Reinforcement Learning China Summer School

RLChina 2020

Control as Inference

Zhanxing Zhu

zhanxing.zhu@pku.edu.cn

School of Mathematical Sciences, Peking University

July 31, 2020

Some of figures are adapted from Sergey Levine's paper and lecture slides

Contents

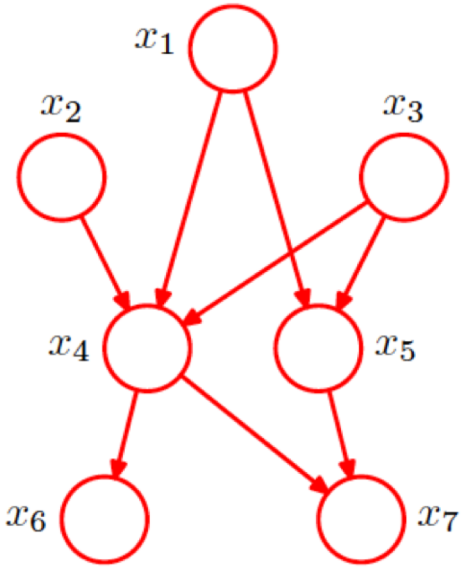
- Basic of (probabilistic) graphical models (GM)
  - D-separation
  - Variational inference
- Connection between RL and inference in GM
  - A graphical model for control as inference
- Maximum entropy RL and variational inference
- Soft Q-Learning
- Soft Actor-Critic

Probabilistic graphical model

- Use graph to represent joint probability distribution over multiple random variables
  - Express the dependency (via edges) between variables (nodes)
  - Simple, suitable for interpretability
- Directed GM: Joint distribution can be represented in product of factorized conditional distributions over its parents.

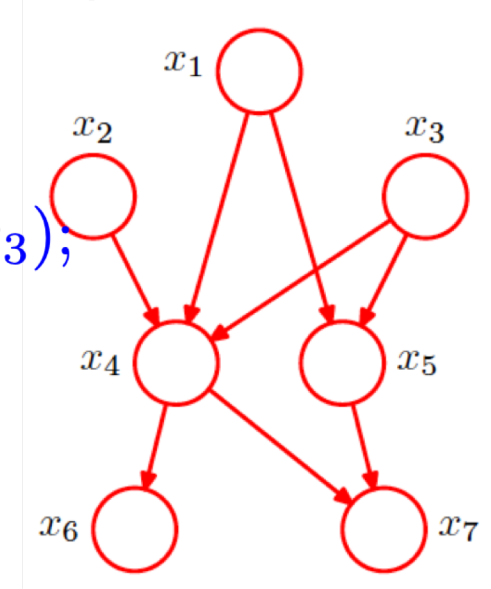
$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

$$p(x_1)p(x_2)p(x_3)p(x_4|x_1,x_2,x_3)p(x_5|x_1,x_3)p(x_6|x_4)p(x_7|x_4,x_5)$$



- Generative process
  - Aiming to obtain a sample according to the directed graphical models
  - **Ancestral sampling:** starting from lowest-numbering node and draw each variable given its already sampled parents.

1. Sample $x_1 \sim p(x_1)$, $x_2 \sim p(x_1)$, $x_3 \sim p(x_3)$;
2. Sample $x_4 \sim p(x_4|x_1, x_2, x_3)$, $x_5 \sim p(x_5|x_1, x_3)$;
3. Sample $x_6 \sim p(x_4|x_4)$, $x_7 \sim p(x_7|x_4, x_5)$.



Conditional independence

If $p(a|b,c) = p(a|c)$, then we say that a is conditionally independent of b given c, denoted as $a \perp b|c$.

How to test conditional dependence given a graphical model?

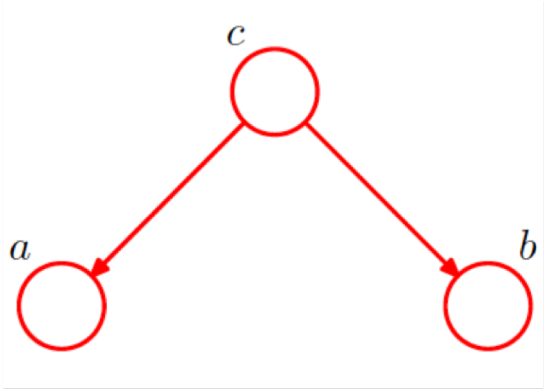
"D-separation" (Pearl, 1988)

Prof. Judea Pearl
ACM Turing Award Laureate, 2011

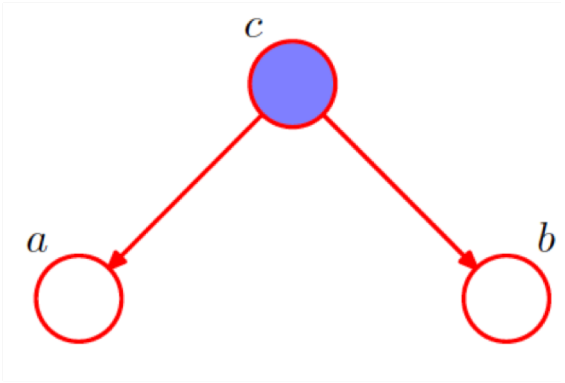
Three example graphs

- Example 1:

$$p(a, b, c) = p(a|c)p(b|c)p(c)$$



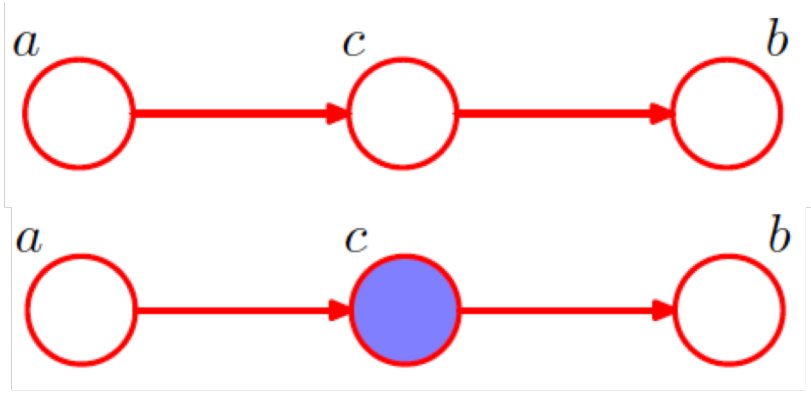
$$p(a, b) = \sum_c p(a|c)p(b|c)p(c) \longrightarrow a \not\perp\!\!\!\perp b \mid \emptyset$$



$$\begin{aligned} p(a, b|c) &= \frac{p(a, b, c)}{p(c)} \\ &= p(a|c)p(b|c) \end{aligned} \longrightarrow a \perp\!\!\!\perp b \mid c$$

tail-to-tail

- Example 2:



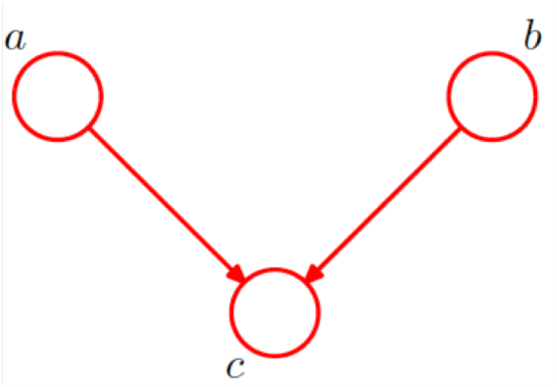
head-to-tail

$$p(a,b,c) = p(a)p(c|a)p(b|c)$$

$$p(a,b) = p(a)\sum_c p(c|a)p(b|c) = p(a)p(b|a) \longrightarrow a \not\perp\!\!\!\perp b \mid \emptyset$$

$$\begin{aligned} p(a,b|c) &= \frac{p(a,b,c)}{p(c)} \\ &= \frac{p(a)p(c|a)p(b|c)}{p(c)} \\ &= p(a|c)p(b|c) \end{aligned} \longrightarrow a \perp\!\!\!\perp b \mid c$$

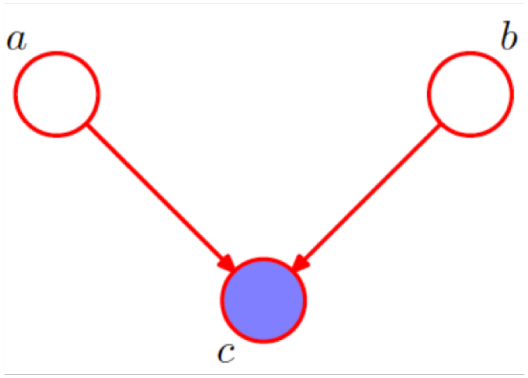
- Example 3:



$$p(a,b,c) = p(a)p(b)p(c|a,b)$$

$$p(a,b) = p(a)p(b) \longrightarrow a \perp\!\!\!\perp b \mid \emptyset$$

$$p(a,b|c) = \frac{p(a,b,c)}{p(c)} = \frac{p(a)p(b)p(c|a,b)}{p(c)} \longrightarrow a \not\perp\!\!\!\perp b \mid c$$



head-to-head

D-separation

- Consider all paths from set of nodes A to B, we call any path "***blocked***" if they includes a node such that either

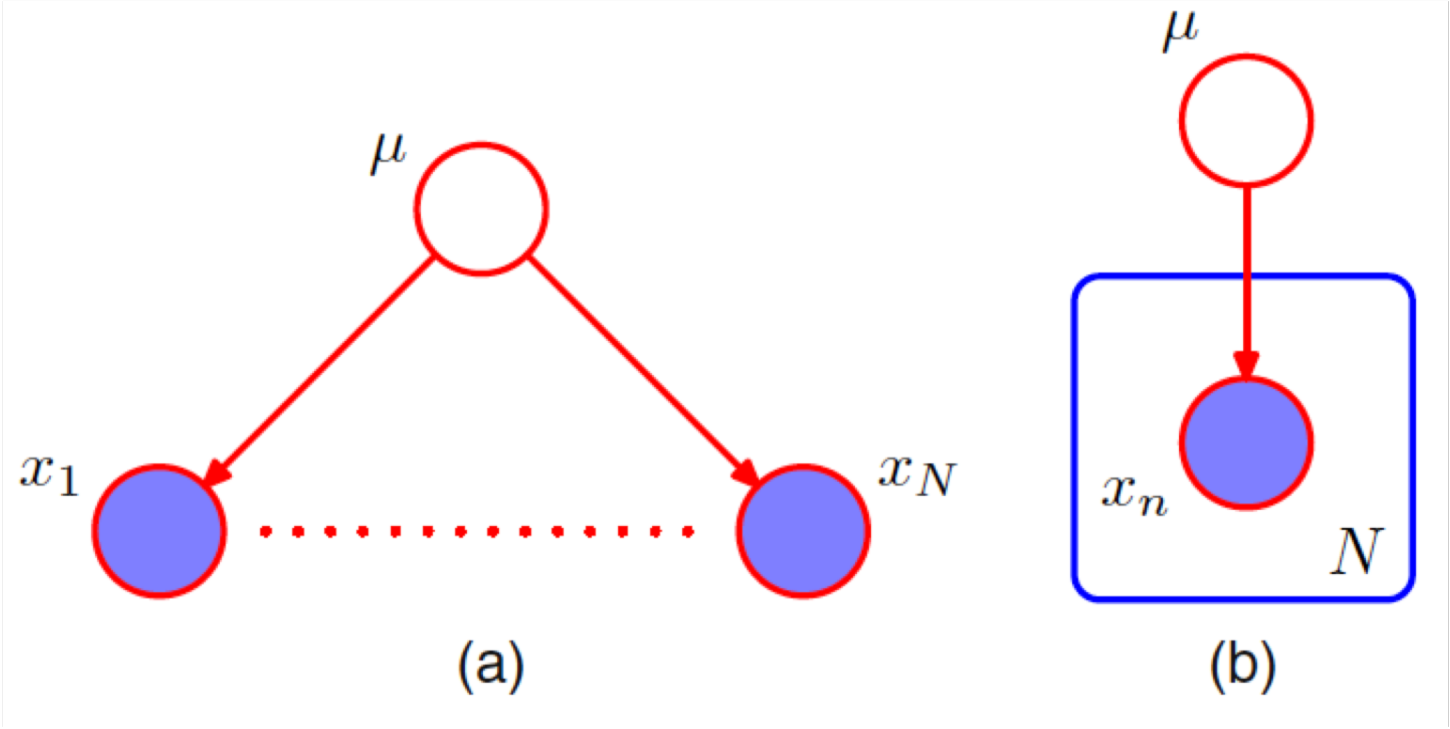
(a) the arrows on the path meet either head-to-tail or tail-to-tail at the node, and the node is in the set C, or

(b) the arrows meet head-to-head at the node, and neither the node, nor any of its descendants, is in the set C.

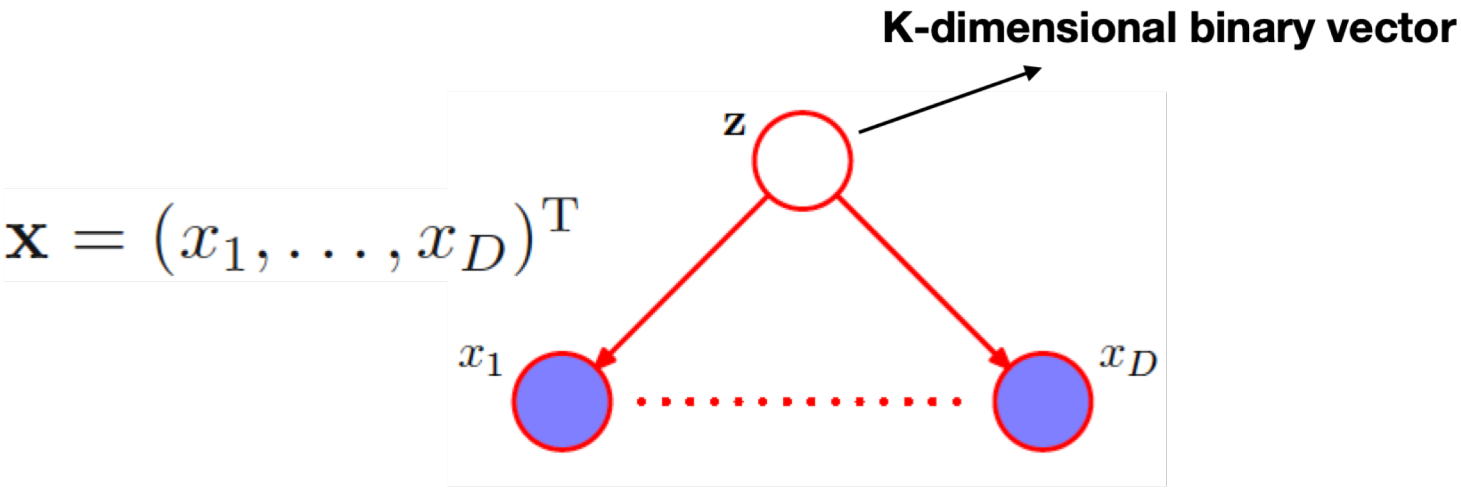
If all paths are blocked, then A is said to be d-separated from B by C, and the joint distribution over all of the variables in the graph will satisfy $A \perp\!\!\!\perp B \mid C$.

- Application of D-separation: **i.i.d Gaussian with mean**

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu)$$



- Application of D-separation: **Naive Bayes classifier**



x_i and x_j are conditionally independent given class variable z.

Could you write out the joint distribution?

Bayesian inference

$$P(\text{hypothesis}|\text{data}) = \frac{P(\text{data}|\text{hypothesis})P(\text{hypothesis})}{P(\text{data})}$$

Rev'd Thomas Bayes (1702–1761)

- Bayes rule tells us how to do inference about hypotheses from data.
- Learning and prediction can be seen as forms of inference.

Approximate inference

- One of central tasks: Bayesian Inference

  can be latent variables or model parameters in general

  - Evaluation of posterior distribution p(Z | X), and expectation of certain function w.r.t. this posterior. E.g. in EM, expected log likelihood w.r.t. p(Z | X)

$$\mathbb{E}_{p(\boldsymbol{Z}|\boldsymbol{X})}[f(\boldsymbol{Z})]$$

  - Unfortunately, the expectation is typically intractable.

  - Approximation schemes are required

    - Deterministic techniques: **Laplace approximation**, **variational inference**

    - Stochastic techniques: **Markov Chain Monte Carlo** (MCMC)

Variational inference

- The idea
  - Using a family of parametric distribution to approximate the posterior distribution

$$q(z|\phi) \sim p(z|x)$$

  - Turn an inference problem to an optimization problem
- Decompose the log marginal probability:

$$\ln p(\mathbf{X}) = \mathcal{L}(q) + \mathrm{KL}(q\|p)$$

Evidence lower bound (ELBO):

$$\mathcal{L}(q) = \int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

$$\mathrm{KL}(q\|p) = -\int q(\mathbf{Z}) \ln \left\{ \frac{p(\mathbf{Z}|\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z}$$

Phi convex

$$\varphi(\mathrm{E}[X]) \leq \mathrm{E}[\varphi(X)].$$

- The ELBO can be derived either by the non-negativity of KL distance or the Jensen's inequality.
- Choose a restricted family of variational distribution
  - Then optimize w.r.t. (the parameters of) variational distribution.

Connection between RL and inference in GM

- The connection
  - Maximum entropy RL is equivalent to exact/variational inference in GM
- Benefits
  - A natural exploration strategy based on entropy maximization
  - Deploy powerful approximate inference algorithms to solve reinforcement learning problems
  - An appealing probabilistic interpretation for the meaning of the reward function, and its effect on the optimal policy

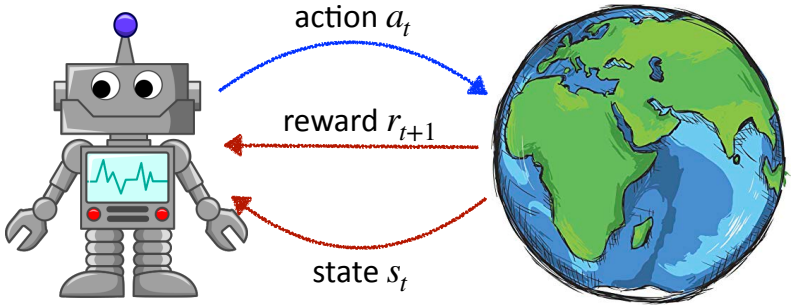
The RL problem

- The optimization problem: policy search problem

$$\theta^{\star} = \arg\max_{\theta} \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim p(\mathbf{s}_t, \mathbf{a}_t | \theta)} \left[r(\mathbf{s}_t, \mathbf{a}_t) \right]$$

- Trajectory distribution

$$p(\tau) = p(\mathbf{s}_1, \mathbf{a}_t, \ldots, \mathbf{s}_T, \mathbf{a}_T | \theta) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{a}_t | \mathbf{s}_t, \theta) p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)$$



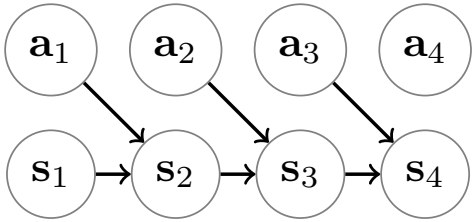
The agent is to maximize the cumulative reward

All goals and purposes can be described by the maximization of the expected cumulative reward

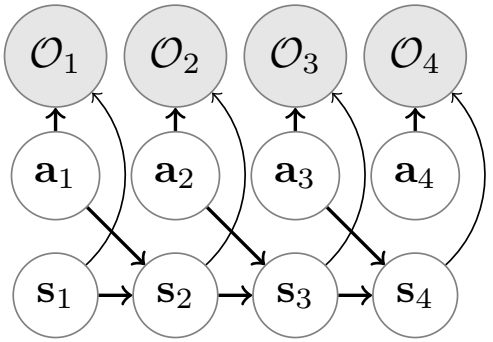
The graphical model for control as inference

- The idea: incorporating reward by introducing the optimality variable O
  - Ot = 1 denotes that time step t is *optimal*

$$p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t))$$



(a) graphical model with states and actions



(b) graphical model with optimality variables

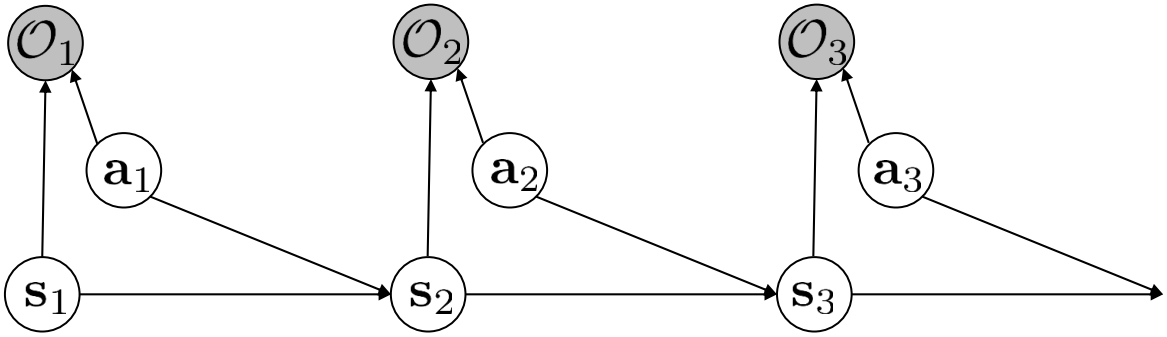
Posterior distribution over trajectory

$$p(\tau|\mathbf{o}_{1:T}) \propto p(\tau, \mathbf{o}_{1:T}) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= p(\mathbf{s}_1) \prod_{t=1}^{T} \exp(r(\mathbf{s}_t, \mathbf{a}_t)) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= \left[p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \right] \exp\left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right)$$

Policy search as probabilistic inference



- Given the GM, infer the optimal policy
  - Similar to inference in HMM-type GMs
  - Recursive message passing alg.

$$\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)$$

$$\beta_t(\mathbf{s}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t)$$

Uniform action prior

$$\beta_t(\mathbf{s}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t) = \int_{\mathcal{A}} p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{a}_t|\mathbf{s}_t) d\mathbf{a}_t = \int_{\mathcal{A}} \beta_t(\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{a}_t|\mathbf{s}_t) d\mathbf{a}_t$$

$$\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t) = \int_{\mathcal{S}} \beta_{t+1}(\mathbf{s}_{t+1}) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) d\mathbf{s}_{t+1}$$

- Infer the optimal policy: recursive message passing

for $t = T-1$ to 1:

$\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[\beta_{t+1}(\mathbf{s}_{t+1})]$ compute recursively from $t = T$ to $t = 1$

$\beta_t(\mathbf{s}_t) = E_{\mathbf{a}_t \sim p(\mathbf{a}_t|\mathbf{s}_t)}[\beta_t(\mathbf{s}_t, \mathbf{a}_t)]$

Soft value functions

let $V_t(\mathbf{s}_t) = \log \beta_t(\mathbf{s}_t)$

let $Q_t(\mathbf{s}_t, \mathbf{a}_t) = \log \beta_t(\mathbf{s}_t, \mathbf{a}_t)$

log of β_t is "Q-function-like"

$$p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{t:T}) = \frac{p(\mathbf{s}_t, \mathbf{a}_t|\mathcal{O}_{t:T})}{p(\mathbf{s}_t|\mathcal{O}_{t:T})} = \frac{p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)p(\mathbf{a}_t|\mathbf{s}_t)p(\mathbf{s}_t)}{p(\mathcal{O}_{t:T}|\mathbf{s}_t)p(\mathbf{s}_t)} \propto \frac{p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)}{p(\mathcal{O}_{t:T}|\mathbf{s}_t)} = \frac{\beta_t(\mathbf{s}_t, \mathbf{a}_t)}{\beta_t(\mathbf{s}_t)}$$

- Or use Log-space messages, i.e. value function

for $t = T-1$ to 1:

$$Q_t(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + E[V_{t+1}(\mathbf{s}_{t+1})]$$

$$V_t(\mathbf{s}_t) = \log \int \exp(Q_t(\mathbf{s}_t, \mathbf{a}_t))\mathbf{a}_t$$

$$\pi(\mathbf{a}_t|\mathbf{s}_t) = \frac{\beta_t(\mathbf{s}_t, \mathbf{a}_t)}{\beta_t(\mathbf{s}_t)} \qquad V_t(\mathbf{s}_t) = \log \beta_t(\mathbf{s}_t)$$

$$Q_t(\mathbf{s}_t, \mathbf{a}_t) = \log \beta_t(\mathbf{s}_t, \mathbf{a}_t)$$

$$\pi(\mathbf{a}_t|\mathbf{s}_t) = \exp(Q_t(\mathbf{s}_t, \mathbf{a}_t) - V_t(\mathbf{s}_t)) = \exp(A_t(\mathbf{s}_t, \mathbf{a}_t))$$

Implicit objective of inference procedure: maximum entropy RL

- Minimizing KL-divergence between posterior trajectory distribution

$$\hat{p}(\tau) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t)$$

$$D_{\mathrm{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$

$$p(\tau) = \left[p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right] \exp\left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)\right)$$

Maximum entropy RL

$$-D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim\hat{p}(\mathbf{s}_t,\mathbf{a}_t))}[r(\mathbf{s}_t, \mathbf{a}_t) + \mathcal{H}(\pi(\mathbf{a}_t|\mathbf{s}_t))]$$

- Dynamic programing approach: optimizing T-th step first

$$E_{(\mathbf{s}_T,\mathbf{a}_T)\sim\hat{p}(\mathbf{s}_T,\mathbf{a}_T)}[r(\mathbf{s}_T, \mathbf{a}_T) - \log \pi(\mathbf{a}_T|\mathbf{s}_T)] =$$

$$E_{\mathbf{s}_T\sim\hat{p}(\mathbf{s}_T)}\left[-D_{\mathrm{KL}}\left(\pi(\mathbf{a}_T|\mathbf{s}_T)\|\frac{1}{\exp(V(\mathbf{s}_T))}\exp(r(\mathbf{s}_T, \mathbf{a}_T))\right) + V(\mathbf{s}_T)\right]$$

$$\pi(\mathbf{a}_T|\mathbf{s}_T) = \exp\left(r(\mathbf{s}_T, \mathbf{a}_T) - V(\mathbf{s}_T)\right)$$

- The recursive case: the policy must optimize the two terms

$$E_{(\mathbf{s}_t,\mathbf{a}_t)\sim\hat{p}(\mathbf{s}_t,\mathbf{a}_t)}[r(\mathbf{s}_t,\mathbf{a}_t) - \log\pi(\mathbf{a}_t|\mathbf{s}_t)] + E_{(\mathbf{s}_t,\mathbf{a}_t)\sim\hat{p}(\mathbf{s}_t,\mathbf{a}_t)}[E_{\mathbf{s}_{t+1}\sim p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V(\mathbf{s}_{t+1})]]$$

$$\downarrow$$

$$E_{\mathbf{s}_t\sim\hat{p}(\mathbf{s}_t)}\left[-D_{\mathrm{KL}}\left(\pi(\mathbf{a}_t|\mathbf{s}_t)\|\frac{1}{\exp(V(\mathbf{s}_t))}\exp(Q(\mathbf{s}_t,\mathbf{a}_t))\right) + V(\mathbf{s}_t)\right]$$

$$\downarrow$$

$$\pi(\mathbf{a}_t|\mathbf{s}_t) = \exp\left(Q(\mathbf{s}_t,\mathbf{a}_t) - V(\mathbf{s}_t)\right)$$

$$Q(\mathbf{s}_t,\mathbf{a}_t) = r(\mathbf{s}_t,\mathbf{a}_t) + E_{\mathbf{s}_{t+1}\sim p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V(\mathbf{s}_{t+1})]$$

$$V(\mathbf{s}_t) = \log\int_{\mathcal{A}}\exp(Q(\mathbf{s}_t,\mathbf{a}_t))d\mathbf{a}_t,$$

Maximum entropy RL and variational inference

- V.I.: using a simpler variational distribution to approximate the posterior distribution

$$p(\tau) = \left[p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \right] \exp\left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right)$$

$$q(\tau) = q(\mathbf{s}_1) \prod_{t=1}^{T} q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) q(\mathbf{a}_t|\mathbf{s}_t)$$

- Let $q(\mathbf{s}_1) = p(\mathbf{s}_1)$ and $q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$

ELBO

$$\begin{aligned}
\log p(\mathcal{O}_{1:T}) &= \log \int \int p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) d\mathbf{s}_{1:T} d\mathbf{a}_{1:T} \\
&= \log \int \int p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \frac{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})}{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} d\mathbf{s}_{1:T} d\mathbf{a}_{1:T} \\
&= \log E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\frac{p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T})}{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \right] \\
&\geq E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\log p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) - \log q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \right]
\end{aligned}$$

$$\log p(\mathcal{O}_{1:T}) \geq E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t) \right]$$

Soft Q-learning

- Parameterize Q-message: $Q_\phi(\mathbf{s}_t, \mathbf{a}_t)$
- The objective function

$$\mathcal{E}(\phi) = E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)}\left[\left(r(\mathbf{s}_t,\mathbf{a}_t) + E_{q(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V_\phi(\mathbf{s}_{t+1})] - Q_\phi(\mathbf{s}_t,\mathbf{a}_t)\right)^2\right]$$

$$V(\mathbf{s}_t) = \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_t,\mathbf{a}_t))d\mathbf{a}_t$$

- Soft Q-learning

$$\phi \leftarrow \phi - \alpha E\left[\frac{dQ_\phi}{d\phi}(\mathbf{s}_t,\mathbf{a}_t)\left(Q_\phi(\mathbf{s}_t,\mathbf{a}_t) - \left(r(\mathbf{s}_t,\mathbf{a}_t) + \log\int_{\mathcal{A}}\exp(Q(\mathbf{s}_{t+1},\mathbf{a}_{t+1}))d\mathbf{a}_{t+1}\right)\right)\right]$$

- Standard Q-learning

$$\phi \leftarrow \phi - \alpha E\left[\frac{dQ_\phi}{d\phi}(\mathbf{s}_t,\mathbf{a}_t)\left(Q_\phi(\mathbf{s}_t,\mathbf{a}_t) - \left(r(\mathbf{s}_t,\mathbf{a}_t) + \max_{\mathbf{a}_{t+1}} Q_\phi(\mathbf{s}_{t+1},\mathbf{a}_{t+1})\right)\right)\right]$$

The alg. of soft Q-learning

soft Q-learning: $\phi \leftarrow \phi + \alpha \nabla_\phi Q_\phi(\mathbf{s}, \mathbf{a})(r(\mathbf{s}, \mathbf{a}) + \gamma V(\mathbf{s}') - Q_\phi(\mathbf{s}, \mathbf{a}))$

target value: $V(\mathbf{s}') = \text{soft}\max_{\mathbf{a}'} Q_\phi(\mathbf{s}', \mathbf{a}') = \log \int \exp(Q_\phi(\mathbf{s}', \mathbf{a}'))d\mathbf{a}'$

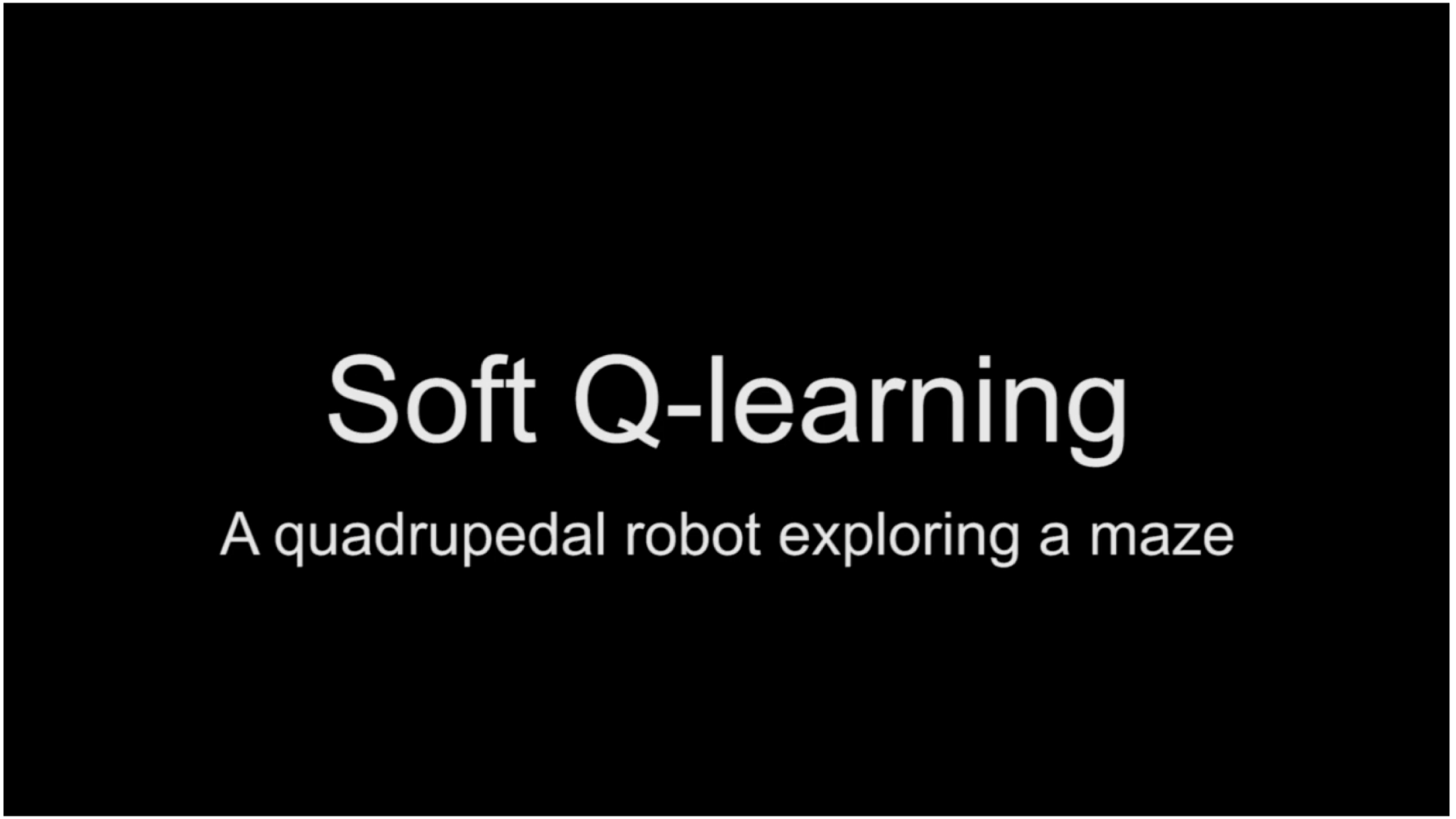
$\pi(\mathbf{a}|\mathbf{s}) = \exp(Q_\phi(\mathbf{s}, \mathbf{a}) - V(\mathbf{s})) = \exp(A(\mathbf{s}, \mathbf{a}))$

1. take some action $\mathbf{a}_i$ and observe $(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)$, add it to $\mathcal{R}$
2. sample mini-batch $\{\mathbf{s}_j, \mathbf{a}_j, \mathbf{s}'_j, r_j\}$ from $\mathcal{R}$ uniformly
3. compute $y_j = r_j + \gamma \text{soft}\max_{\mathbf{a}'_j} Q_{\phi'}(\mathbf{s}'_j, \mathbf{a}'_j)$ using *target* network $Q_{\phi'}$
4. $\phi \leftarrow \phi - \alpha \sum_j \frac{dQ_\phi}{d\phi}(\mathbf{s}_j, \mathbf{a}_j)(Q_\phi(\mathbf{s}_j, \mathbf{a}_j) - y_j)$
5. update ϕ': copy ϕ every N steps, or Polyak average $\phi' \leftarrow \tau\phi' + (1-\tau)\phi$

- Evaluation of the integral
  - Easy for discrete actions
  - Continuous actions: extra approximate methods: v.i. or sampling
    - Stein variational gradient descent (SVGD, Liu & Wang 2016)

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy- based policies. In *International Conference on Machine Learning (ICML)*.

An example of soft Q-learning



https://sites.google.com/view/softqlearning/home

Maximum entropy policy gradient

- Parameterize policy: $q_\theta(\mathbf{a}_t|\mathbf{s}_t)$
- The objective function

$$J(\theta) = \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[r(\mathbf{s}_t,\mathbf{a}_t) - \mathcal{H}(q_\theta(\mathbf{a}_t|\mathbf{s}_t)) \right]$$

- The policy gradient

$$\begin{aligned}
\nabla_\theta J(\theta) &= \sum_{t=1}^{T} \nabla_\theta E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[r(\mathbf{s}_t,\mathbf{a}_t) + \mathcal{H}(q_\theta(\mathbf{a}_t|\mathbf{s}_t)) \right] \\
&= \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[\nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t) \left(\sum_{t'=t}^{T} r(\mathbf{s}_{t'},\mathbf{a}_{t'}) - \underline{\log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'})} - 1 \right) \right] \\
&= \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[\nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t) \left(\sum_{t'=t}^{T} r(\mathbf{s}_{t'},\mathbf{a}_{t'}) - \log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'}) - \underline{b(\mathbf{s}_{t'})} \right) \right] \\
\nabla_\theta J(\theta) &= \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[\nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t) \hat{A}(\mathbf{s}_t,\mathbf{a}_t) \right]
\end{aligned}$$

Encourage maximum entropy

State-dependent baseline

Soft Actor-Critic

- Parameterize both policy and value functions (off-policy)
- Soft value function

Replay buffer

$$J_V(\psi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}} \left[\frac{1}{2} \left(V_\psi(\mathbf{s}_t) - \mathbb{E}_{\mathbf{a}_t \sim \pi_\phi} \left[Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \log \pi_\phi(\mathbf{a}_t | \mathbf{s}_t) \right] \right)^2 \right]$$

$$\hat{\nabla}_\psi J_V(\psi) = \nabla_\psi V_\psi(\mathbf{s}_t) \left(V_\psi(\mathbf{s}_t) - Q_\theta(\mathbf{s}_t, \mathbf{a}_t) + \log \pi_\phi(\mathbf{a}_t | \mathbf{s}_t) \right)$$

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \hat{Q}(\mathbf{s}_t, \mathbf{a}_t) \right)^2 \right]$$

with

$$\hat{Q}(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \, \mathbb{E}_{\mathbf{s}_{t+1} \sim p} \left[V_{\bar{\psi}}(\mathbf{s}_{t+1}) \right]$$

Exponential average for stability

$$\hat{\nabla}_\theta J_Q(\theta) = \nabla_\theta Q_\theta(\mathbf{a}_t, \mathbf{s}_t) \left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - r(\mathbf{s}_t, \mathbf{a}_t) - \gamma V_{\bar{\psi}}(\mathbf{s}_{t+1}) \right)$$

- Policy

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}} \left[\mathrm{D}_{\mathrm{KL}} \left(\pi_\phi(\,\cdot\,|\mathbf{s}_t) \,\middle\|\, \frac{\exp\left(Q_\theta(\mathbf{s}_t,\,\cdot\,)\right)}{Z_\theta(\mathbf{s}_t)} \right) \right]$$

Reparameterization

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}, \epsilon_t \sim \mathcal{N}} \left[\log \pi_\phi(f_\phi(\epsilon_t; \mathbf{s}_t)|\mathbf{s}_t) - Q_\theta(\mathbf{s}_t, f_\phi(\epsilon_t; \mathbf{s}_t)) \right]$$

$$\begin{aligned} \hat{\nabla}_\phi J_\pi(\phi) &= \nabla_\phi \log \pi_\phi(\mathbf{a}_t|\mathbf{s}_t) \\ &\quad + (\nabla_{\mathbf{a}_t} \log \pi_\phi(\mathbf{a}_t|\mathbf{s}_t) - \nabla_{\mathbf{a}_t} Q(\mathbf{s}_t, \mathbf{a}_t)) \nabla_\phi f_\phi(\epsilon_t; \mathbf{s}_t) \end{aligned}$$

Algorithm 1 Soft Actor-Critic

Initialize parameter vectors ψ, $\bar{\psi}$, θ, ϕ.
for each iteration **do**
  for each environment step **do**
    $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$
    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$
  end for
  for each gradient step **do**
    $\psi \leftarrow \psi - \lambda_V \hat{\nabla}_\psi J_V(\psi)$
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    $\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}$
  end for
end for

Minimum of two independent Qs to reduce positive bias in policy improvement

Tuomas Haarnoja*, Aurick Zhou*, Kristian Hartikainen*, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, Sergey Levine. *Soft Actor-Critic Algorithms and Applications.* arXiv preprint, 2018.

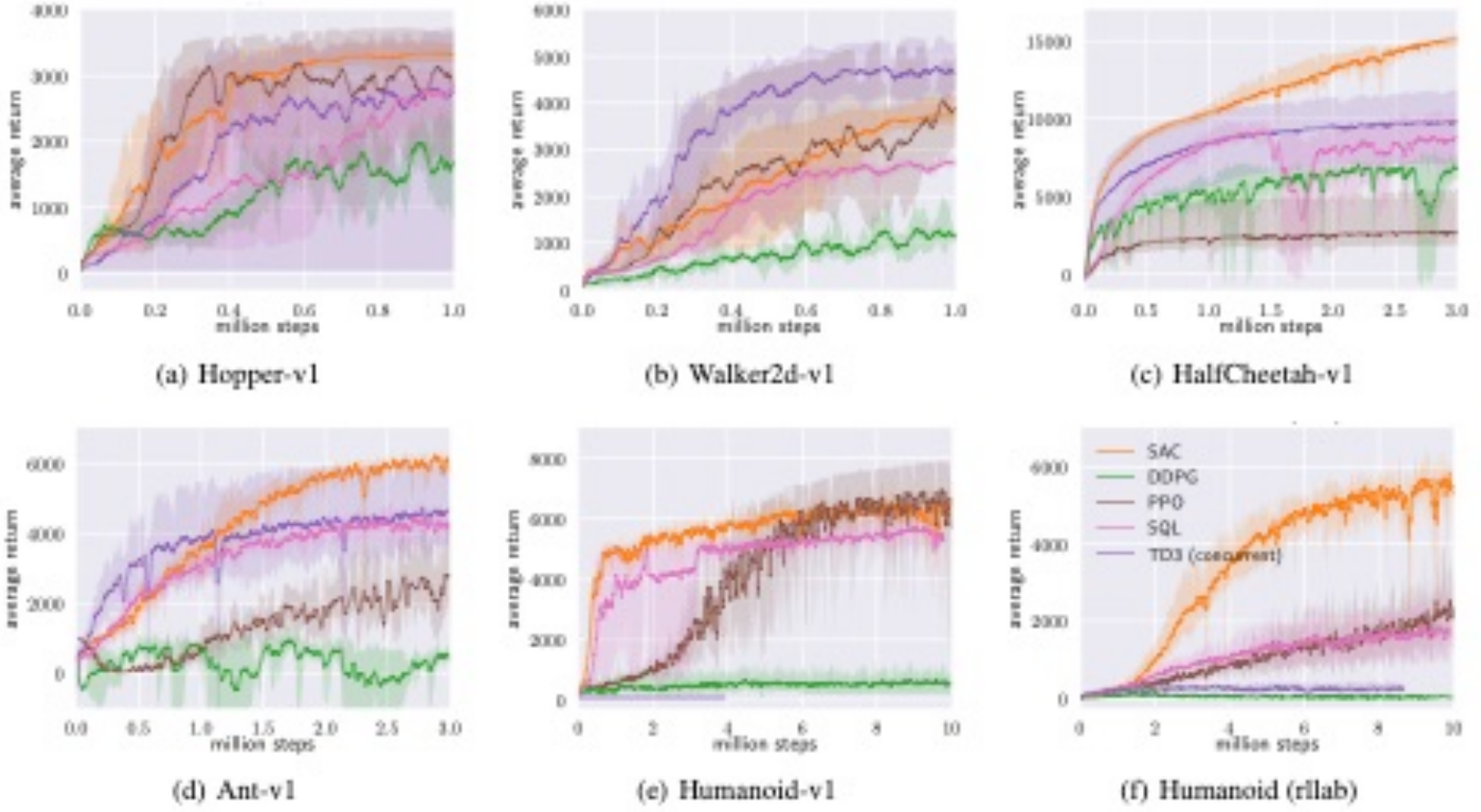


Figure 1. Training curves on continuous control benchmarks. Soft actor-critic (yellow) performs consistently across all tasks and outperforming both on-policy and off-policy methods in the most challenging tasks.

Summary

- RL can be viewed as inference in graphical models
  - Value function is backward message
  - Maximize reward and entropy
- Induced algorithms
  - Soft Q-learning
  - Soft policy gradient
  - Soft actor-critic
- Benefits of soft optimality
  - Encourage exploration and prevent entropy collapse
  - Better robustness due to more coverage of states
  - More general, can reduce to hard optimality when rewards are large

Reference

- Pattern recognition and machine learning by Bishop 2006
- Levine, S., 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.
- Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy- based policies. In *International Conference on Machine Learning (ICML)*.
- Tuomas Haarnoja*, Aurick Zhou*, Kristian Hartikainen*, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, Sergey Levine. *Soft Actor-Critic Algorithms and Applications.* arXiv preprint, 2018.
- Kappen. (2009). Optimal control as a graphical model inference problem: frames control as an inference problem in a graphical model.
- Ziebart. (2010). Modeling interaction via the principle of maximal causal entropy: connection between soft optimality and maximum entropy modeling.

Thanks!