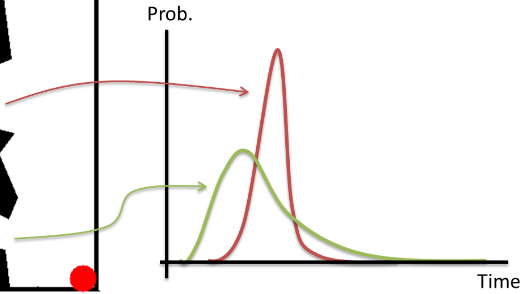
Rest of this talk: Risk sensitivity and Robustness

Why should we be risk-sensitive?

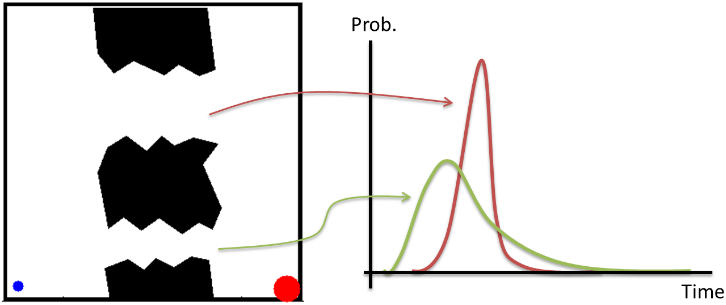


Why should we be risk-sensitive?





Why should we be risk-sensitive?



Risk-awareness → robust policies!

Three Types of Uncertainties

1. Parmeter uncertainty

- Uncertainty in MDP *parameters* (transitions, rewards)
- Objective:

$$\max_{\pi} \min_{P \in \text{possible MDP parameters}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right]$$

- Origins in robust control

Three Types of Uncertainties

2. Inherent uncertainty

- Cumulative reward is stochastic
- Expectation does not capture *variability*
- Example:
  - Policy 1 : $\begin{cases} 1\$, & \text{w.p. } 0.5 \\ -1\$, & \text{w.p. } 0.5 \end{cases}$,
  - Policy 2 : $\begin{cases} 1000\$, & \text{w.p. } 0.5 \\ -1000\$, & \text{w.p. } 0.5 \end{cases}$.

Three Types of Uncertainties

2. Inherent uncertainty

- Cumulative reward is stochastic
- Expectation does not capture *variability*
- Objective:

$$\max_{\pi} \rho \left[\sum_{t=0}^{\infty} \gamma^t r(s_t) \right]$$

- ρ is a *risk measure*, e.g., $\rho(X) = \mathbb{E}[X] - \beta \mathrm{Var}[X]$
- Explicit safety against 'unluckiness'
- Humans tend to be risk aware

Three Types of Uncertainties

3. Model uncertainty

- Model itself not known (observations/features/order)
- Objective:

$$\max_{\pi} \min_{\text{possible models}} \mathbb{E}_{model}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)\right]$$

- Model mismatch handled explicitly
- Origins in multi-model control

Applicability

When is risk-sensitivity important?

- Cost of failure is high
  - Finance
  - Smart-grids
  - Health
  - Robotics (e.g., safety)
- Model is not known (always) and created from a few samples

Applicability

When is risk-sensitivity important?

- Cost of failure is high
  - Finance
  - Smart-grids
  - Health
  - Robotics (e.g., safety)
- Model is not known (always) and created from a few samples

We desire:

- Scalability
- Adaptivity
- Accountability

What about computational complexity?

Most risk related problems are (really) hard, so forget about exact solutions for all but simplest problems.

SM and J. N. Tsitsiklis, EJOR 2013 and other refs

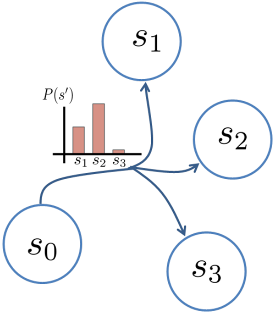
Part 1
Robust MDPs with function approximation

A. Tamar, SM, and H. Xu, ICML 2014
A. Tamar, Y. Chow, M. Ghavamzadeh, and SM, NIPS 2015

Introduction - Planning with *Parameter* Uncertainty

Setting:

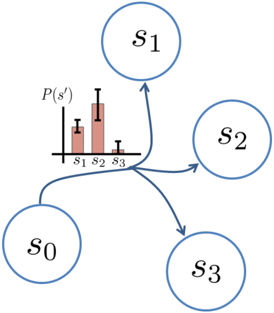
- Planning problem
- Uncertain transitions
  - Confidence intervals
  - Heuristic simulator
  - Time changing dynamics
  - etc.



Introduction - Planning with *Parameter* Uncertainty

Setting:

- Planning problem
- Uncertain transitions
  - Confidence intervals
  - Heuristic simulator
  - Time changing dynamics
  - etc.
- Potentially large impact [SM et. al, Management Science 2010]
  - Uncertainty amplification
  - Disasters / safety
  - Smart grids, finance

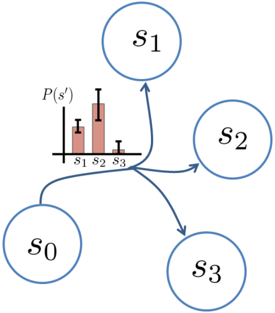


Background: Robust MDPs

Definitions:

- Robust Markov decision processes:
- State, actions and rewards as in the standard model
- Transitions $P(s'|s,a) \in \mathcal{P}$
- Policy π
- Worst-case objective

$$\sup_{\pi} \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t) \right]$$



Background: Robust MDPs

Dynamic programming solution

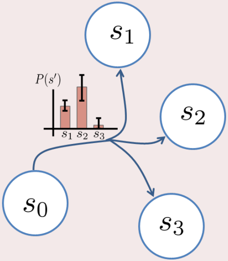
- Robust value function (fixed policy)

$$V^{\pi}(s) \doteq \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s\right]$$

Dynamic programming solution

- Robust value function (fixed policy)

$$V^\pi(s) \doteq \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s\right]$$

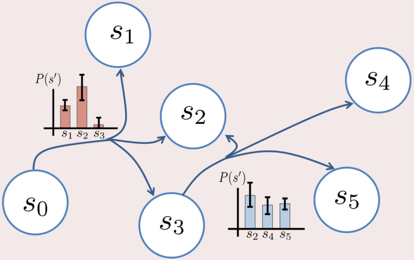


Background: Robust MDPs

Dynamic programming solution

- Robust value function (fixed policy)

$$V^{\pi}(s) \doteq \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s\right]$$



Background: Robust MDPs

Dynamic programming solution

- Robust value function (fixed policy)

$$V^\pi(s) \doteq \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s\right]$$

- Robust Bellman equation (fixed policy)

$$V^\pi(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^P\left[V^\pi(s')|s, \pi(s)\right]$$

Background: Robust MDPs

Dynamic programming solution

- Robust value function (fixed policy)

$$V^{\pi}(s) \doteq \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t) | s_0 = s\right]$$

- Robust Bellman equation (fixed policy)

$$V^{\pi}(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^{P}\left[V^{\pi}(s') | s, \pi(s)\right]$$

- Small problems: solved Policy iteration [Iyengar, 2005] and value iteration approach [Nilim et al. 2005]
- Large problems: Dynamic Programming cannot handle large spaces ("the curse of dimensionality")

Approximate value function

- Given state-dependent features $\phi(s)$
- Linear function approximation

$$\tilde{V}^{\pi}(s) = \phi(s)^{\top} w$$

- How to select w?

- For standard (non-robust) problems:

$$V^{\pi}(s) \doteq \mathbb{E}^{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^{t} r(s_t) | s_0 = s\right]$$

Sample and regress w.

Approximate value function

- Given state-dependent features $\phi(s)$
- Linear function approximation

$$\tilde{V}^{\pi}(s) = \phi(s)^{\top} w$$

- How to select w?

- For robust problems

$$V^{\pi}(s) = \inf_{P \in \mathcal{P}} \mathbb{E}^{\pi, P}\left[\sum_{t=0}^{\infty} \gamma^{t} r(s_t) | s_0 = s\right]$$

Cannot regress w: how to sample trajectories from worst-case model?

Robust Policy Evaluation

Our approach

- Recall the Bellman equation

$$V^{\pi}(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^{P}\left[V^{\pi}(s') | s, \pi(s)\right]$$

- Idea: **bootstrap!**

Our approach

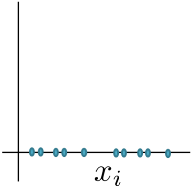
- Recall the Bellman equation

$$V^\pi(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^P \left[V^\pi(s') | s, \pi(s) \right]$$

- Idea: **bootstrap!**

Algorithm

Given: initial weights w_0, sample states $x_1 \ldots x_N$



Robust Policy Evaluation

Our approach

- Recall the Bellman equation

$$V^\pi(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^P \left[V^\pi(s') | s, \pi(s) \right]$$

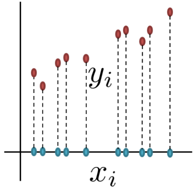
- Idea: **bootstrap!**

Algorithm

Given: initial weights w_0, sample states $x_1 \ldots x_N$

- At iterate $k+1$ generate regression targets

$$y_i = r(x_i) + \gamma \inf_{P \in \mathcal{P}(x_i)} \sum_{x'} P(x'|x_i, \pi(x_i)) \underbrace{\phi(x')^\top w_k}_{\tilde{V}_k^\pi(x')}$$



Robust Policy Evaluation

Our approach

- Recall the Bellman equation

$$V^\pi(s) = r(s) + \gamma \inf_{P \in \mathcal{P}(s)} \mathbb{E}^P \left[V^\pi(s') | s, \pi(s) \right]$$

- Idea: **bootstrap!**

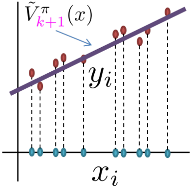
Algorithm

Given: initial weights w_0, sample states $x_1 \ldots x_N$

- At iterate $k+1$ generate regression targets

$$y_i = r(x_i) + \gamma \inf_{P \in \mathcal{P}(x_i)} \sum_{x'} P(x'|x_i, \pi(x_i)) \underbrace{\phi(x')^\top w_k}_{\tilde{V}_k^\pi(x')}$$

- Solve for w_{k+1} using least squares



Results:

Guarantees

- The magic: Convergence + Error bounds

Policy improvement

- Can iterate between policy evaluation and policy improvement
- Can derive deep Q-learning (model free) simulation based algorithm
- Error bounds follow through

American put option

- A contract, giving its owner the right to sell a stock at **strike price** K, at any time until the **maturity time** T

Image source: Yahoo! Finance

- Execution profit: $\max(\text{stock price} - K, 0)$
- Policy: When to execute? Maximize expected profit!

Experiments

American put option

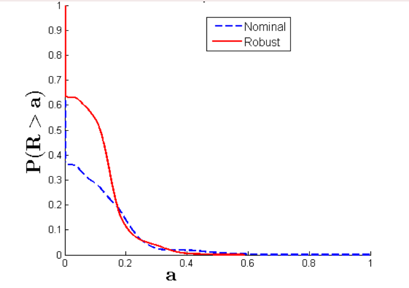
- A contract, giving its owner the right to sell a stock at **strike price** K, at any time until the **maturity time** T

Image source: Yahoo! Finance

- Execution profit: $\max(\text{stock price} - K, 0)$
- Policy: When to execute? Maximize expected profit!
- MDP formulation
- Price transitions – estimated from historical data
- Robust value – **consider estimation uncertainty!**

Model mis-specification

- True model: transitions depend on price (mean reversion)
- Estimated model: constant transitions
- Mismatch in model class, not only in parameters
- In practice we never know the true model class!
- Robust policy $\rightarrow$ robust to model mis-specification



Part 2
Risk Sensitive Policy Gradient: CVaR

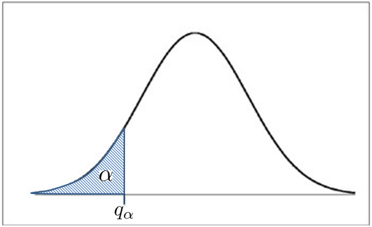
A. Tamar, Y. Glassner and SM, AAAI 2015
Y. Chow, A. Tamar, SM and M. Pavone, NIPS 2015

Conditional Value at Risk (CVaR)

CVaR definition

- X - random variable
- $q_\alpha(X)$ - α quantile
- α-CVaR:

$$\Phi_\alpha(X) = \mathbb{E}\left[X \mid X \leq q_\alpha\right]$$



Estimation

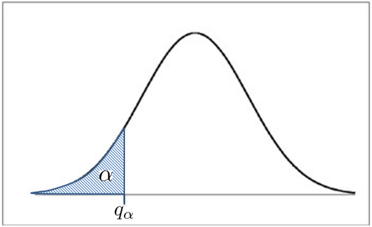
Conditional Value at Risk (CVaR)

CVaR definition

- X - random variable
- $q_\alpha(X)$ - α quantile
- α-CVaR:

$$\Phi_\alpha(X) = \mathbb{E}\left[X \mid X \leq q_\alpha\right]$$

- Expected shortfall
- Sensitive to rare, disastrous events



Estimation

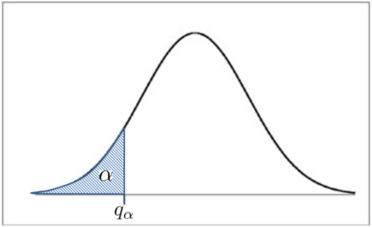
Conditional Value at Risk (CVaR)

CVaR definition

- X - random variable
- $q_\alpha(X)$ - α quantile
- α-CVaR:

$$\Phi_\alpha(X) = \mathbb{E}[X | X \leq q_\alpha]$$

- Expected shortfall
- Sensitive to rare, disastrous events

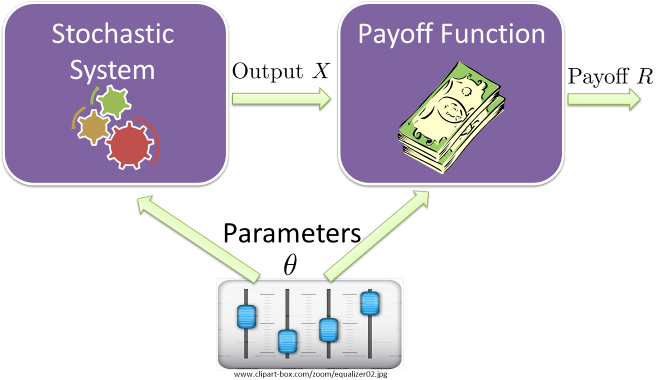


Estimation

$$\mathbb{E}[X] \approx \frac{1}{N} \sum_{1 \leq i \leq N} x_i$$

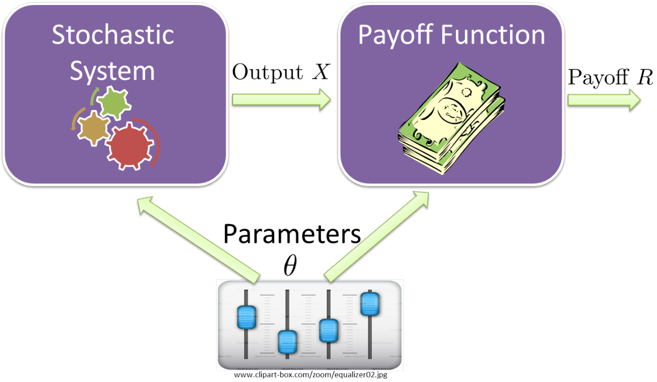
$$\Phi_\alpha[X] \approx \frac{1}{\alpha N} \sum_{\alpha N \text{ worst}} x_i$$

Risk Sensitive Policy Optimization



Policy encoded by parameters: robotics, grids, games (e.g., deep network params)

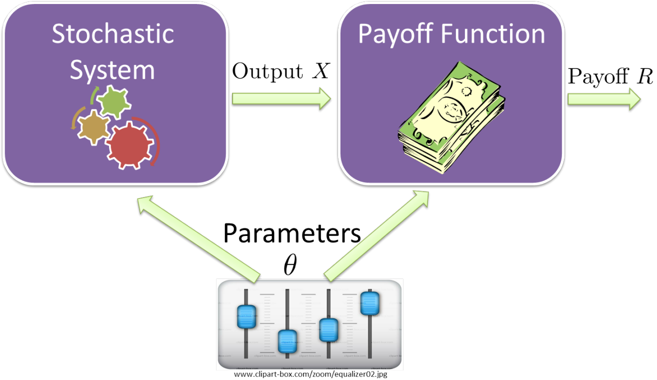
Risk Sensitive Policy Optimization



Standard objective

$$\max_{\theta} \mathbb{E}[R]$$

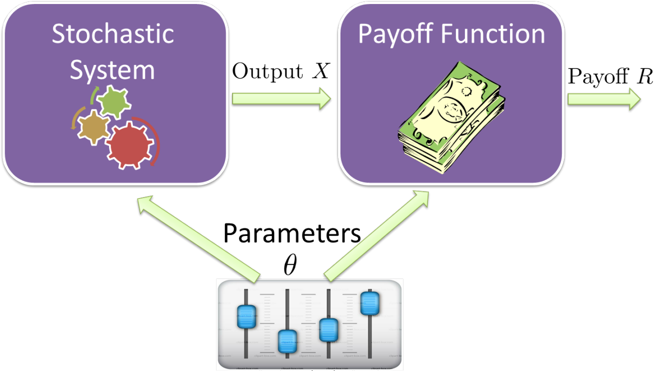
Risk Sensitive Policy Optimization



Risk-sensitive objective

$$\max_{\theta} \Phi_{\alpha}(R)$$

Risk Sensitive Policy Optimization



www.clipart-box.com/zoom/equalizer02.jpg

Risk-sensitive objective (example)

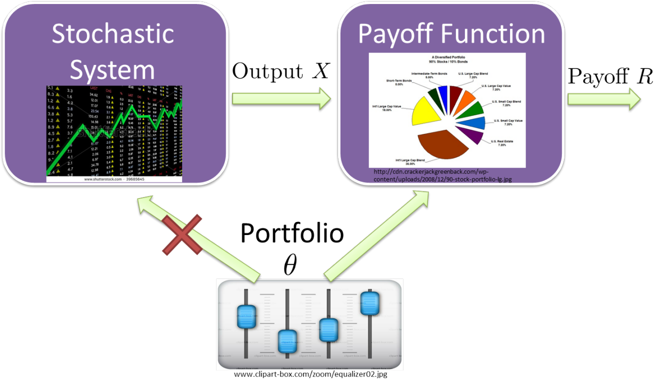
$$\max_{\theta} \mathbb{E}[R]$$

$$\text{s.t. } \Phi_{\alpha}(R) \geq \beta$$

Risk Sensitive Policy Optimization

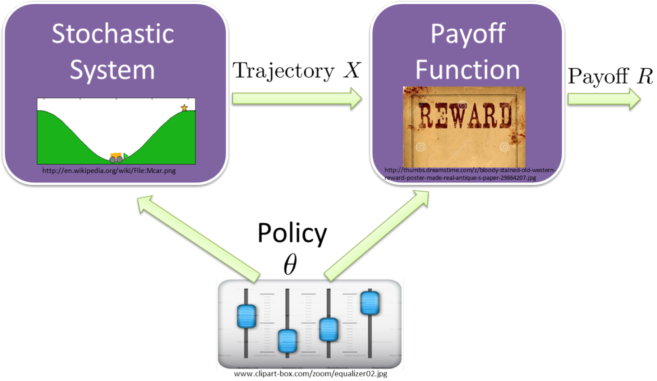
Previous work: θ does not affect trajectories X

- Stochastic programming



This work: θ controls the distribution of trajectories X

- Reinforcement learning (power grids, robotics, health, some financial problems...)



Our Approach

Approach overview

1. Estimate gradient $\nabla\Phi_\alpha(R)$ (w.r.t. θ)
   - New gradient formula
   - Sampling-based estimator
2. Update θ
   - Stochastic gradient descent

Gradient Estimation

Gradient estimation

- **Likelihood ratio** method (Glynn 1990; a.k.a. policy gradient)
- Estimate gradient $\nabla\mathbb{E}(X)$

$$
\begin{aligned}
\nabla\mathbb{E}(X) &= \nabla \int_{-\infty}^{\infty} f_X(x)xdx \\
&= \int_{-\infty}^{\infty} \nabla f_X(x)xdx \\
&= \int_{-\infty}^{\infty} \frac{\nabla f_X(x)}{f_X(x)} f_X(x)xdx \\
&= \mathbb{E}\left(\frac{\nabla f_X(X)}{f_X(X)}X\right)
\end{aligned}
$$

Gradient Estimation

Gradient estimation

- **Likelihood ratio** method (Glynn 1990; a.k.a. policy gradient)
- Estimate gradient $\nabla \mathbb{E}(X)$

$$
\begin{aligned}
\nabla \mathbb{E}(X) &= \nabla \int_{-\infty}^{\infty} f_X(x) x dx \\
&= \int_{-\infty}^{\infty} \nabla f_X(x) x dx \\
&= \int_{-\infty}^{\infty} \frac{\nabla f_X(x)}{f_X(x)} f_X(x) x dx \\
&= \mathbb{E}\left(\frac{\nabla f_X(X)}{f_X(X)} X \right) \\
&\approx \frac{1}{N} \sum_{1 \le i \le N} \frac{\nabla f_X(x_i)}{f_X(x_i)} x_i = \frac{1}{N} \sum_{1 \le i \le N} \nabla \log(f_X(x_i)) x_i
\end{aligned}
$$

Gradient Estimation

Gradient estimation - CVaR

- Estimate gradient $\nabla\Phi_\alpha(X)$
- Maybe:

$$\nabla\Phi_\alpha(X) \approx \frac{1}{\alpha N} \sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)} x_i$$

Gradient Estimation

Gradient estimation - CVaR

- Estimate gradient $\nabla \Phi_\alpha(X)$
- Maybe:

$$\nabla \Phi_\alpha(X) \approx \frac{1}{\alpha N} \sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)} x_i$$

- **No!**: Leibniz integral law.

Gradient Estimation

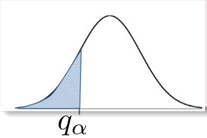
Gradient estimation - CVaR

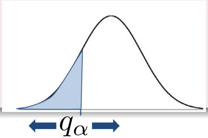
- Estimate gradient $\nabla \Phi_\alpha(X)$
- Maybe:

$$\nabla \Phi_\alpha(X) \approx \frac{1}{\alpha N} \sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)} x_i$$

- **No!**: Leibniz integral law.

$$\nabla \Phi_\alpha(X) = \nabla \int_{-\infty}^{q} \alpha^{-1} f_X(x) x dx$$





Gradient Estimation

Gradient estimation - CVaR

- Estimate gradient $\nabla\Phi_\alpha(X)$
- Maybe:

$$\nabla\Phi_\alpha(X) \approx \frac{1}{\alpha N} \sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)} x_i$$

- **No!**: Leibniz integral law.

$$\begin{aligned}\nabla\Phi_\alpha(X) &= \nabla \int_{-\infty}^{q} \alpha^{-1} f_X(x) x dx \\ &= \int_{-\infty}^{q} \alpha^{-1} \nabla f_X(x) x dx + \alpha^{-1} \nabla q f_X(q) q\end{aligned}$$

Gradient Estimation

Proposition

We have

$$\nabla \Phi_\alpha(R(X)) = \mathbb{E}\left[\frac{\nabla f_X(X)}{f_X(X)} (R(X) - q) \middle| R(X) \leq q \right]$$

Gradient Estimation

Proposition

We have

$$\nabla\Phi_\alpha(R(X)) = \mathbb{E}\left[\frac{\nabla f_X(X)}{f_X(X)}(R(X)-q)\middle| R(X) \le q\right]$$

Estimation algorithm

$$\nabla\Phi_\alpha(R(X)) \approx \frac{1}{\alpha N}\sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)}(R(x_i)-\hat{q})$$

where $\hat{q}$ is empirical quantile.

Gradient Estimation

Proposition

We have

$$\nabla\Phi_\alpha(R(X)) = \mathbb{E}\left[\frac{\nabla f_X(X)}{f_X(X)}(R(X)-q)\,\middle|\,R(X)\le q\right]$$

Estimation algorithm

$$\nabla\Phi_\alpha(R(X)) \approx \frac{1}{\alpha N}\sum_{\alpha N \text{ worst}} \frac{\nabla f_X(x_i)}{f_X(x_i)}(R(x_i)-\hat{q})$$

where $\hat{q}$ is empirical quantile.

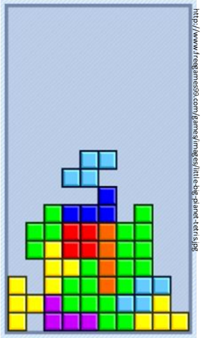
Guarantees

- Gradient estimate bias is $O(N^{-1/2})$
- Convergence w.p. 1 of SGD to local CVaR optimum

RL "Application"

Tetris

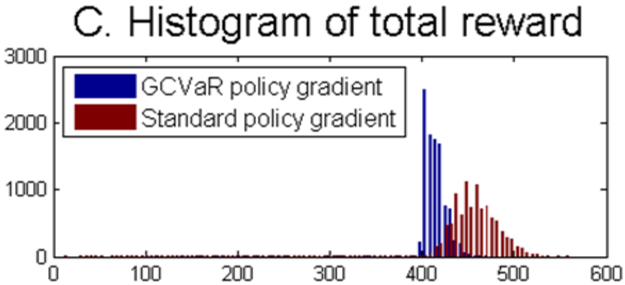
- Softmax policy, standard features (Thiery and Scherrer, 2009)
- Bonus for clearing multiple rows
- Compare standard policy gradient with CVaRSGD



RL “Application”

Tetris

- Softmax policy, standard features (Thiery and Scherrer, 2009)
- Bonus for clearing multiple rows
- Compare standard policy gradient with `CVaRSGD`

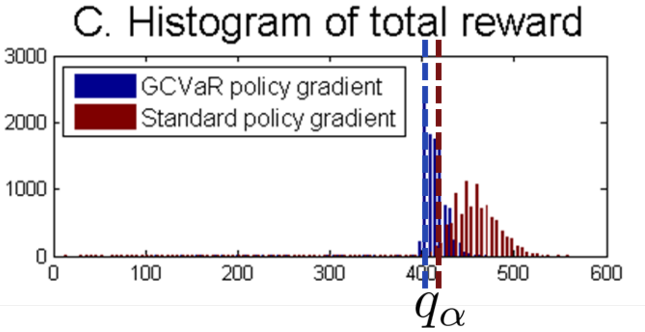


Avg. reward: **451** vs. 414

RL "Application"

Tetris

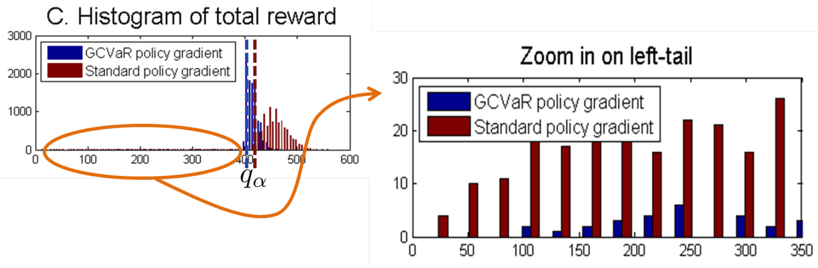
- Softmax policy, standard features (Thiery and Scherrer, 2009)
- Bonus for clearing multiple rows
- Compare standard policy gradient with CVaRSGD



Avg. reward: **451** vs. 414

Tetris

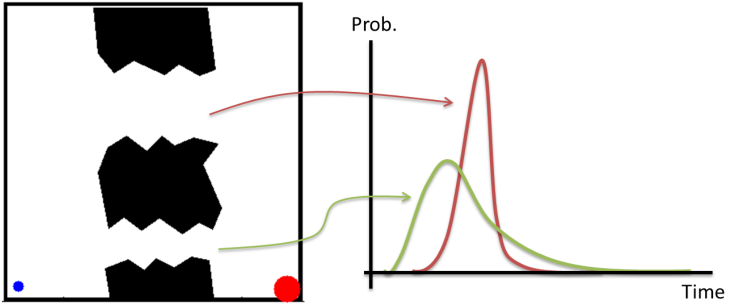
- Softmax policy, standard features (Thiery and Scherrer, 2009)
- Bonus for clearing multiple rows
- Compare standard policy gradient with `CVaRSGD`



Avg. reward: **451** vs. 414
Reward CVaR: 323 vs. **394**

Let's go a bit deeper...

Motivation - Revisited



Risk-awareness → robust policies!

CVaR Risk and Model Uncertainty

Temporally-budgeted perturbations

- Multiplicative perturbations

$$\hat{P}(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t, a_t) \cdot \delta_t(s_{t+1}|s_t, a_t)$$

- Uncertainty budget $\eta > 1$

$$\delta_1(s_1|s_0, a_0)\delta_2(s_2|s_1, a_1) \cdots \delta_T(s_T|s_{T-1}, a_{T-1}) \le \eta, \quad \forall s_0, \ldots, s_T, \forall a_0, \ldots, a_T$$

The worst cannot happen at every time!

- Set of possible perturbations Δ_η

CVaR Risk and Model Uncertainty

Temporally-budgeted perturbations

- Multiplicative perturbations

$$\hat{P}(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t, a_t) \cdot \delta_t(s_{t+1}|s_t, a_t)$$

- Uncertainty budget $\eta > 1$

$$\delta_1(s_1|s_0, a_0)\delta_2(s_2|s_1, a_1) \cdots \delta_T(s_T|s_{T-1}, a_{T-1}) \leq \eta,$$
$$\forall s_0, \ldots, s_T, \forall a_0, \ldots, a_T$$

The worst cannot happen at every time!

- Set of possible perturbations Δ_η

Theorem: CVaR = robustness in a broad sense

$$\text{CVaR}_{\frac{1}{\eta}}\left(\sum_{t=0}^{T} r(s_t)\right) = \inf_{(\delta_1,\ldots,\delta_T)\in\Delta_\eta} \mathbb{E}_{\hat{P}}\left[\sum_{t=0}^{T} r(s_t)\right]$$

Part 3
Recent advances in robustness

Two issues remain:

1. Uncertainty set construction
2. Online adaptivity

C. Tessler, Y. Efroni, and SM, ICML 2019

E. Derman, D. Mankowitz, T. Mann, and SM, UAI 2019.

Action Robustness

A trembling hand model

$$\pi_\alpha^{mix}(\pi, \pi') = \begin{cases} \pi, & \text{w.p. } 1 - \alpha. \\ \pi', & \text{w.p. } \alpha. \end{cases}$$

The policy π' is potentially adversarial.
Continuous extension: agent chooses a, adversary can modify to $(1 - \alpha)a + \alpha a'$.

Action Robustness

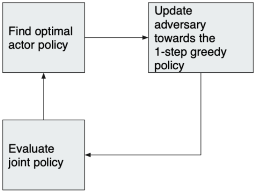
A trembling hand model

$$\pi_{\alpha}^{mix}(\pi, \pi') = \begin{cases} \pi, & \text{w.p. } 1 - \alpha. \\ \pi', & \text{w.p. } \alpha. \end{cases}$$

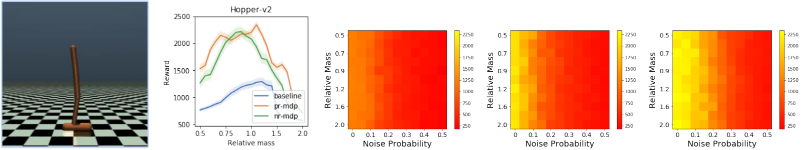
The policy π' is potentially adversarial.
Continuous extension: agent chooses a, adversary can modify to $(1 - \alpha)a + \alpha a'$.

AR-DDPG:

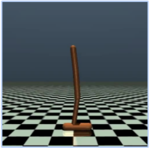
1. Train Actor
2. Train Adversary
3. Train Critic for the joint policy

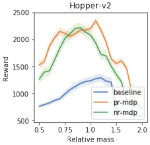


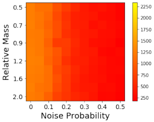
Some results

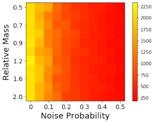


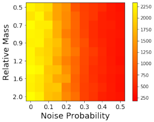
Some results











- Robustness: uncertainty + transfer to unseen domains
- A gradient based approach for robust reinforcement learning with convergence guarantees
- Does not require explicit definition of the uncertainty set
- Application to Deep RL

Posterior Uncertainty Sets: Online Construction of Uncertainty Sets

- Dirichlet prior on distribution over next states.
- Observation history $\mathcal{H}$ up to time h
- Time h - current step and t - current episode
- $$\widehat{\mathcal{P}}^h_{sa}(\psi_{sa}) = \{p_{sa} \in \Delta_{\mathcal{S}} : \|p_{sa} - \bar{p}_{sa}\|_1 \leq \psi_{sa}\}$$

  $\bar{p}_{sa} = \mathbb{E}[p_{sa} \mid \mathcal{H}]$ is the *nominal* transition.

This uncertainty set is

- Rectangular:
  $$\widehat{\mathcal{P}}^h = \bigotimes_{s \in \mathcal{S}, a \in \mathcal{A}} \widehat{\mathcal{P}}^h_{s,a}$$
- Updated **online** according to new observations

Uncertainty Robust Bellman Equation

- Posterior robust Q-value random variables satisfy a **robust Bellman recursion**

$$\hat{Q}^h_{sa} \overset{D}{=} r^h_{sa} + \gamma \inf_{p \in \widehat{\mathcal{P}}^h_{sa}} \sum_{s',a'} \pi^h_{s'a'} p_{sas'} \widehat{Q}^{h+1}_{s'a'}$$

- Posterior worst-case transition:
  $\widehat{p}^h_{sa} \in \arg\min_{p \in \widehat{\mathcal{P}}^h_{sa}} \sum_{s',a'} \pi^h_{s'a'} p_{sas'} \widehat{Q}^{h+1}_{s'a'}$

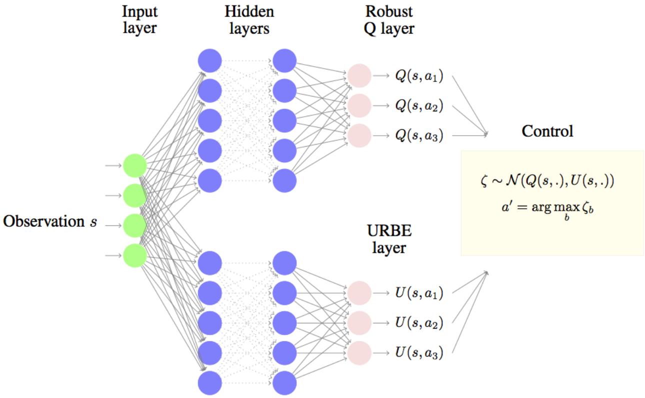
Theorem (Solution of URBE)

There exists a unique mapping w that satisfies the URBE:

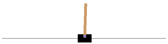
$$w^h_{sa} = \nu^h_{sa} + \gamma^2 \sum_{s' \in \mathcal{S}, a' \in \mathcal{A}} \pi^h_{s'a'} \mathbb{E}_t(\widehat{p}^h_{sas'}) w^{h+1}_{s'a'}$$

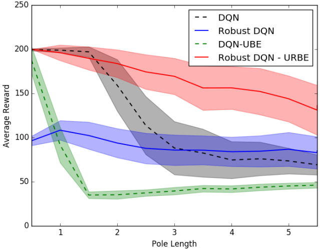
- Approximate Q-values as $\mathcal{N}(Q, diag(w))$.

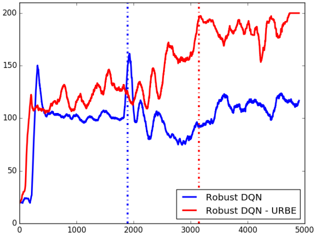
Deep Learning Approximation



Q-head uses robust TD error. URBE layer uses approximation.







- DQN/DQN-UBE: Overly **sensitive** to change of dynamics
- Robust DQN: Overly **conservative**

Discussion

- Adding URBE as a variance bonus leads to **less conservative solutions**
- DQN-URBE encourages **safe exploration** by implicitly updating the uncertainty set
- DQN-URBE is able to **adapt to changing dynamics** online
- Connections to Thompson sampling and pseudo-Bayesian approaches

Conclusion for Risk/robustness

Adaptivity and awareness are served by risk/robustness

- Handles 'unluckiness'
- Overcomes model misspecification
- CVaR = robustness
- Works online with deep models (scalability)

Take home message: solve robust/risk-sensitive MDPs

Scalable, works, and even has theoretical guarantees!

Applications: health, energy, finance, robotics, cyber, e-commerce

Conclusions

Extended Intelligence:

RL = data + representation + algorithms + Awareness + Accountability + Adaptivity + Life-cycle+ Scalability

Full autonomy is very far

RL works best:

- Highly stochastic + simulator
- High throughput control
- Real-world is neither

Conclusions

Extended Intelligence:

RL = data + representation + algorithms + Awareness + Accountability + Adaptivity + Life-cycle+ Scalability

Full autonomy is very far

RL works best:

- Highly stochastic + simulator
- High throughput control
- Real-world is neither

RL for EI: Key is applications and the lessons we learn from them (application = something you get paid real $ to do.)

Thanks

Looking for a postdoc? Email shie@technion.ac.il for details.

Joint work with:
E. Boccara (Technion), Y. Chow (Google AI), G. Dallal (Ford), Y. Efroni (Technion), M. Ghavamzadeh (FAIR), E. Gilboa (Ford), A. Hallak (Ford), M. Kozdoba (Technion), O. Maillard (CNRS Lille), D. Mankowitz (Google DeepMind), T. Mann (Google DeepMind), M. Pavone (Stanford), A. Tamar (Technion), C. Tessler (Technion), J. Tsitsiklis (MIT), H. Xu (Alibaba).