IMPALA: Scalable Distributed Deep-RL with **Imp**ortance Weighted **A**ctor-**L**earner **A**rchitectures

ICML 2018

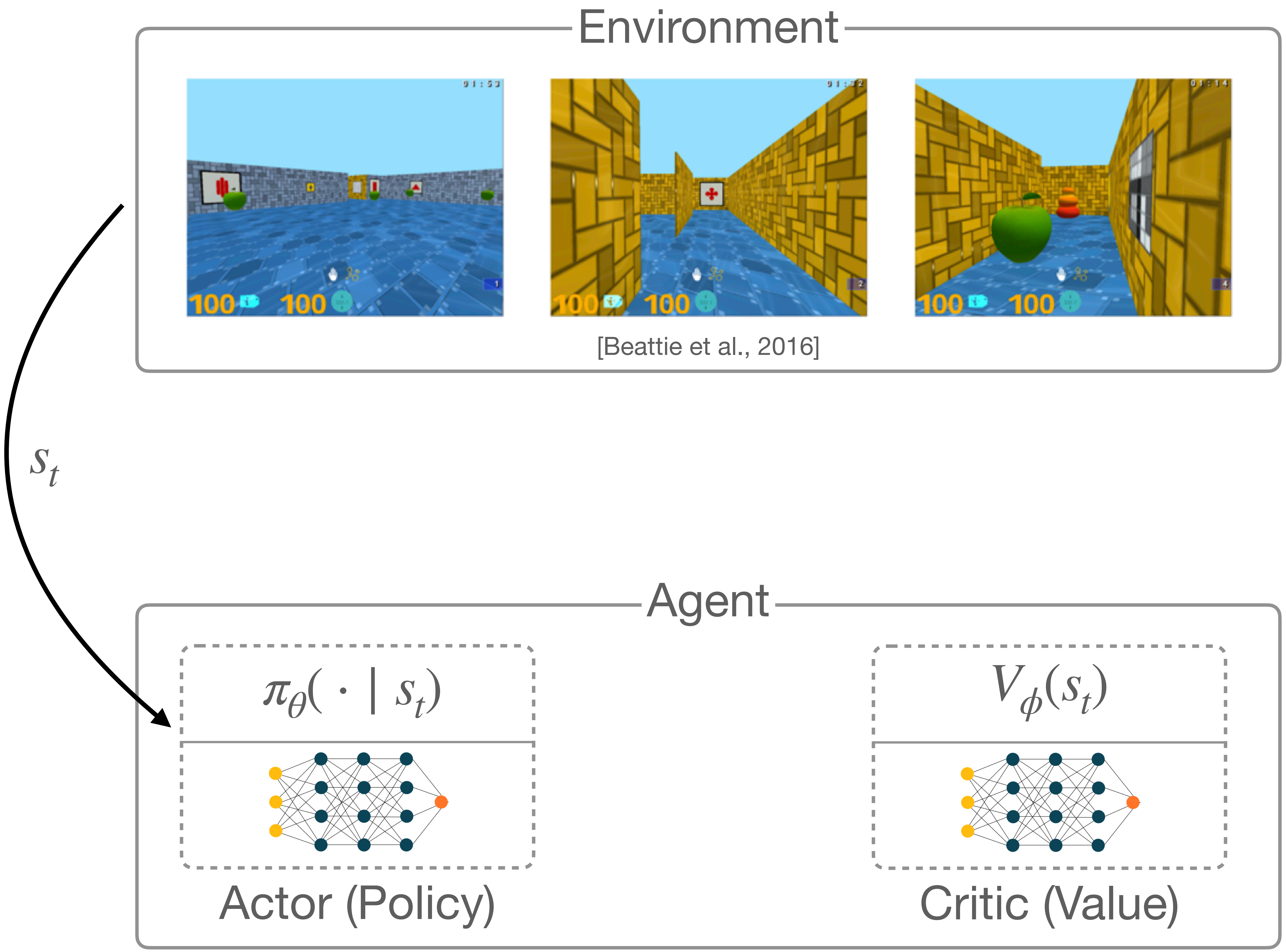
Lasse Espeholt *, Hubert Soyer *, Remi Munos *, Karen Simonyan, Volodymyr Mnih , Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, Koray Kavukcuoglu

Google DeepMind

Presenter - Aditya Thimmaiah

* Equal Contribution

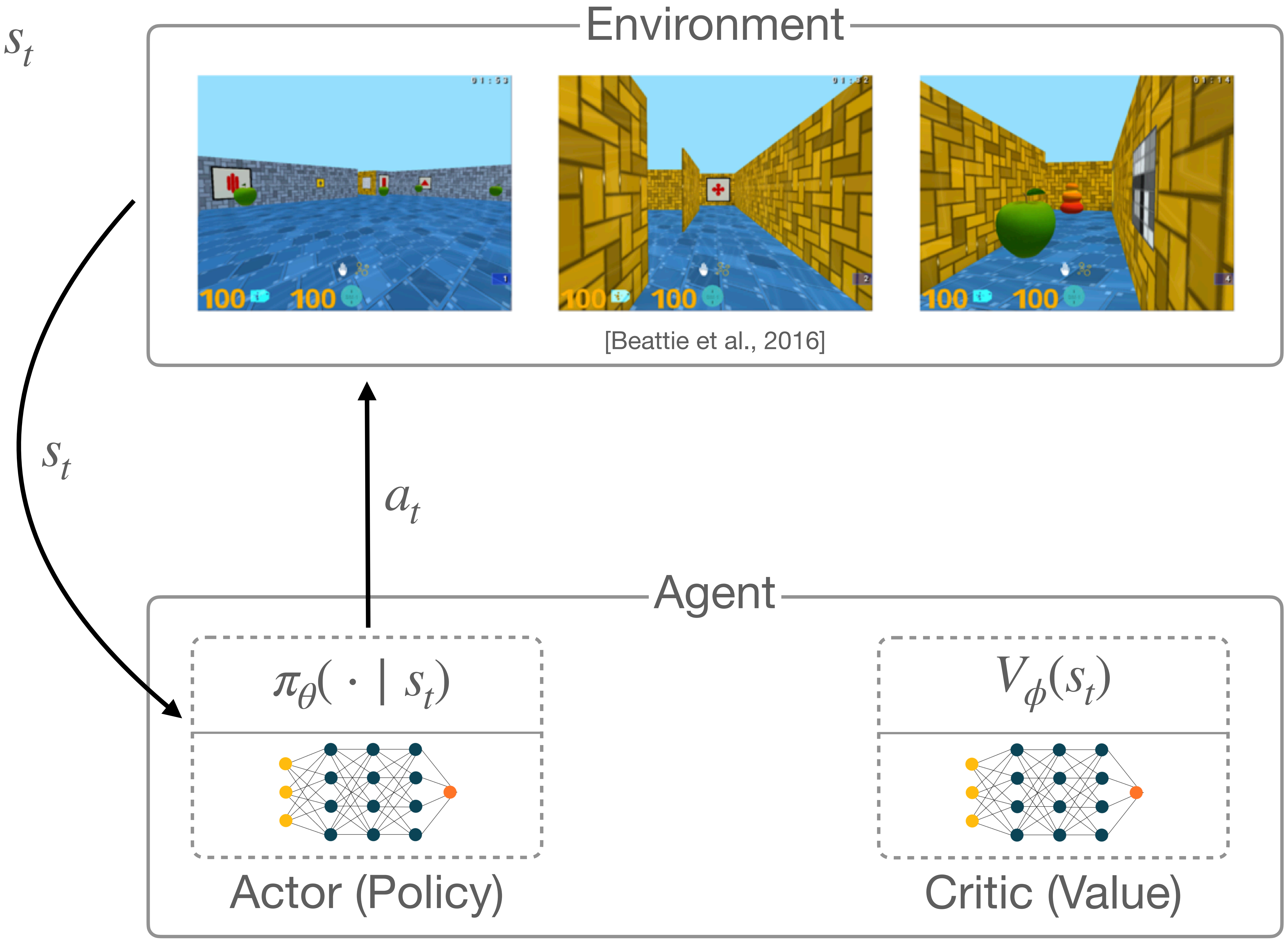
Motivation | Multi-Task RL



Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$



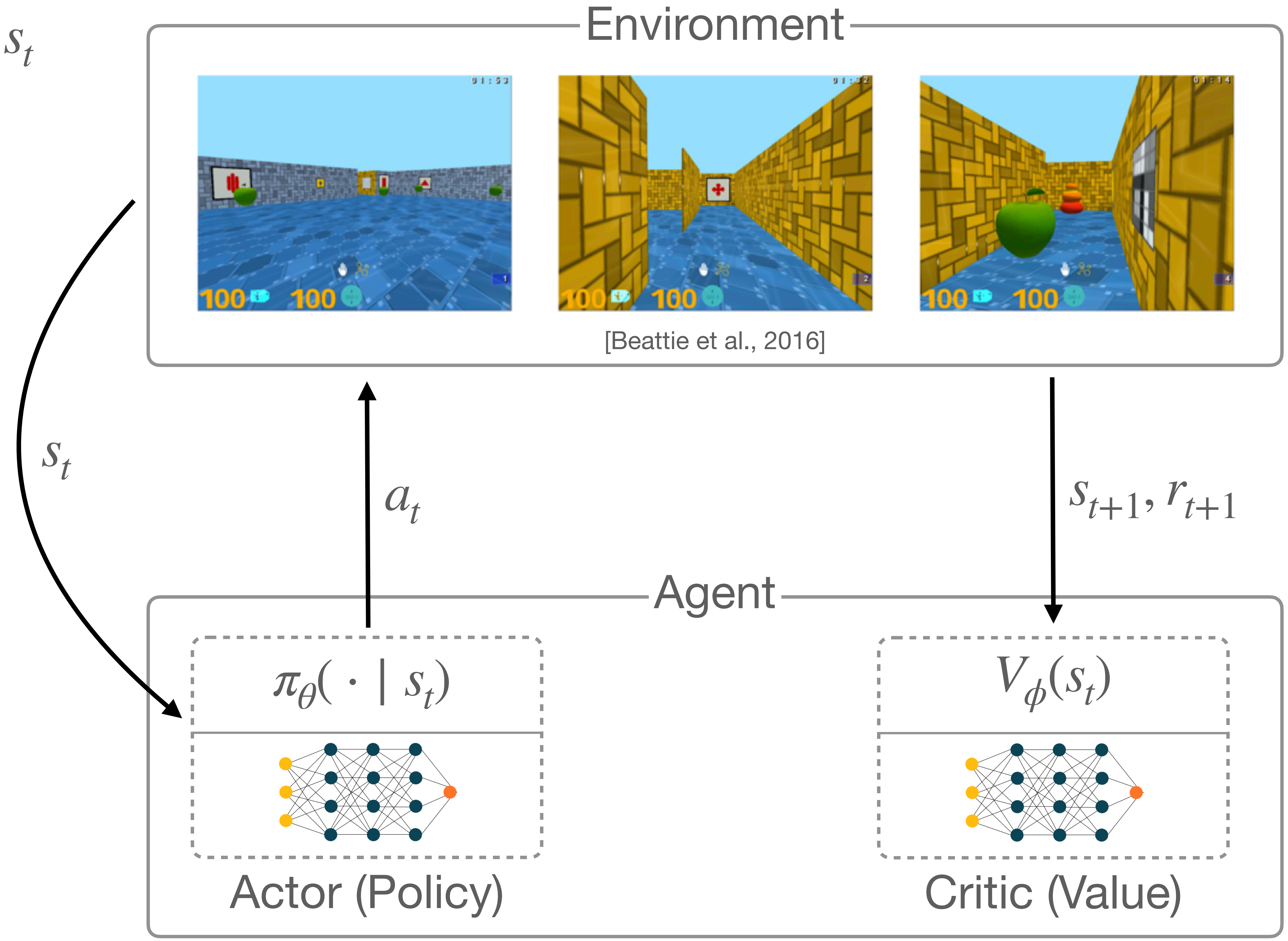
Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$



Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

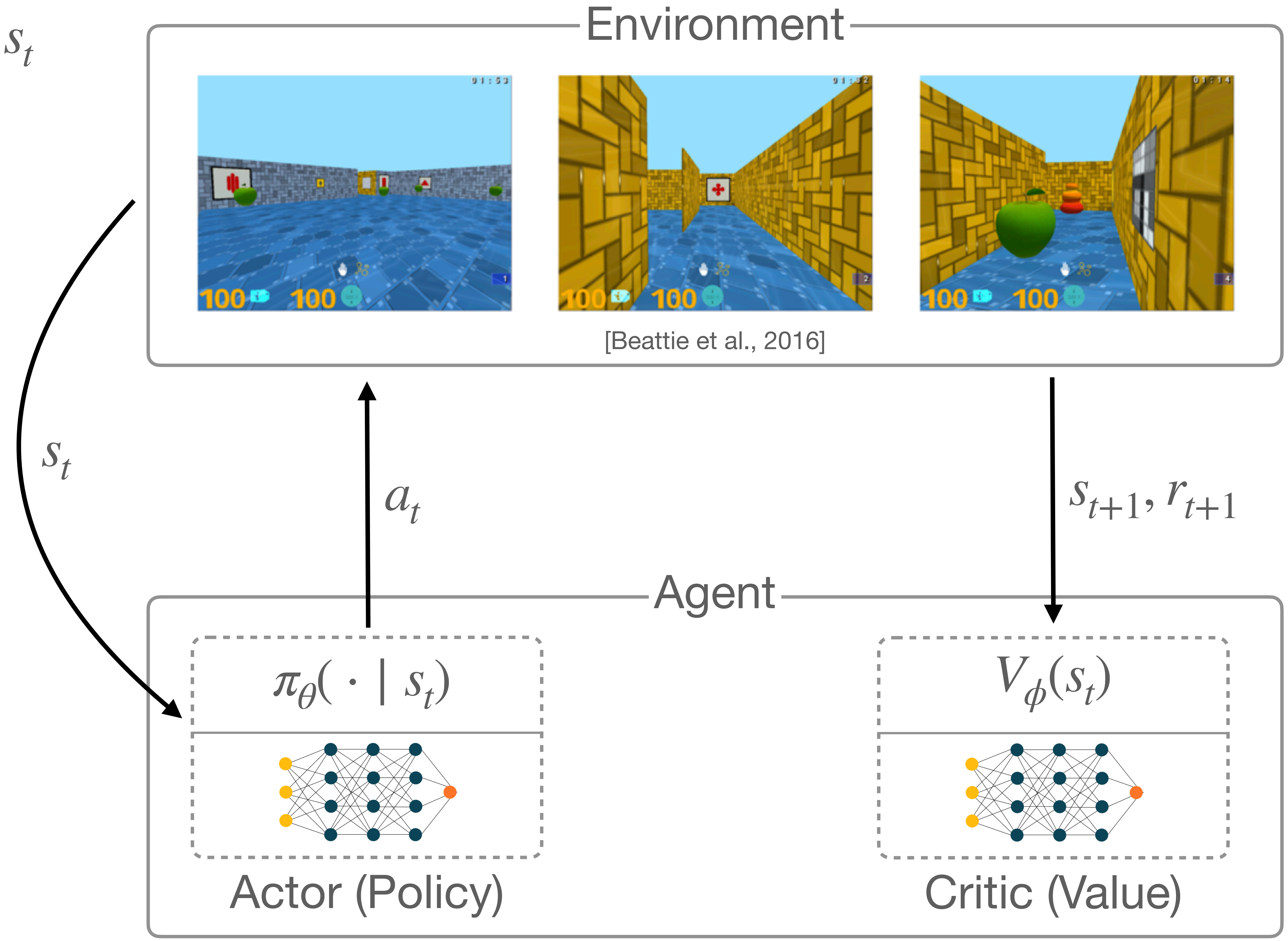
$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$



Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

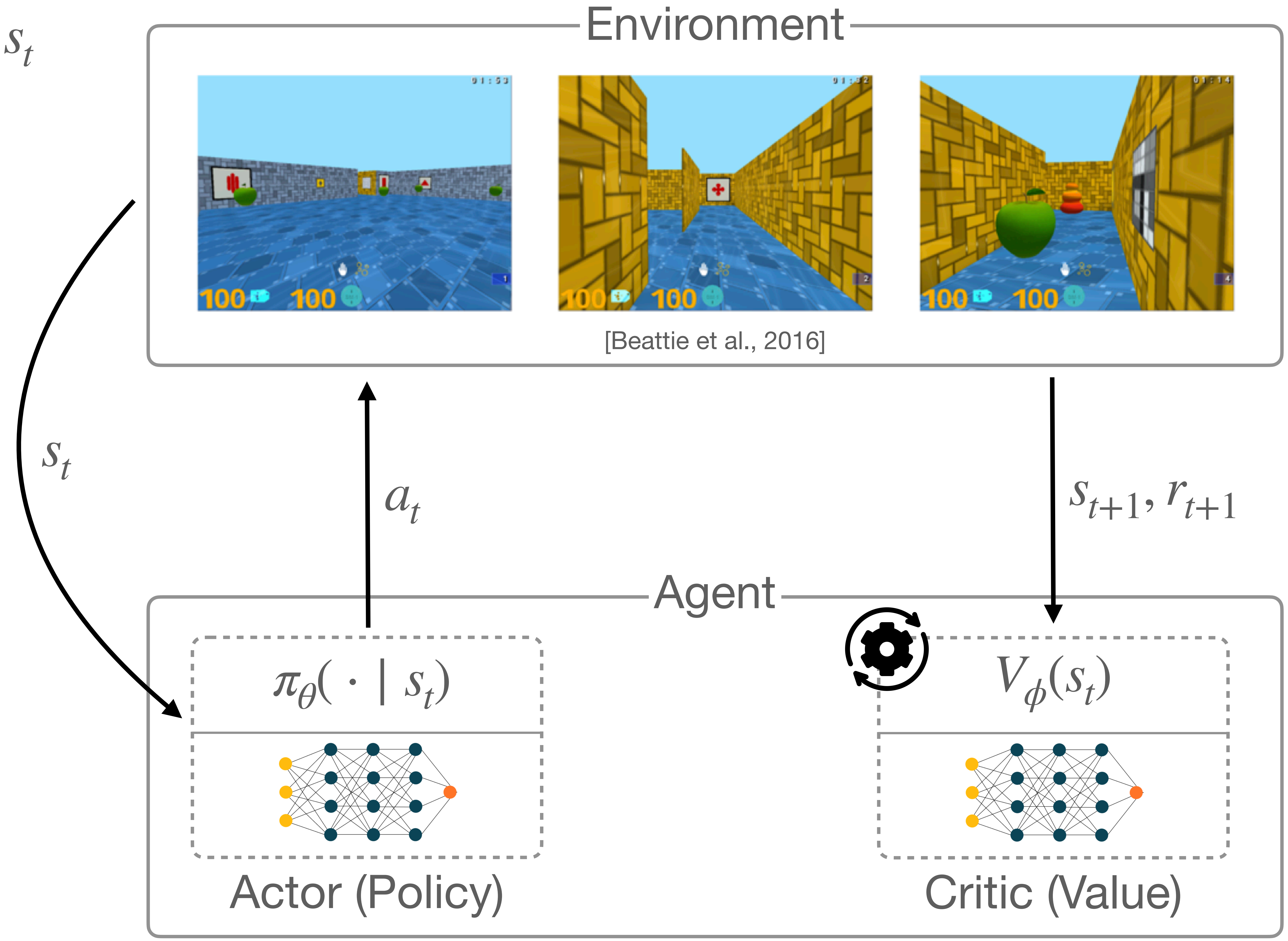
$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$



Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

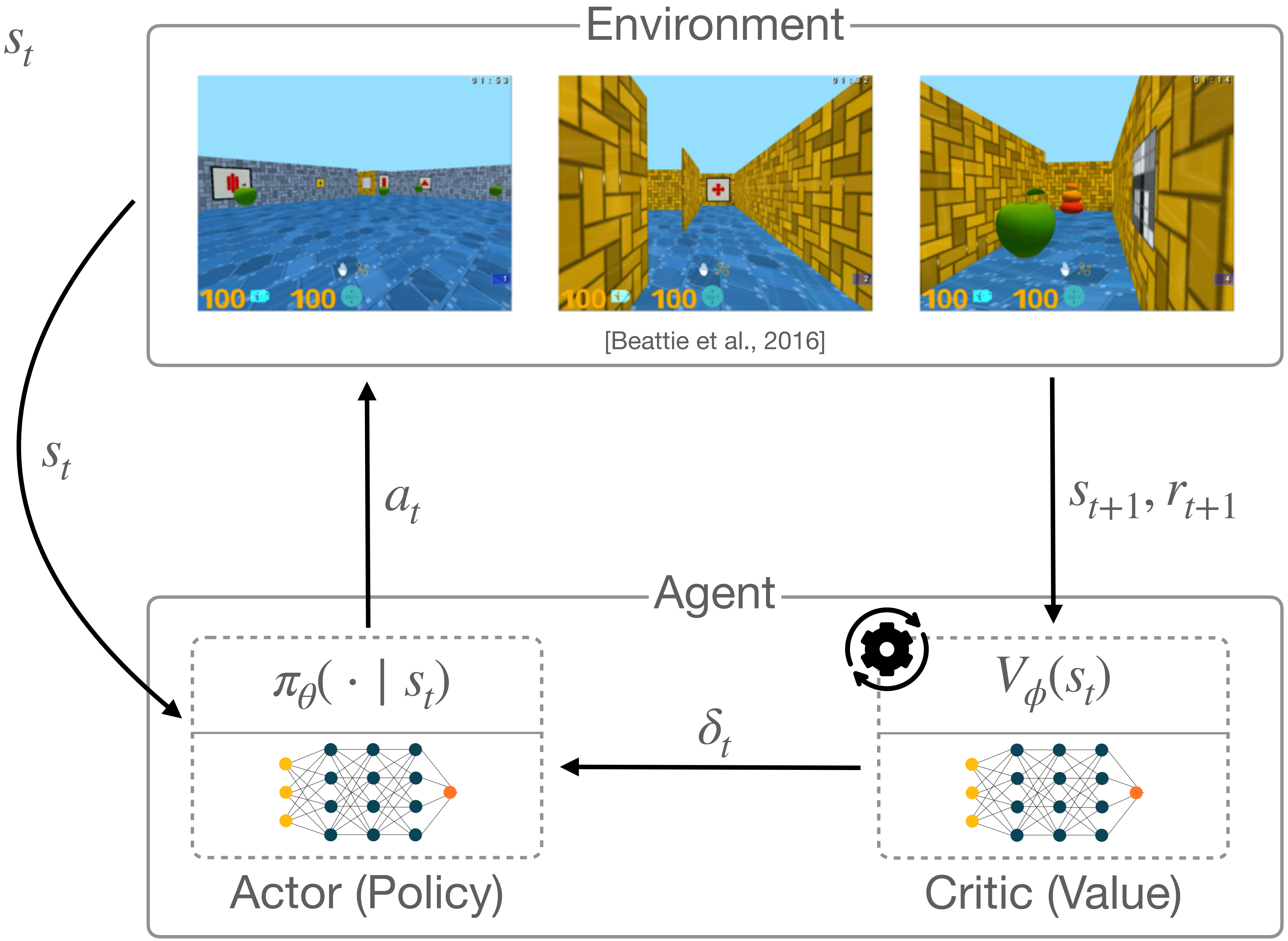
$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$



Motivation | Multi-Task RL

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

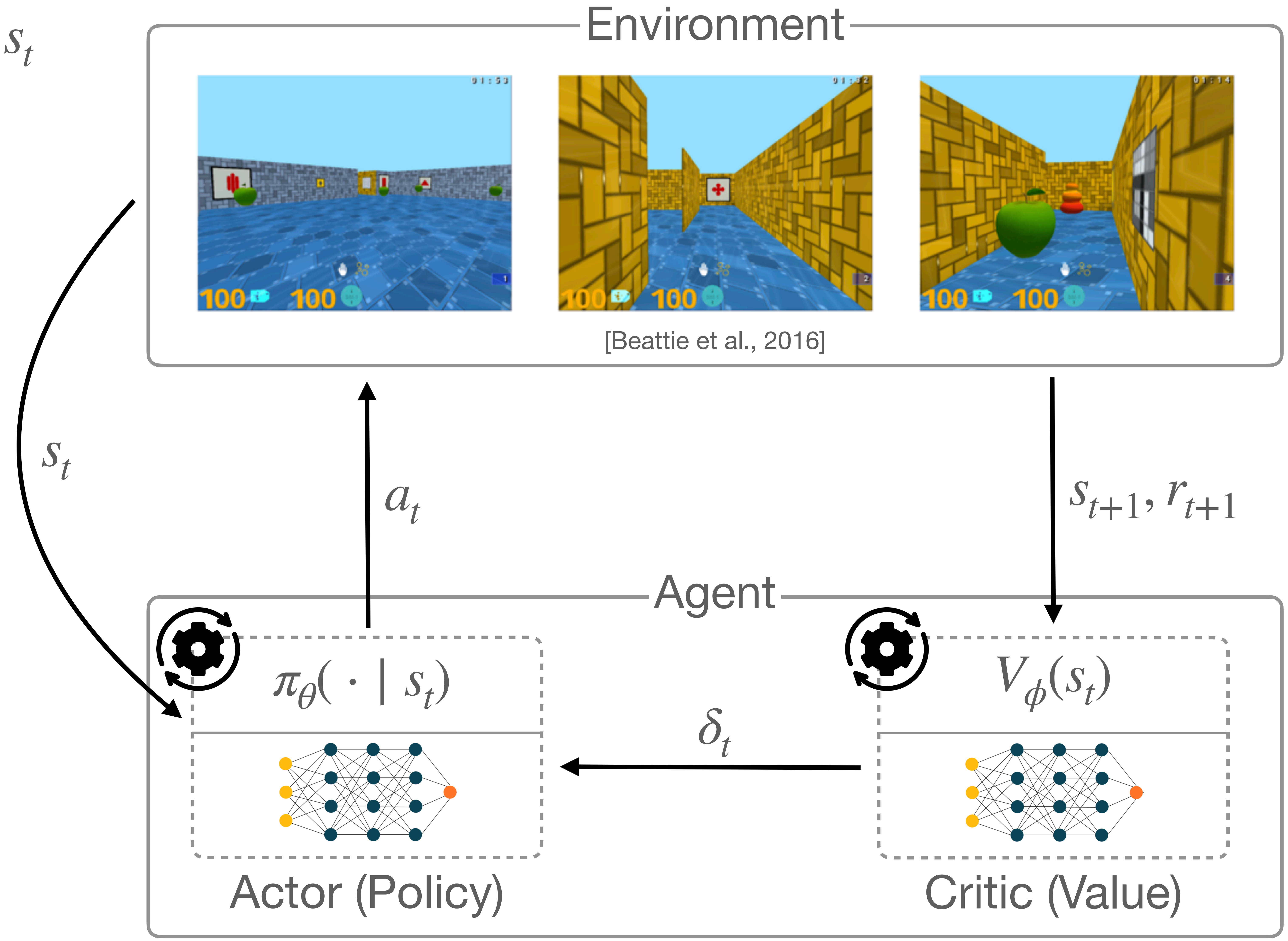
$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Update Value parameters ϕ

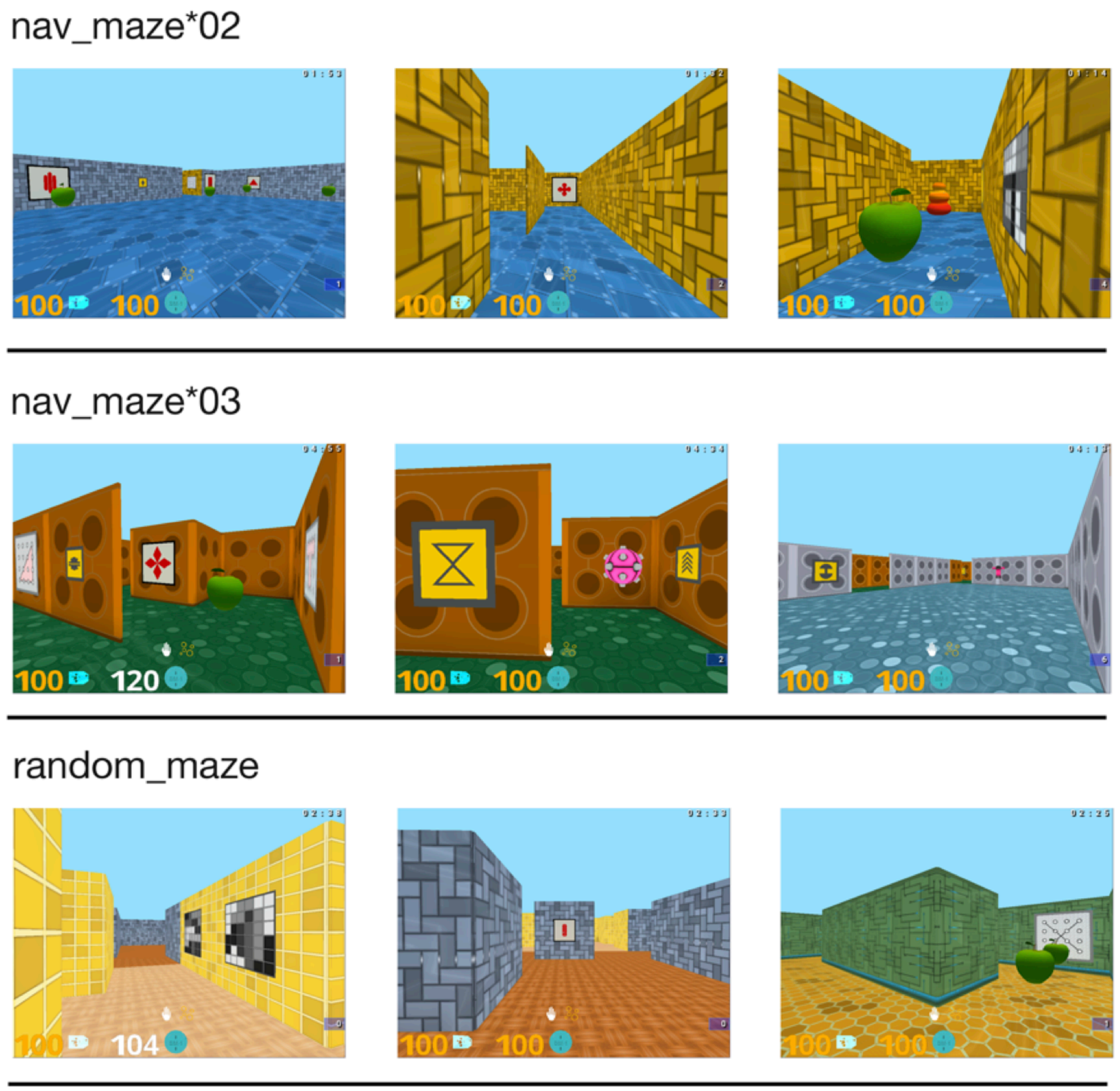
$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

5. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

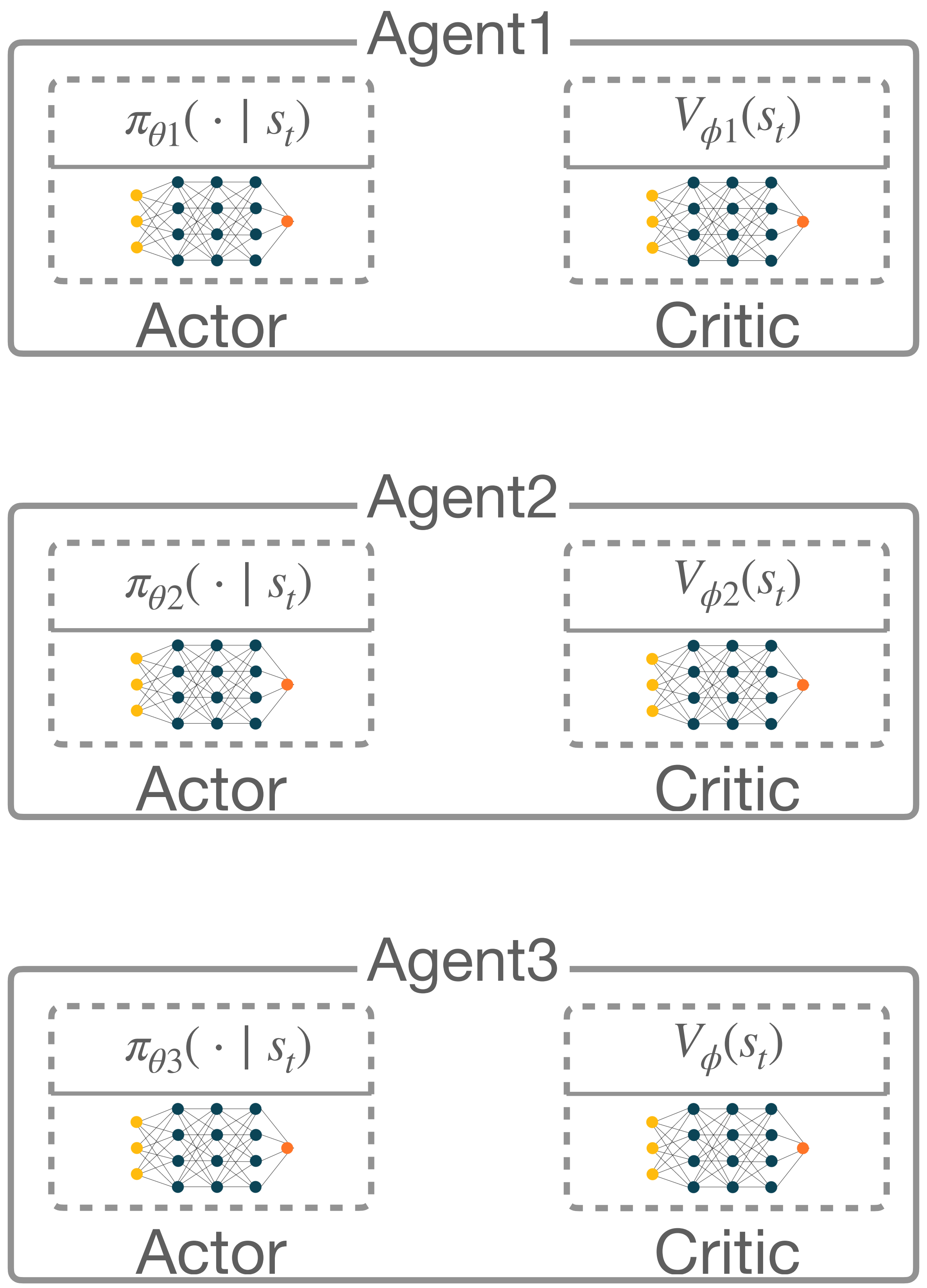


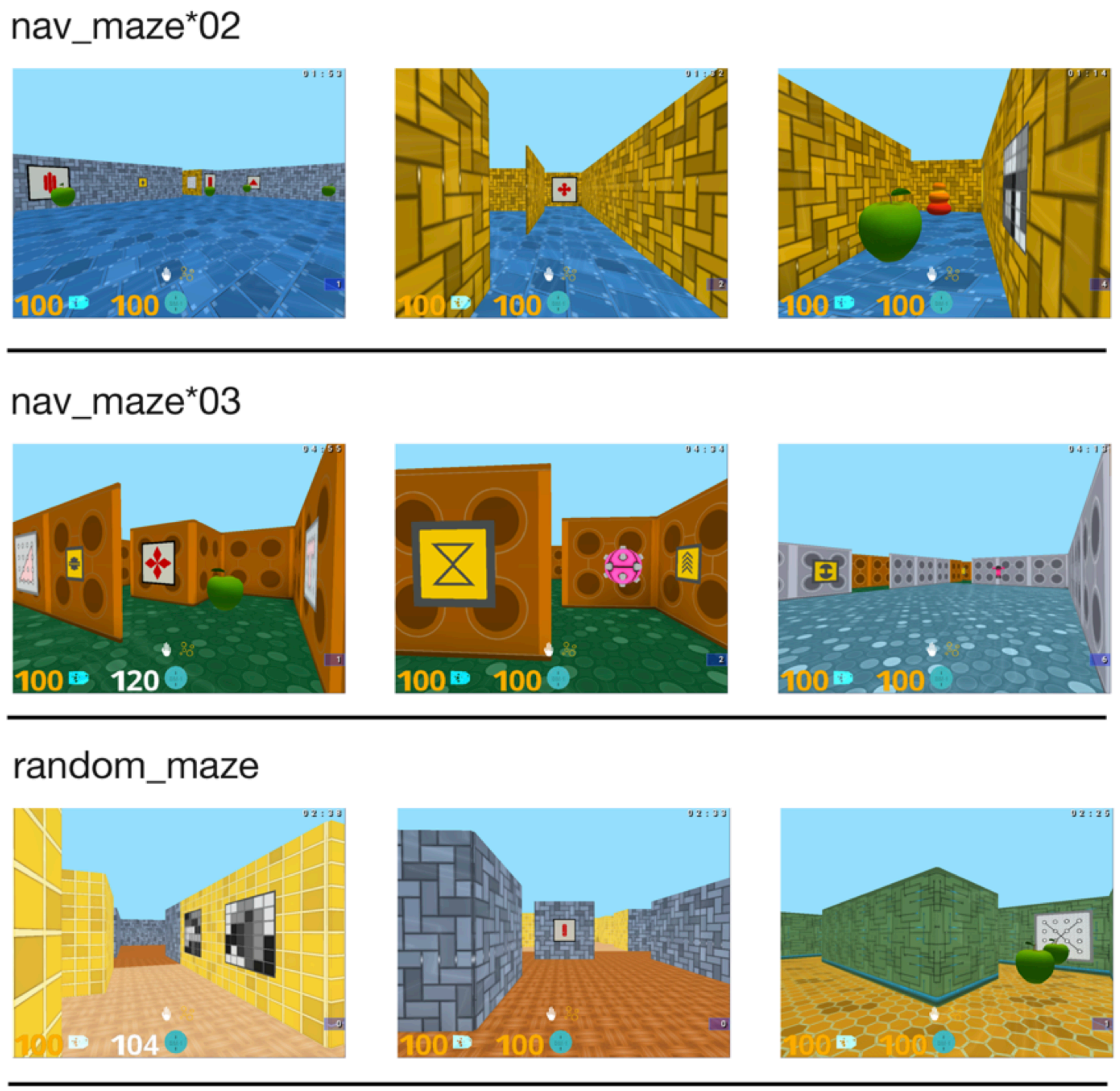
Motivation | Multi-Task RL



[Beattie et al., 2016]

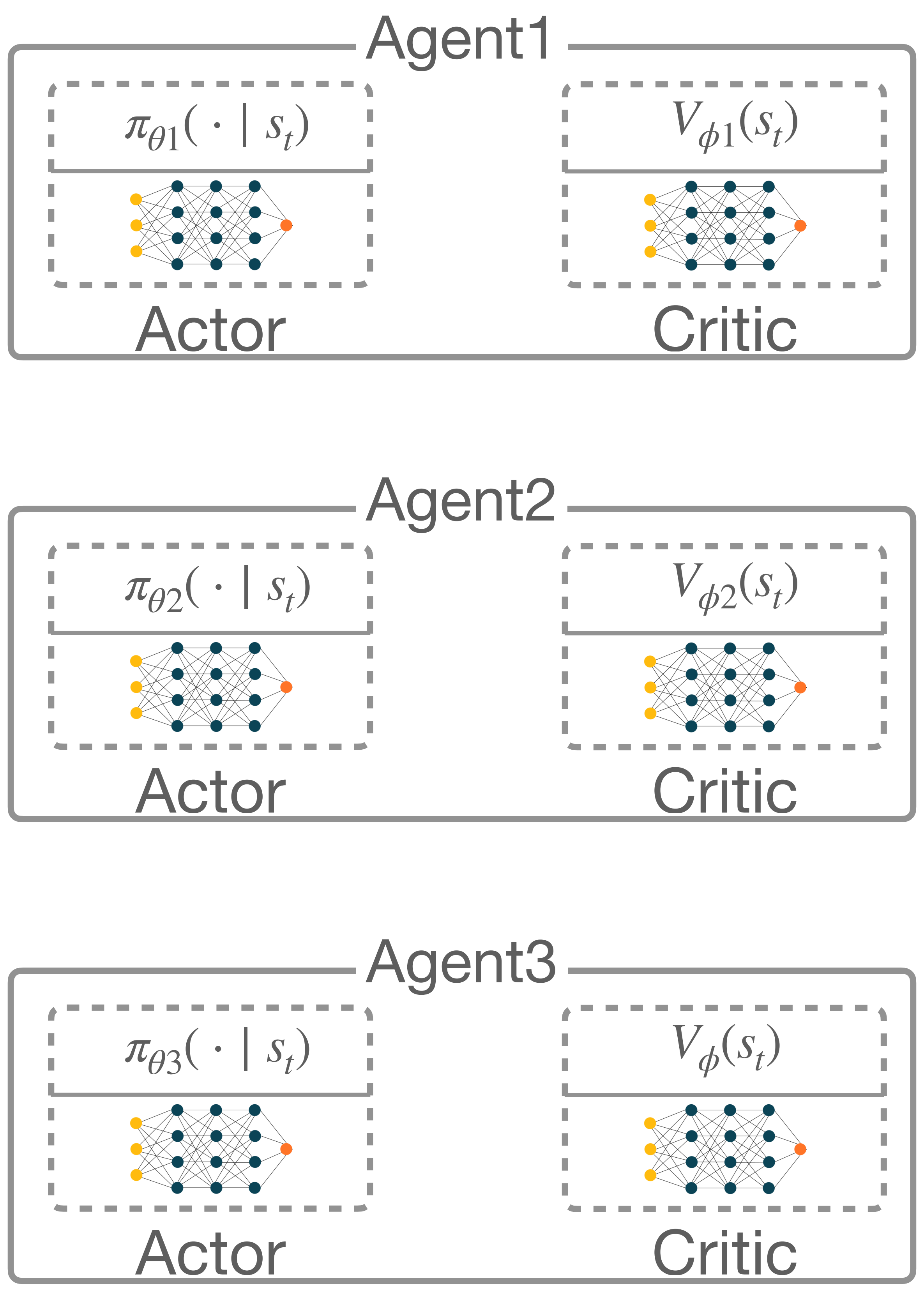
Motivation | Multi-Task RL

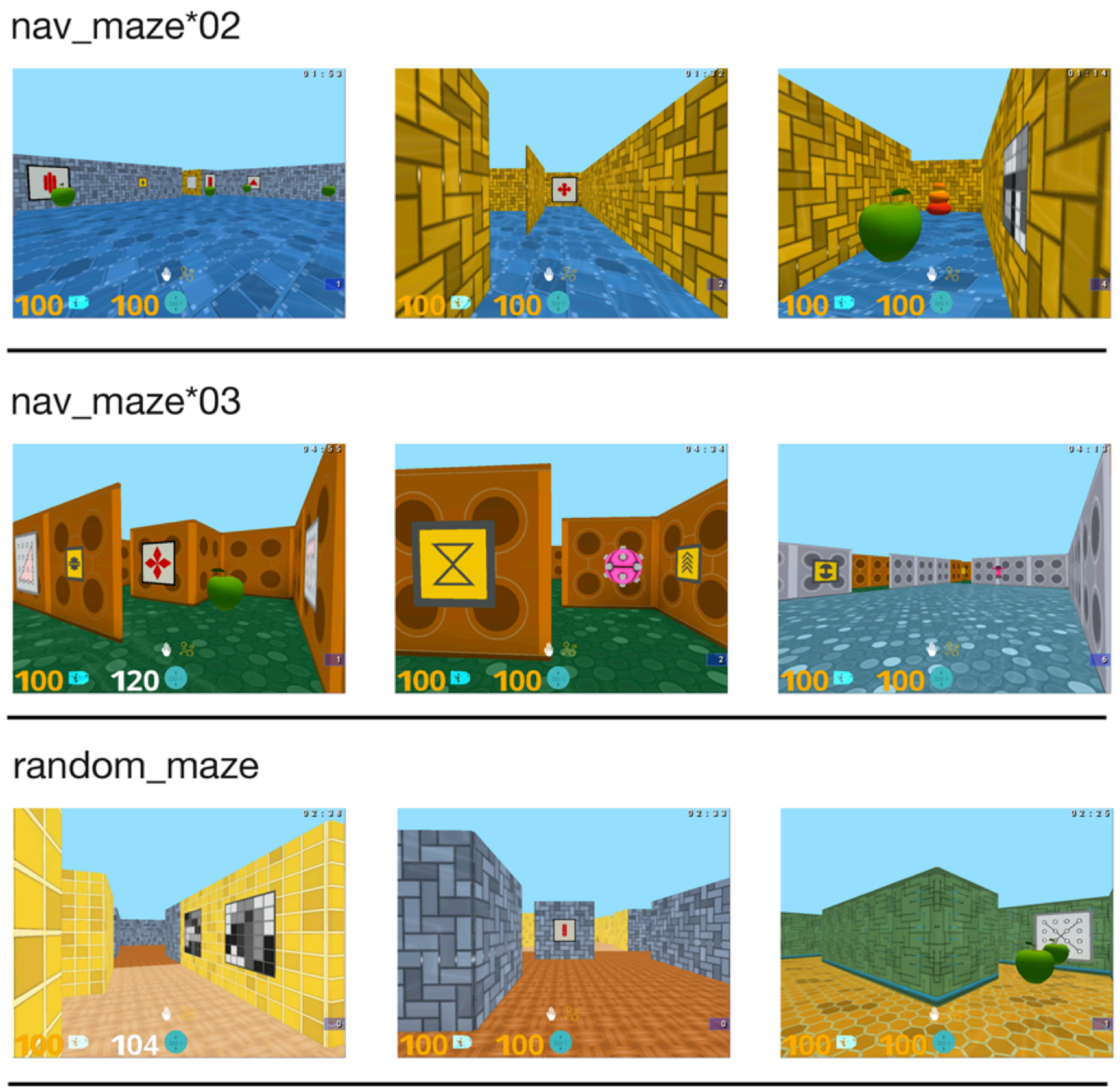




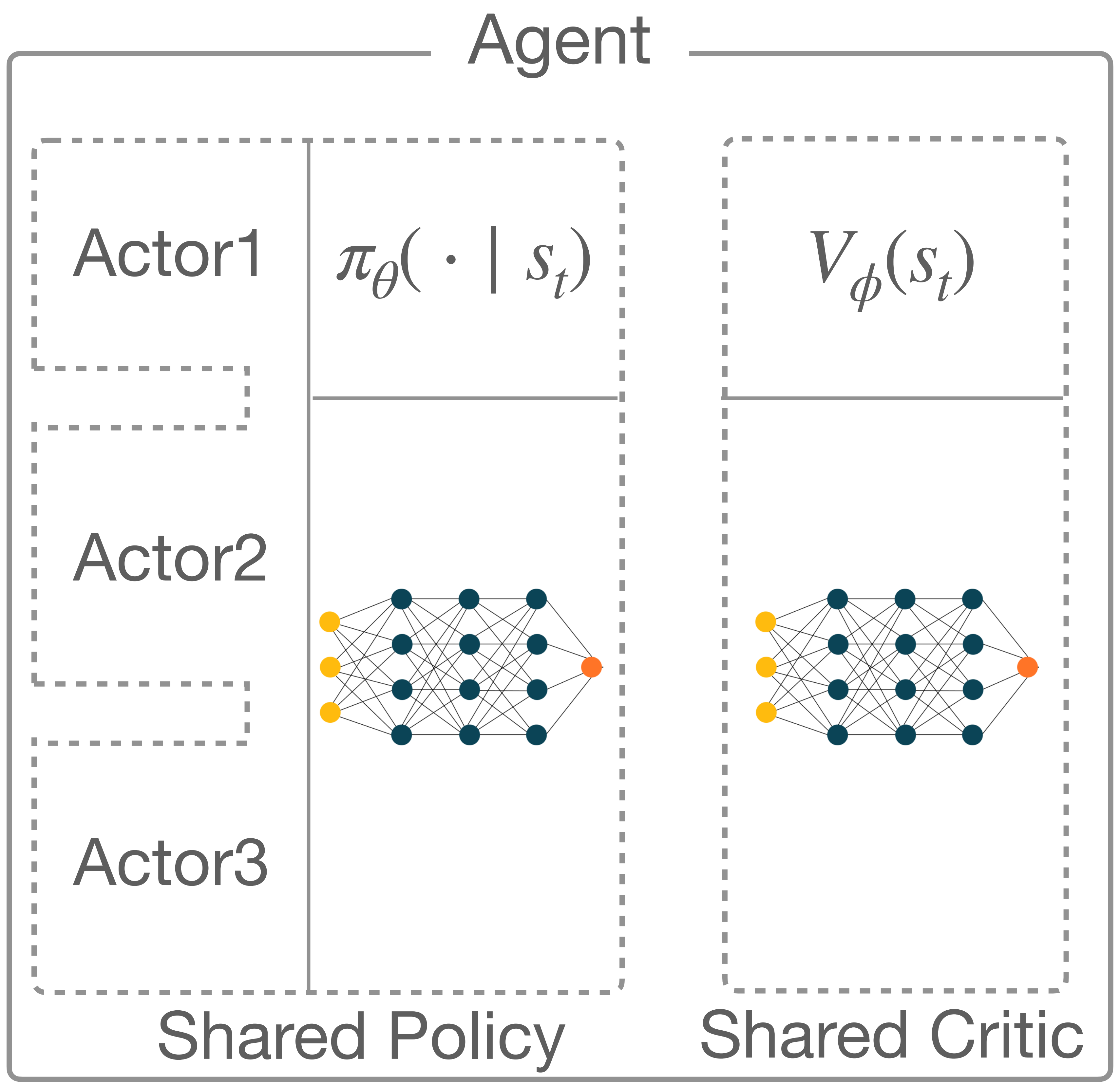
[Beattie et al., 2016]

Motivation | Multi-Task RL

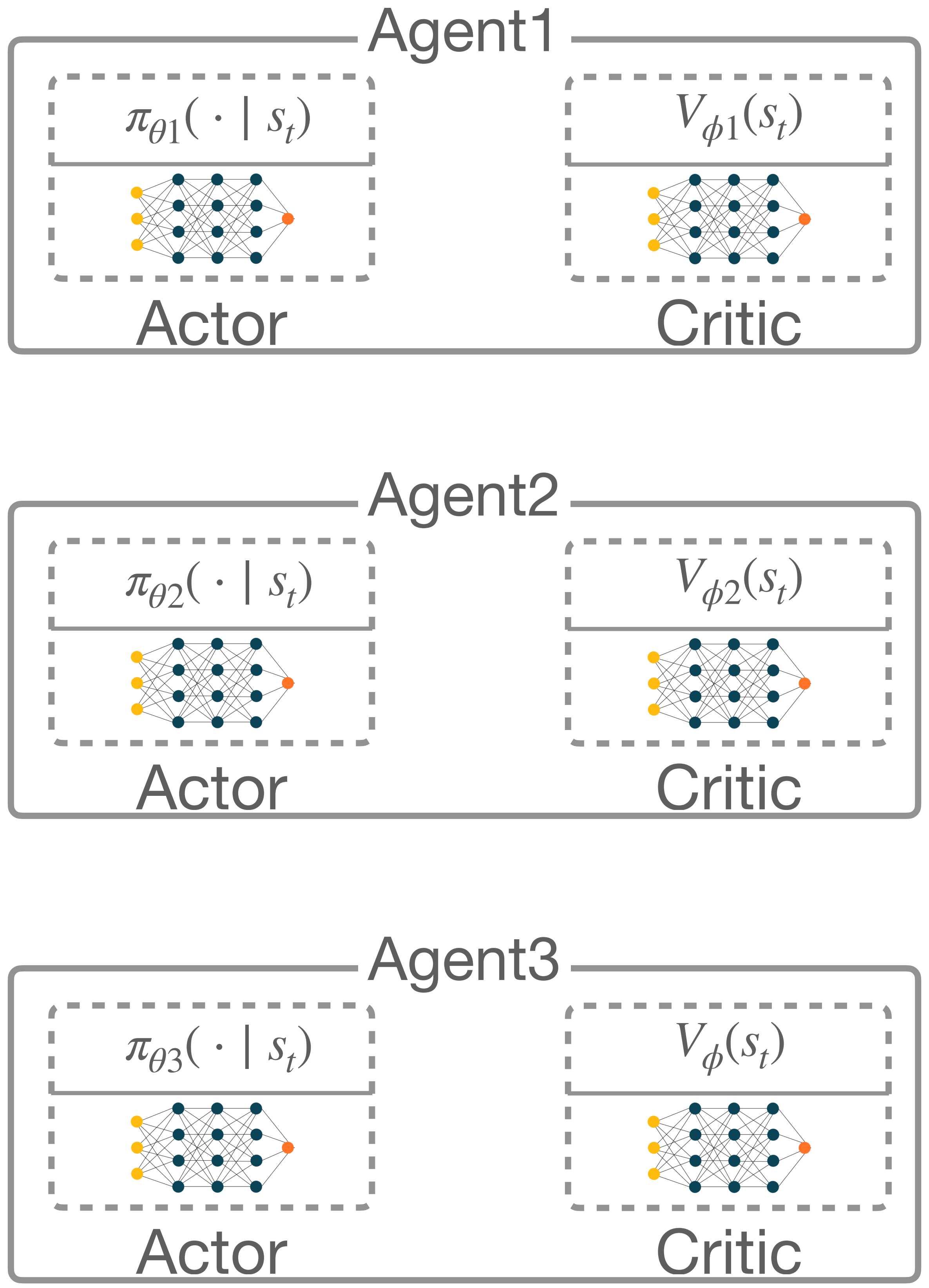


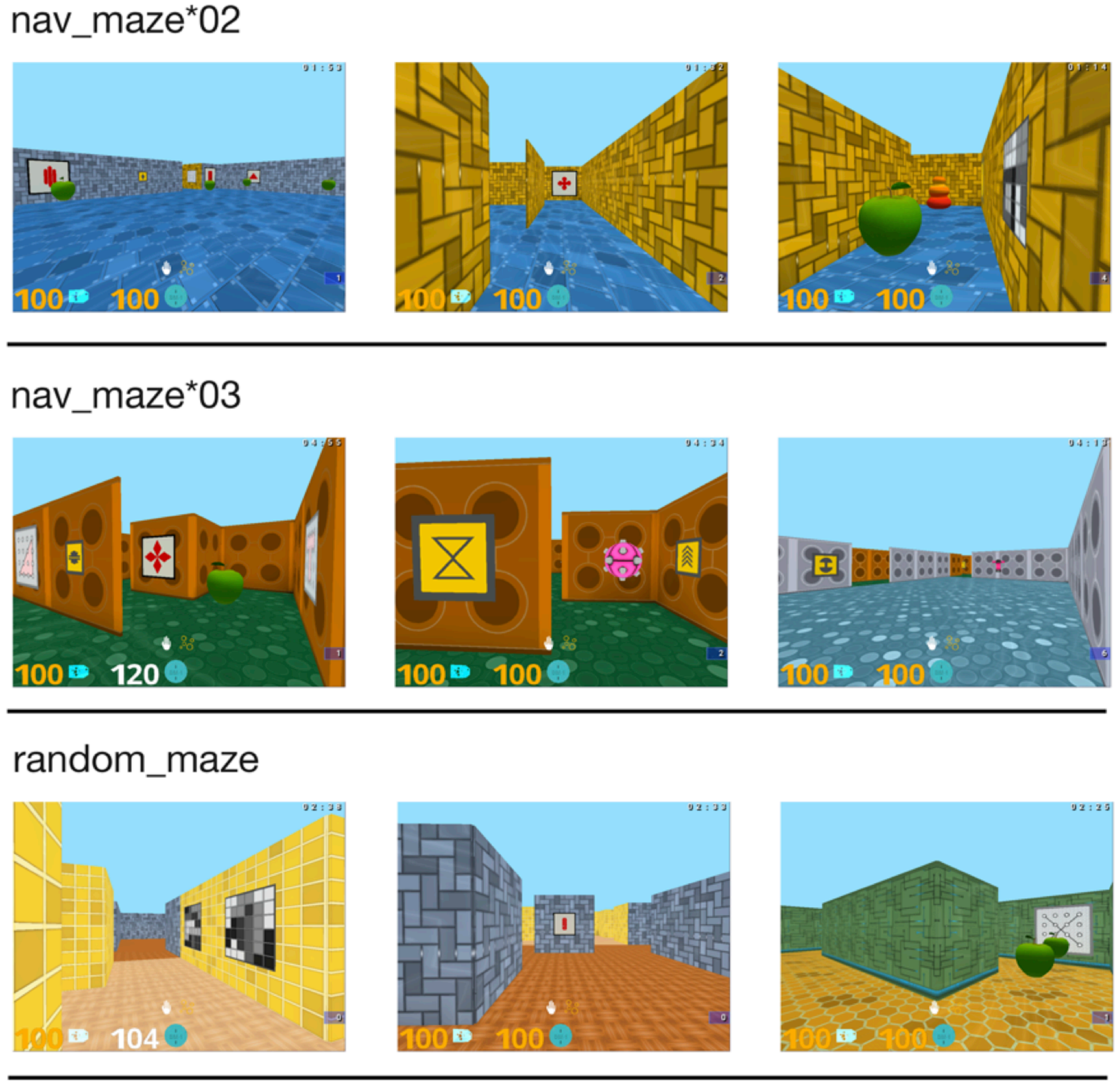


[Beattie et al., 2016]

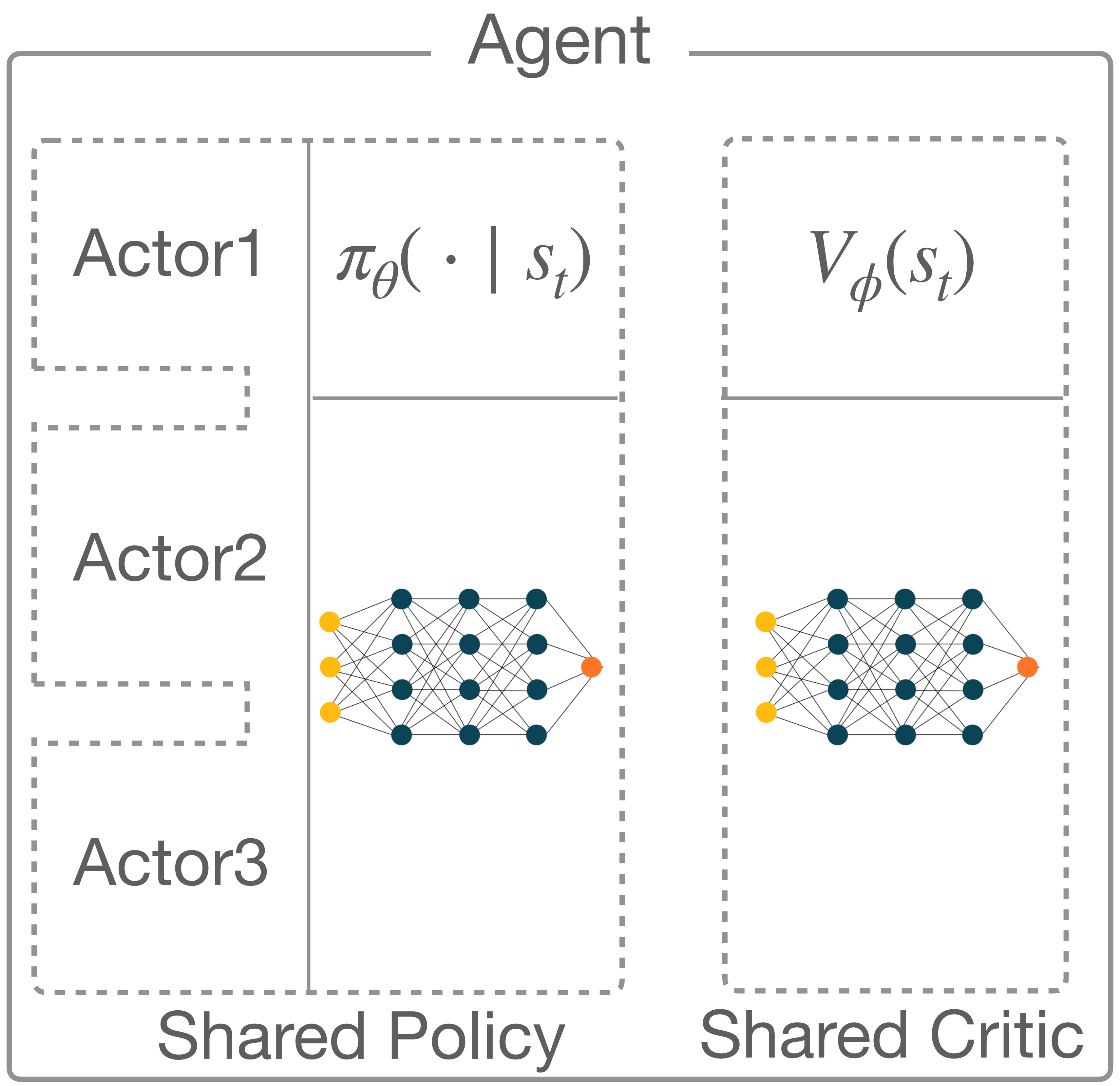


Motivation | Multi-Task RL



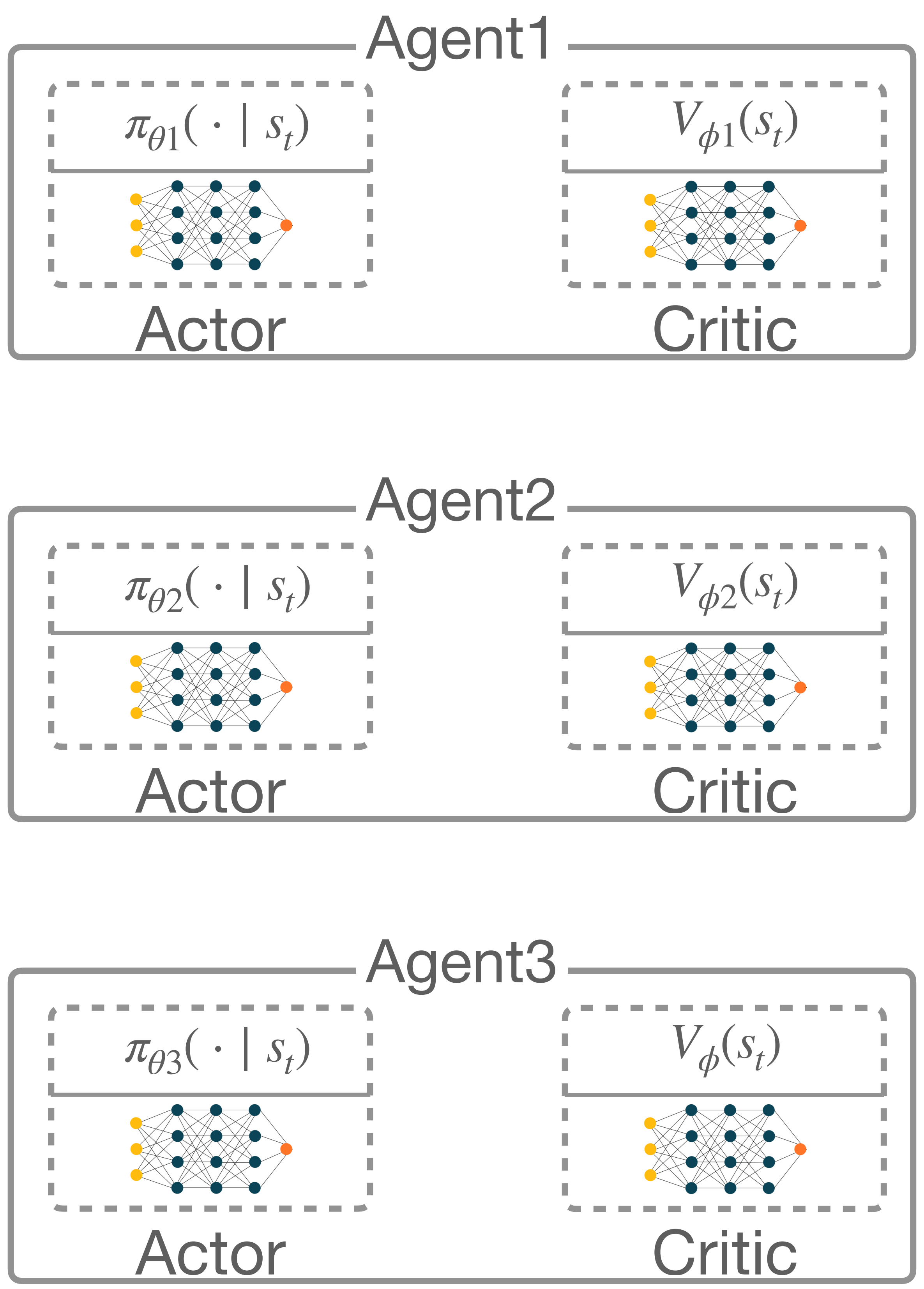


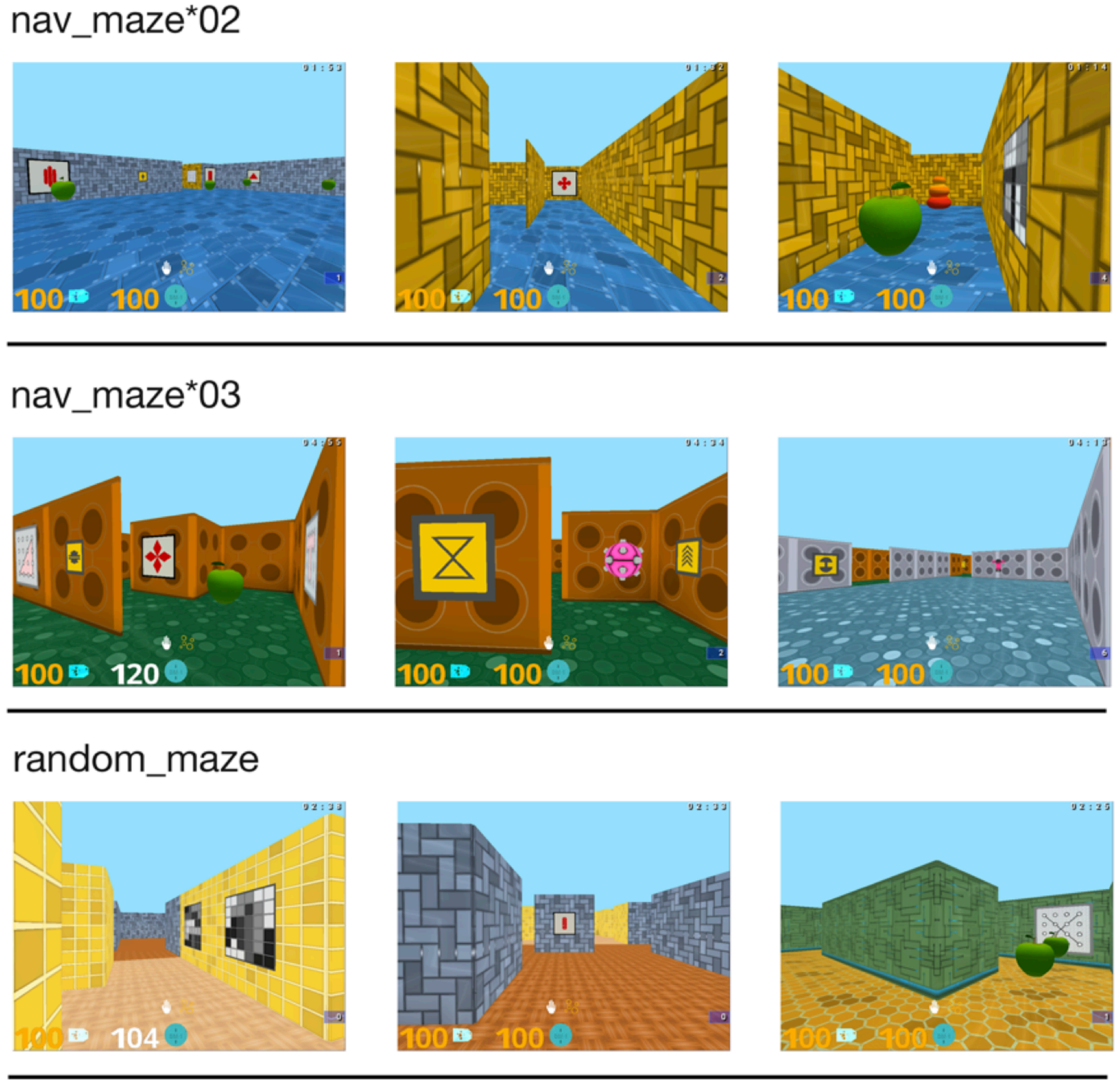
[Beattie et al., 2016]



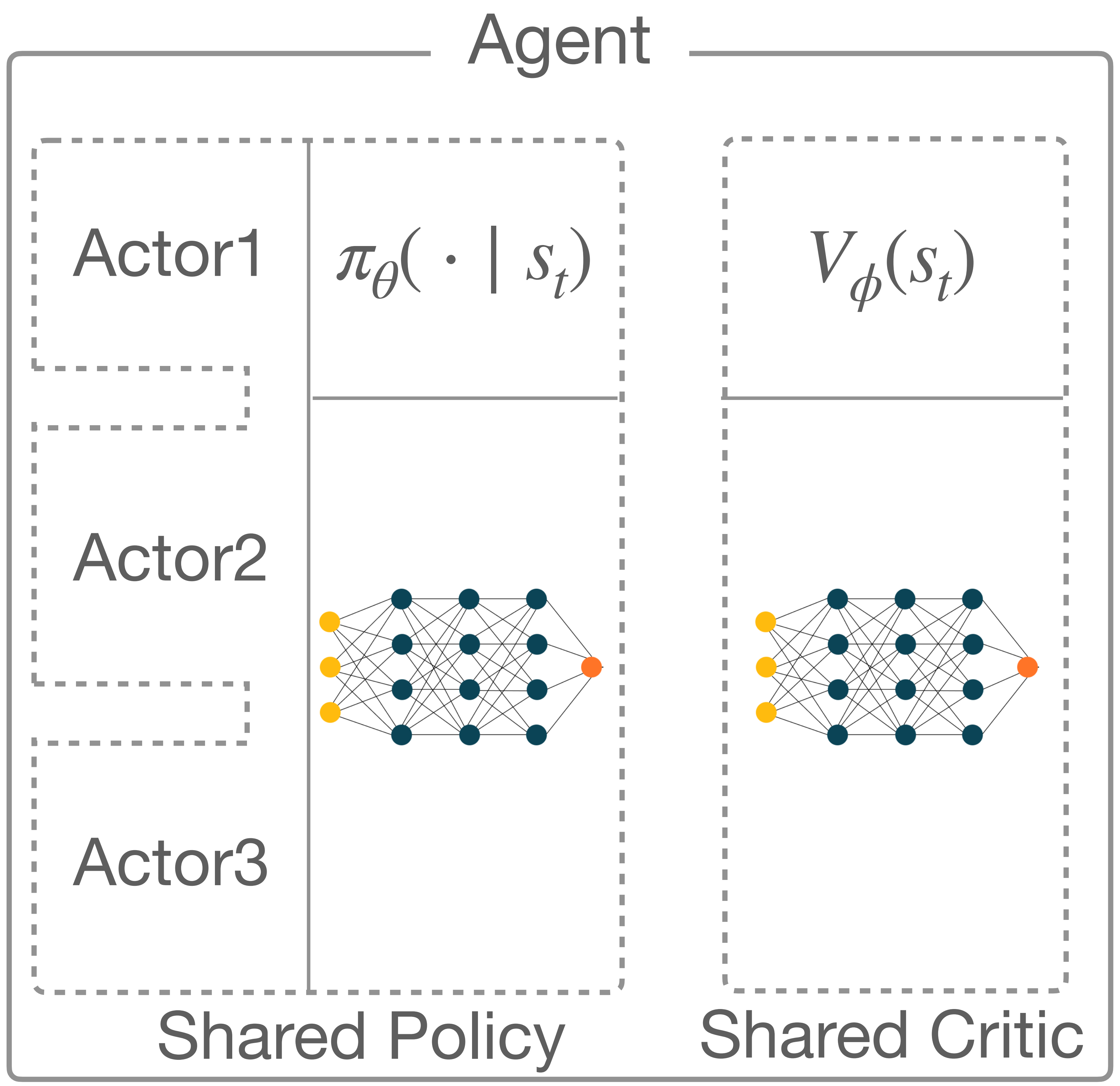
✓ Positive transfer of learned features across tasks

Motivation | Multi-Task RL



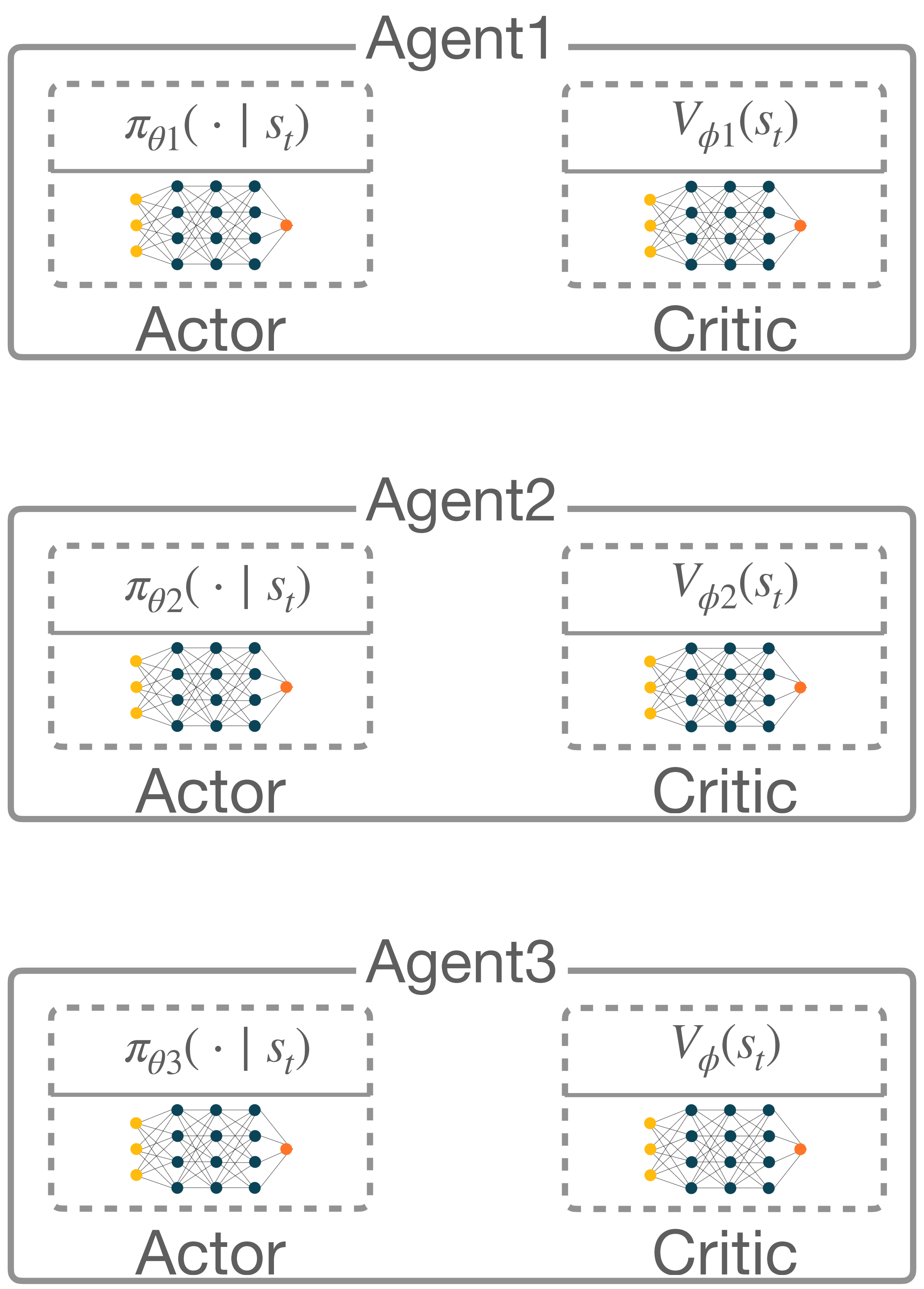


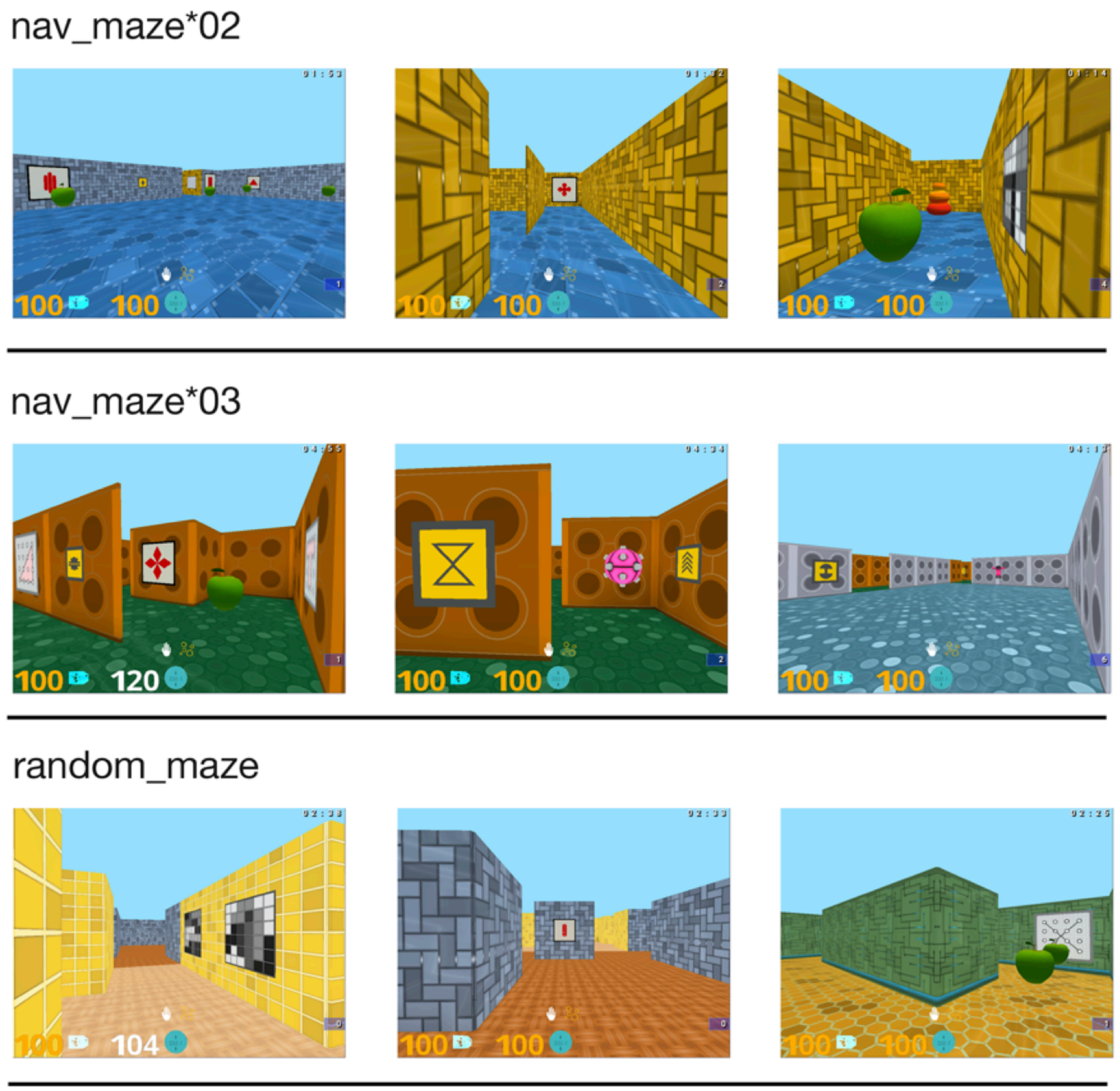
[Beattie et al., 2016]



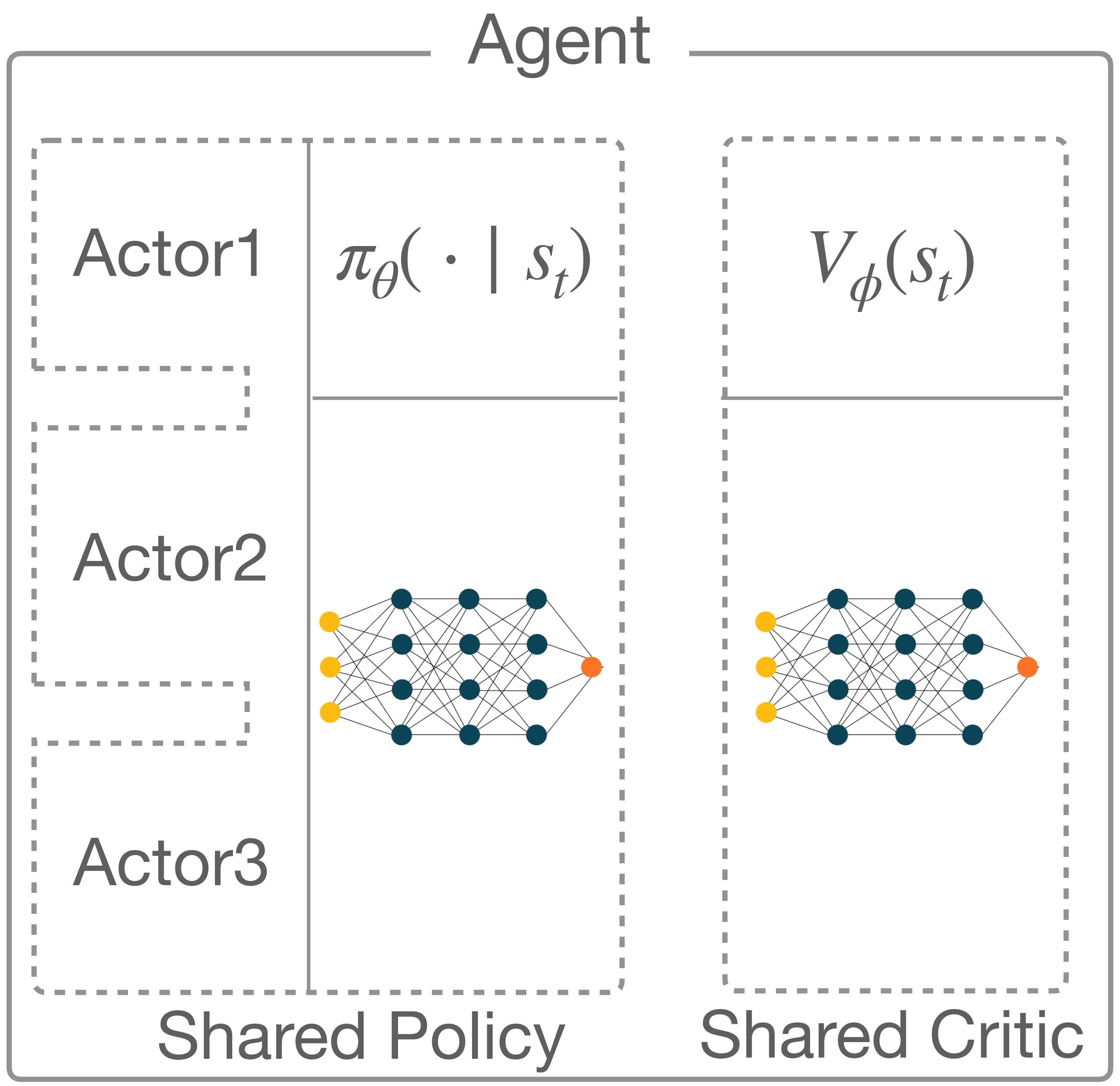
- ✓ Positive transfer of learned features across tasks
- ✓ Better compute efficiency

Motivation | Multi-Task RL



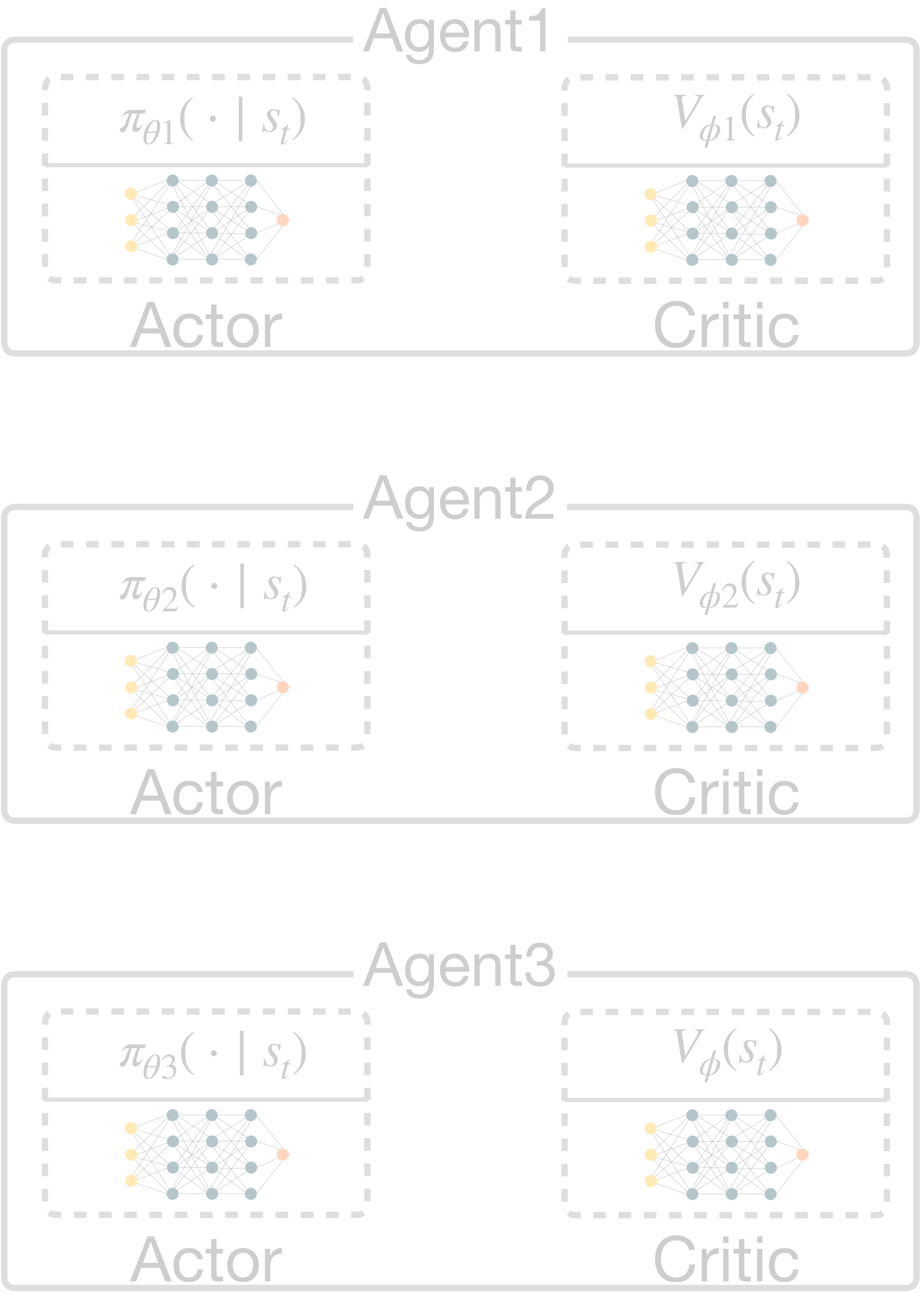


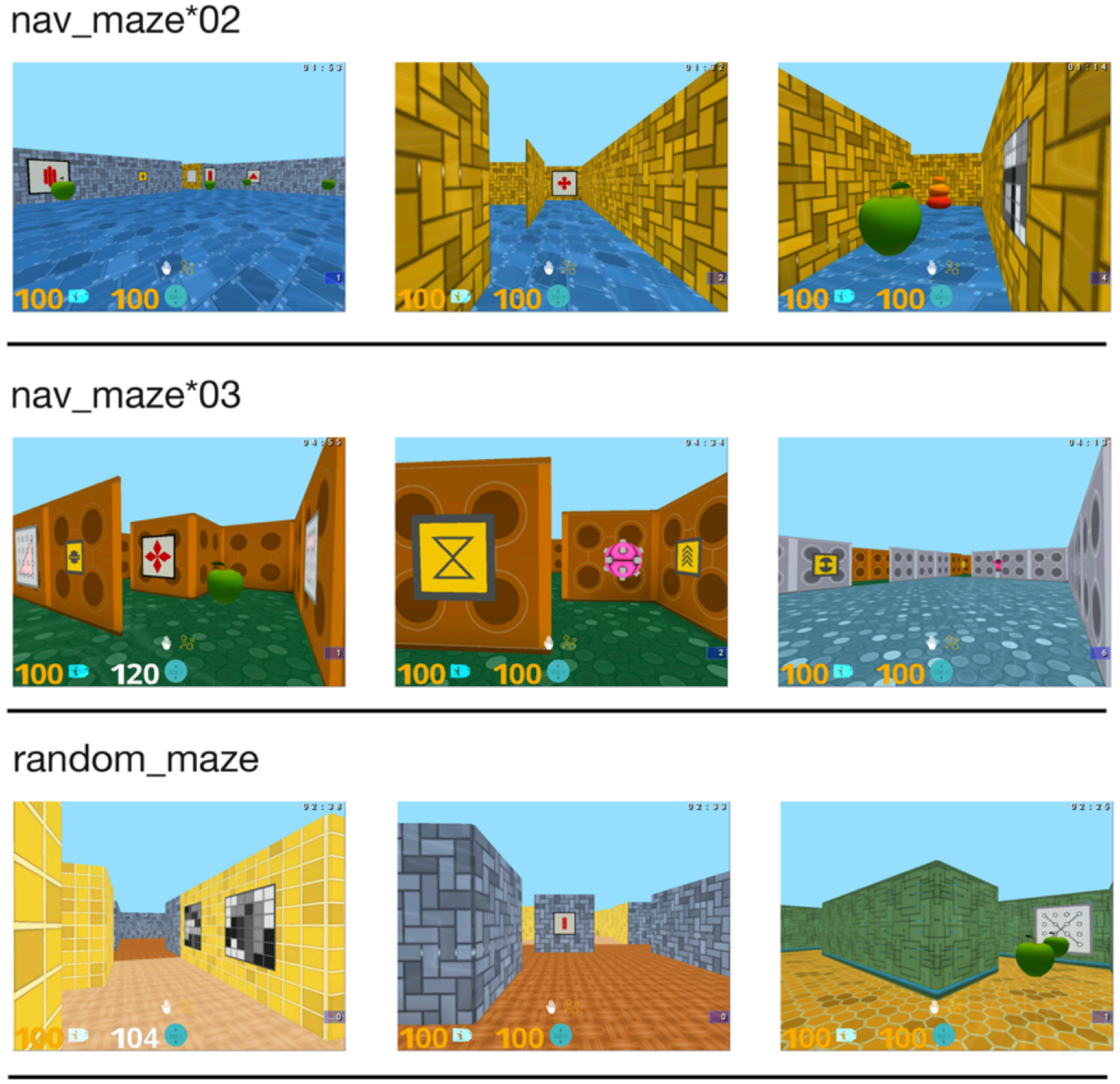
[Beattie et al., 2016]



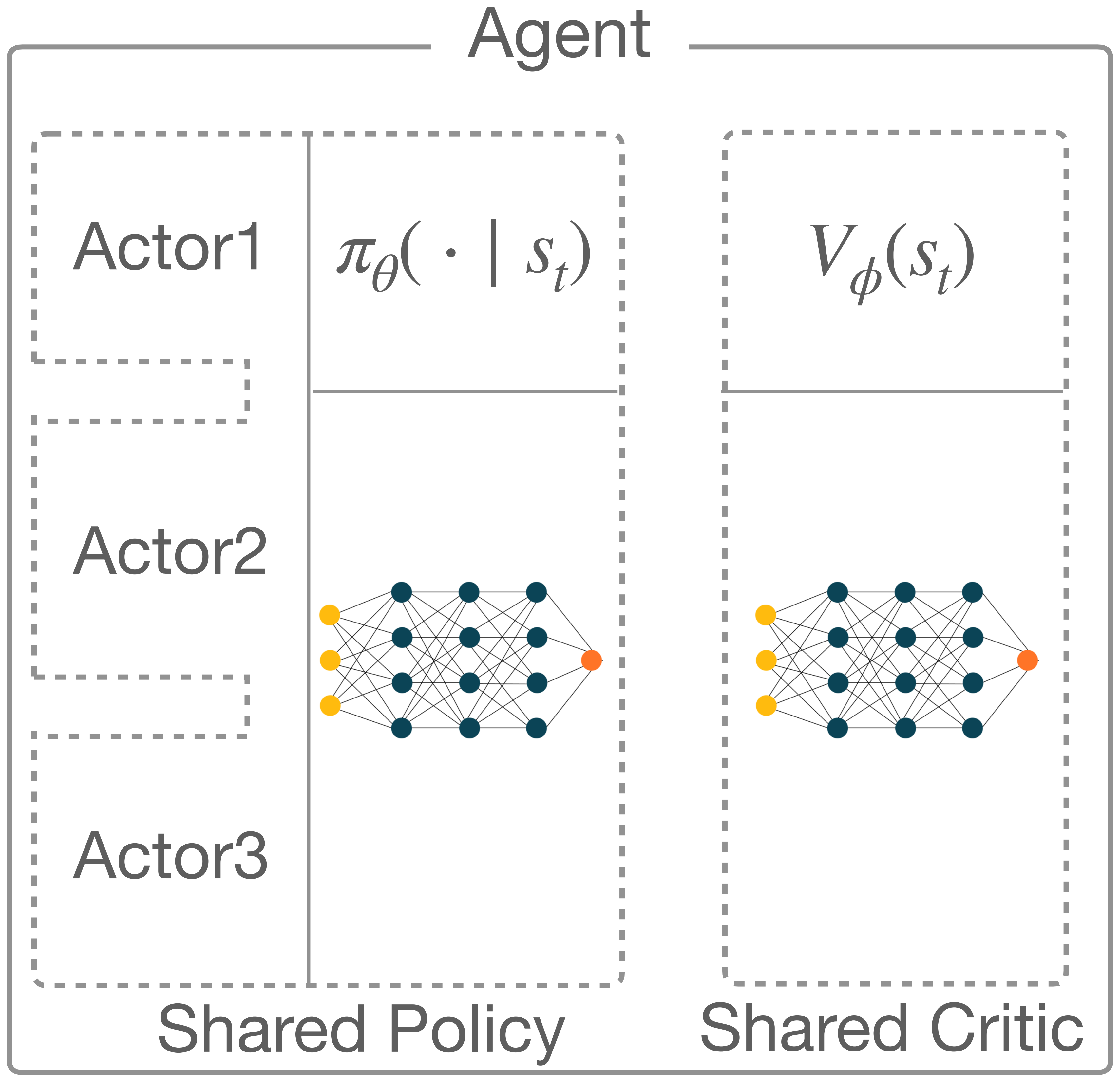
- ✓ Positive transfer of learned features across tasks
- ✓ Better compute efficiency
- ✗ Compatible action and state space

Motivation | Multi-Task RL



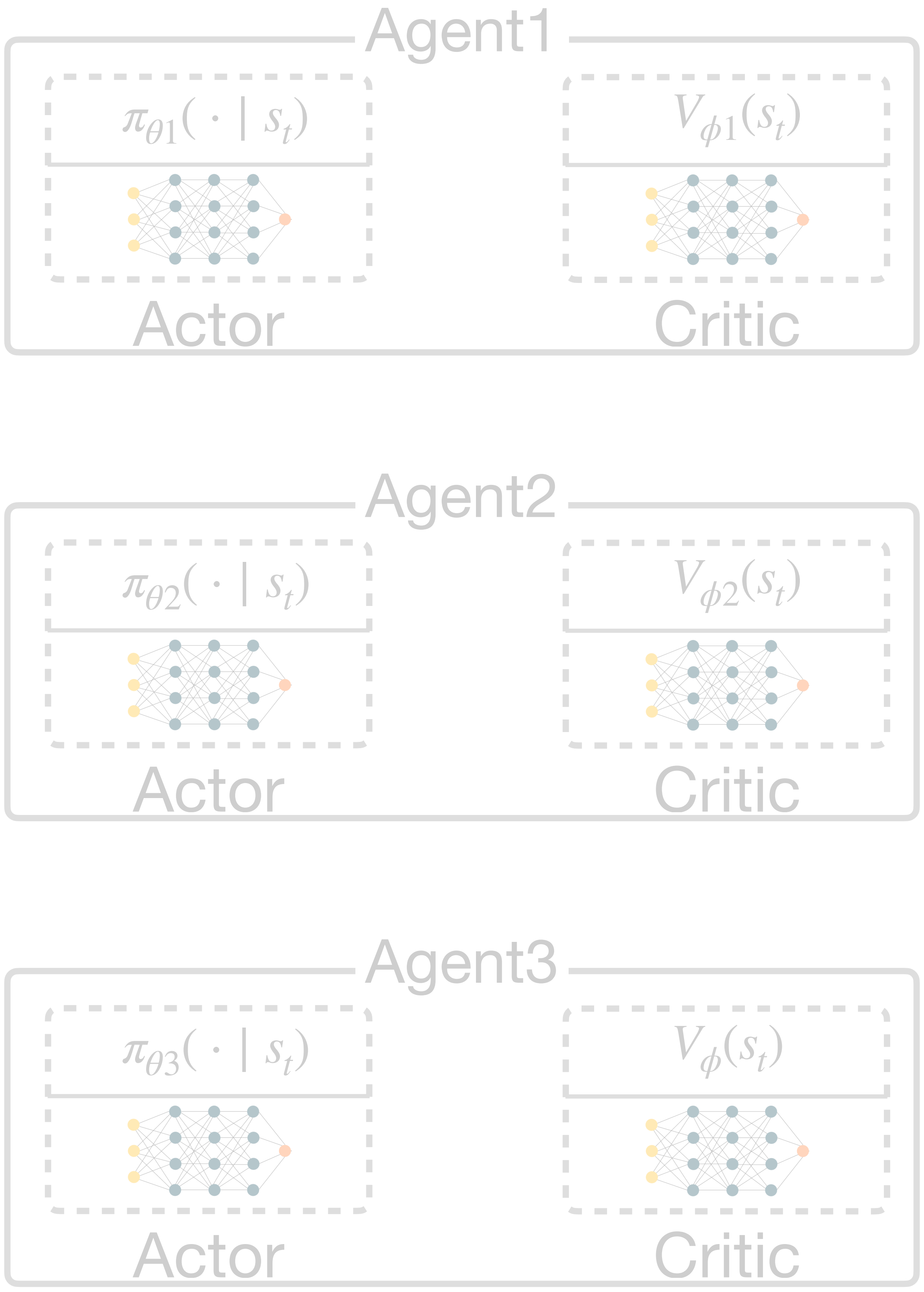


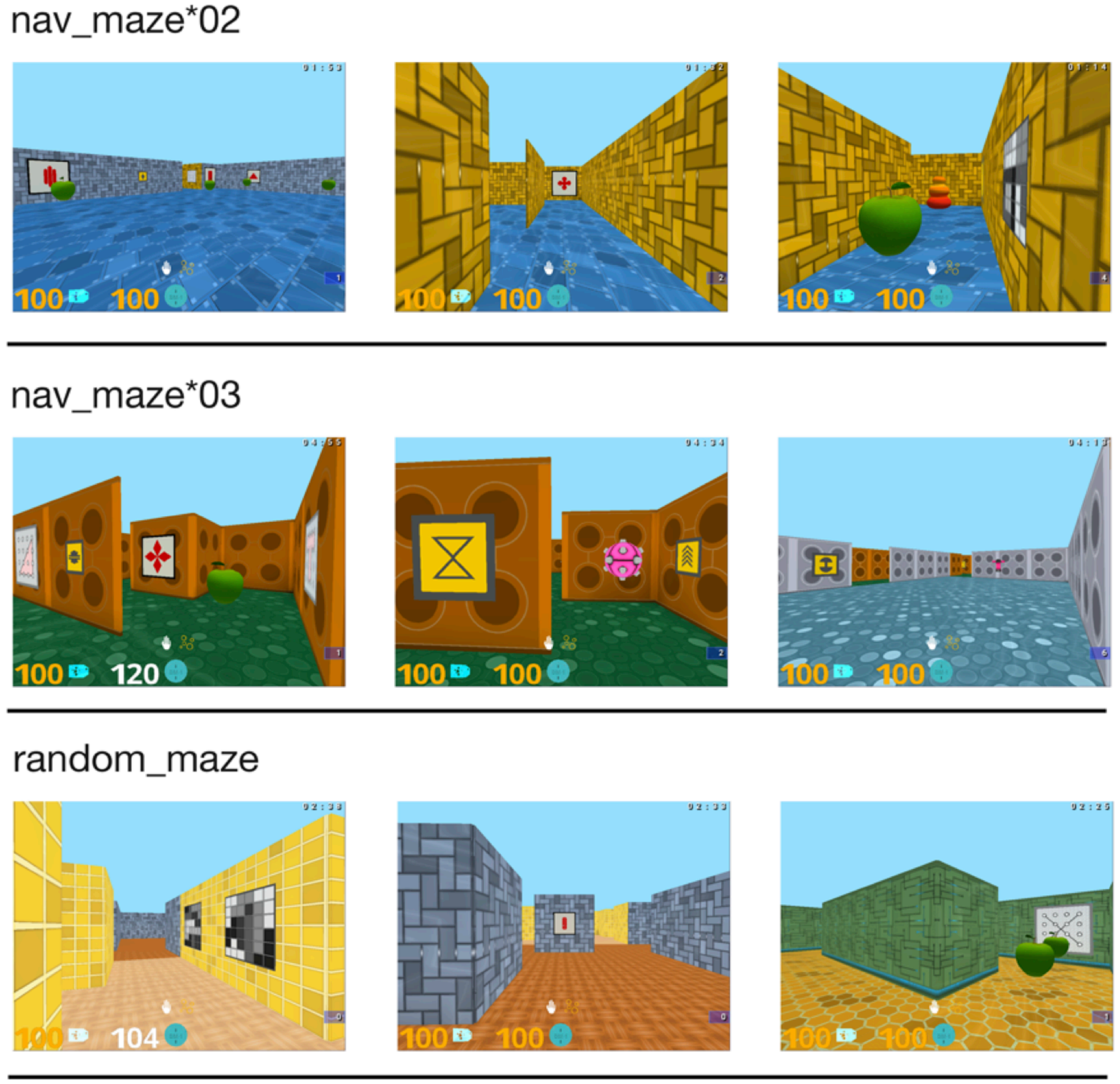
[Beattie et al., 2016]



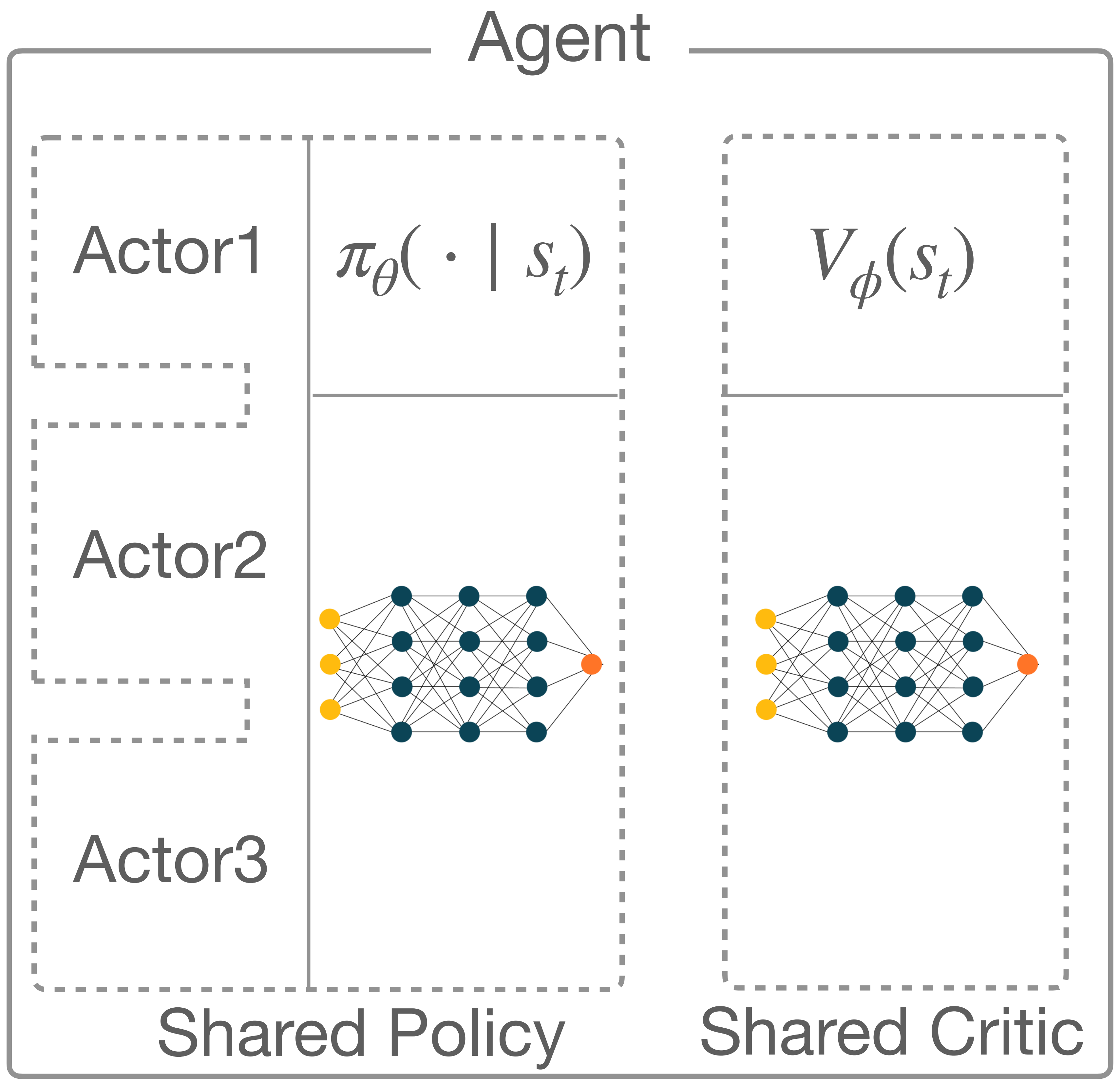
- ✓ Positive transfer of learned features across tasks
- ✓ Better compute efficiency
- ✗ Compatible action and state space

Motivation | Multi-Task RL





[Beattie et al., 2016]



- ✓ Positive transfer of learned features across tasks
- ✓ Better compute efficiency
- ✗ Compatible action and state space

One agent learns to perform optimally across several different tasks sharing similar characteristics

Motivation | Multi-Task RL | Division of Labor

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Motivation | Multi-Task RL | Division of Labor

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

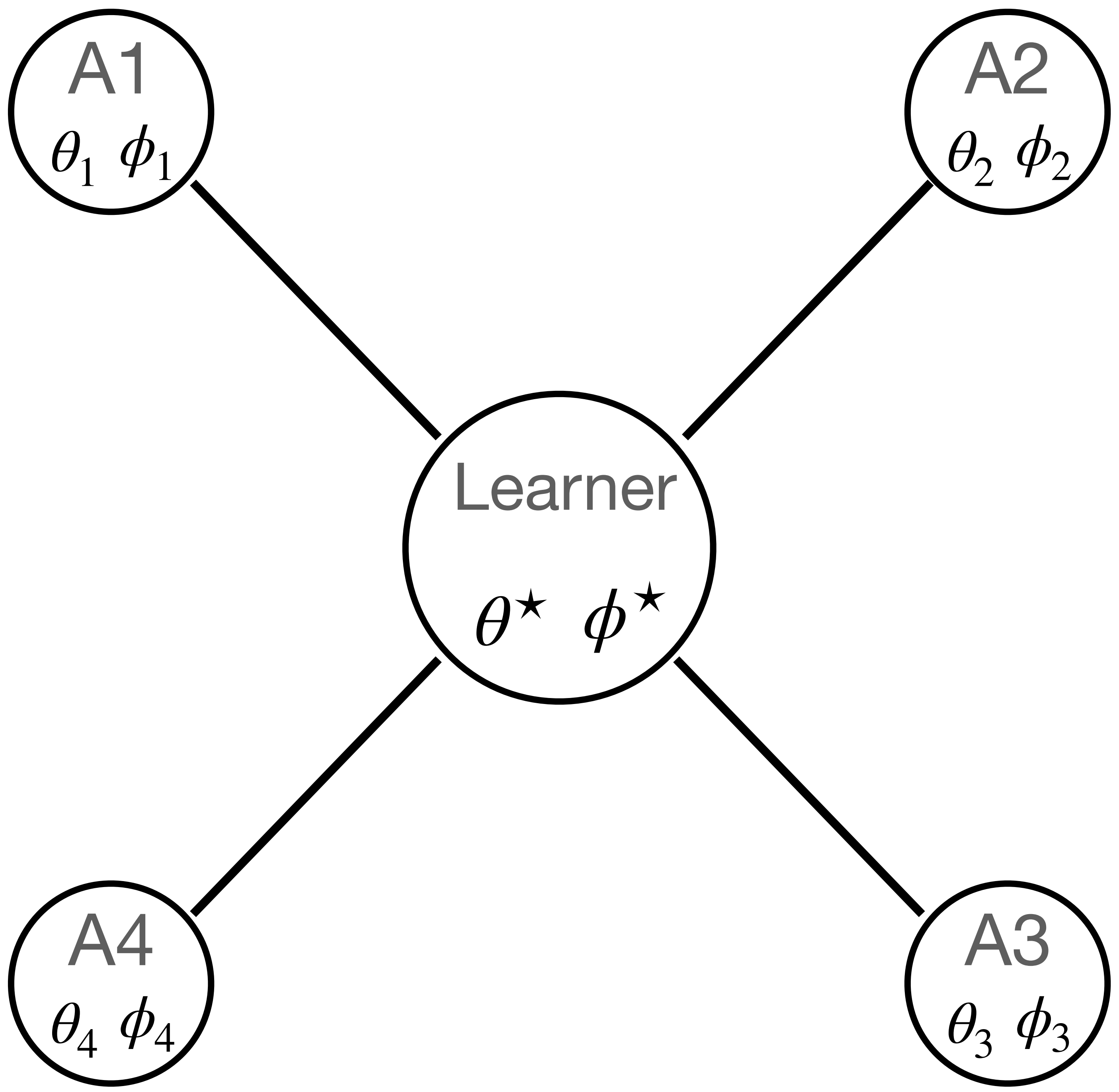
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Motivation | Multi-Task RL | Division of Labor

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

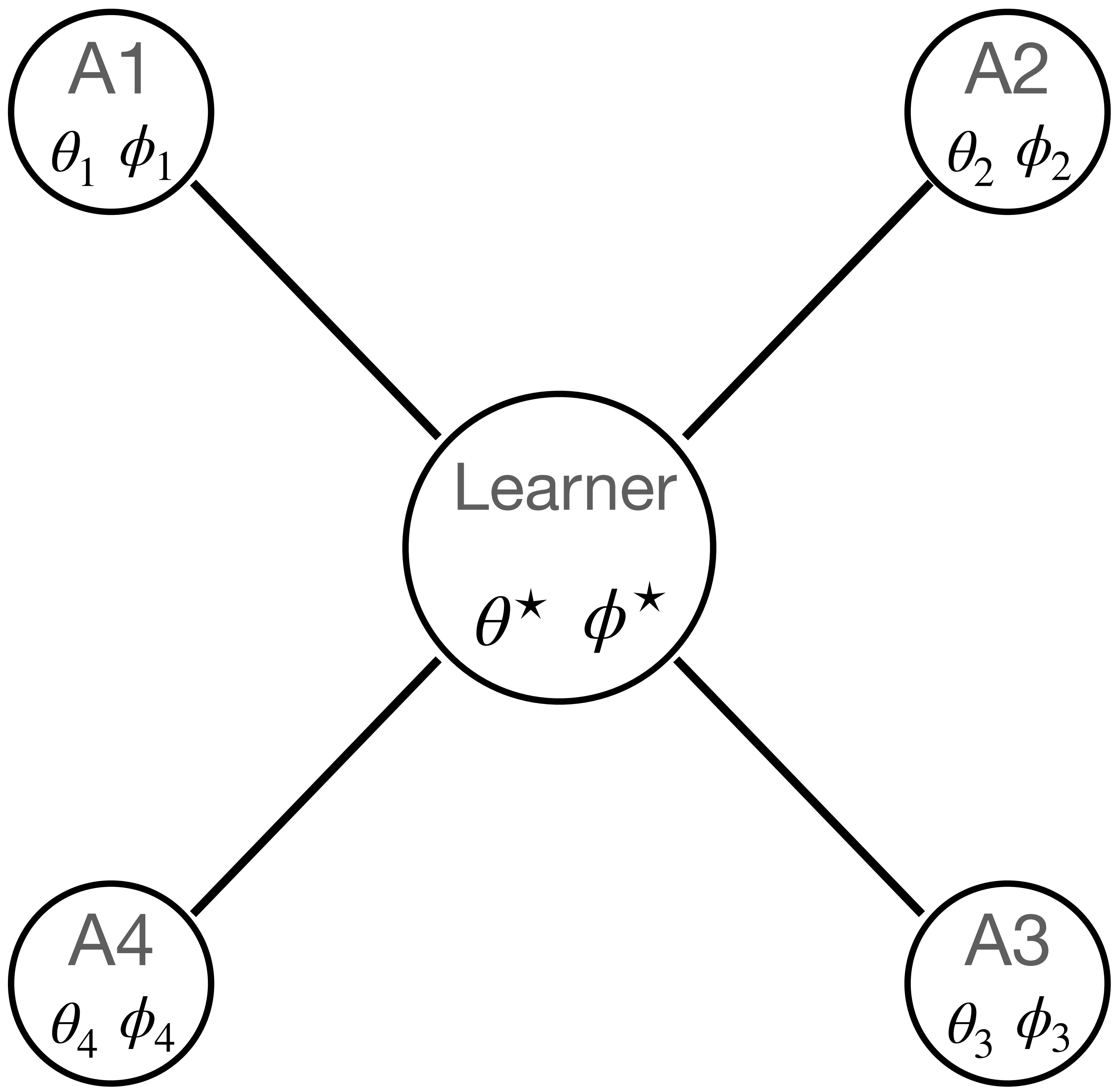
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Division of Labor

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

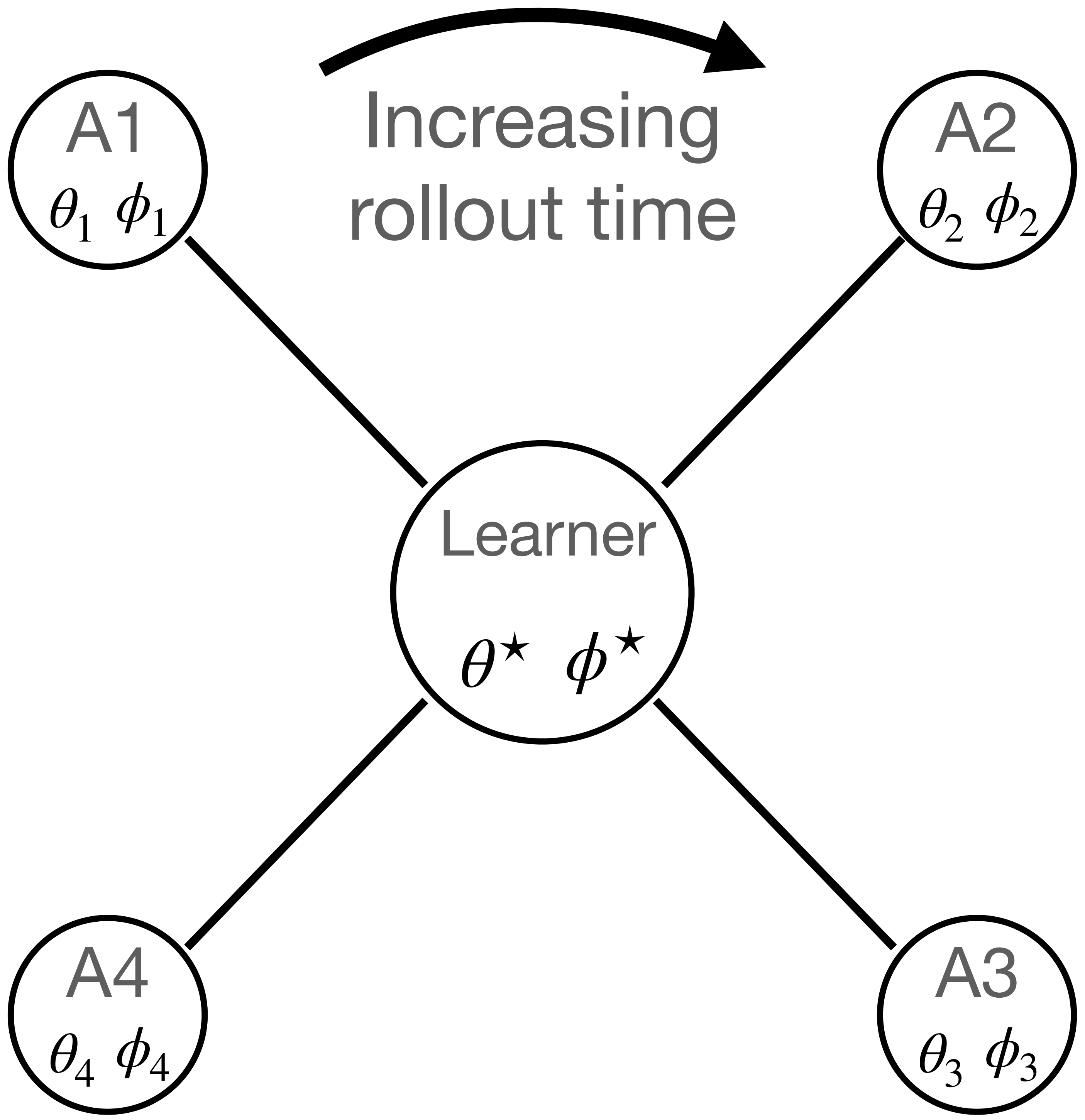
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Division of Labor

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

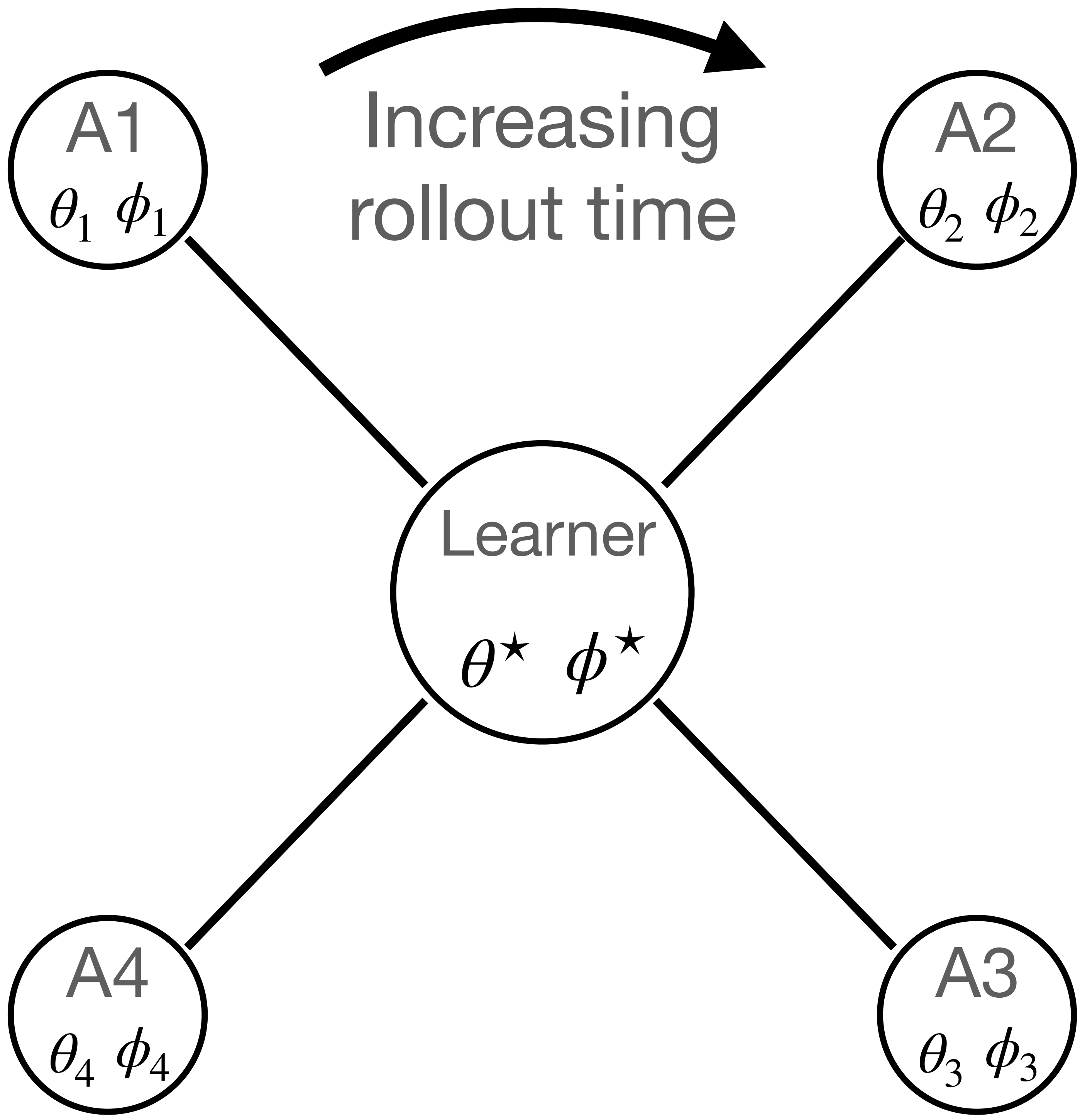
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Update global parameters for individual actor rollout completions
OR
Update global parameters for combined actor rollout completions

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

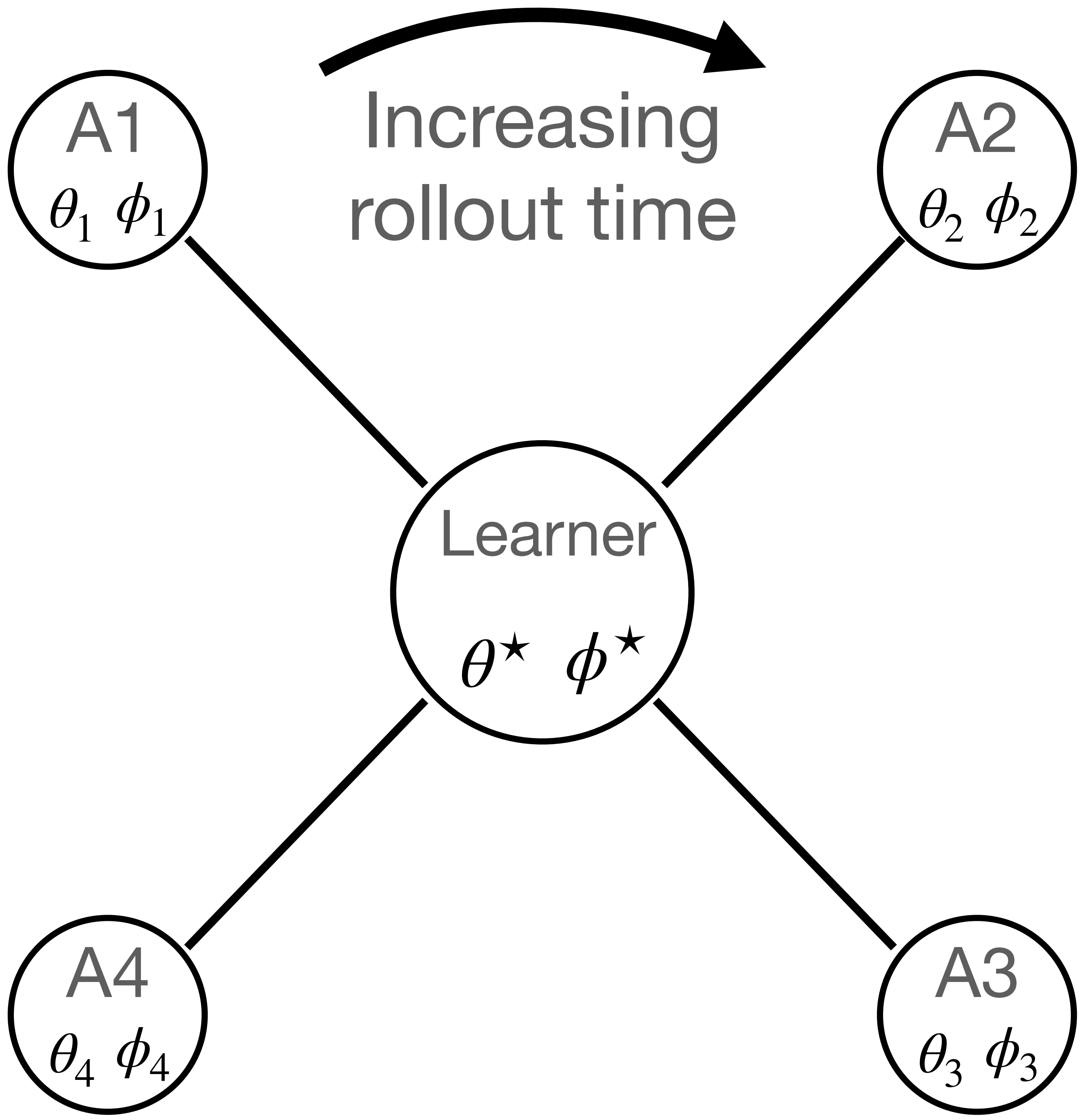
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Asynchronous Update

1 Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2 Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3 Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4 Value Gradient $\nabla_\phi V_\phi(s_t)$

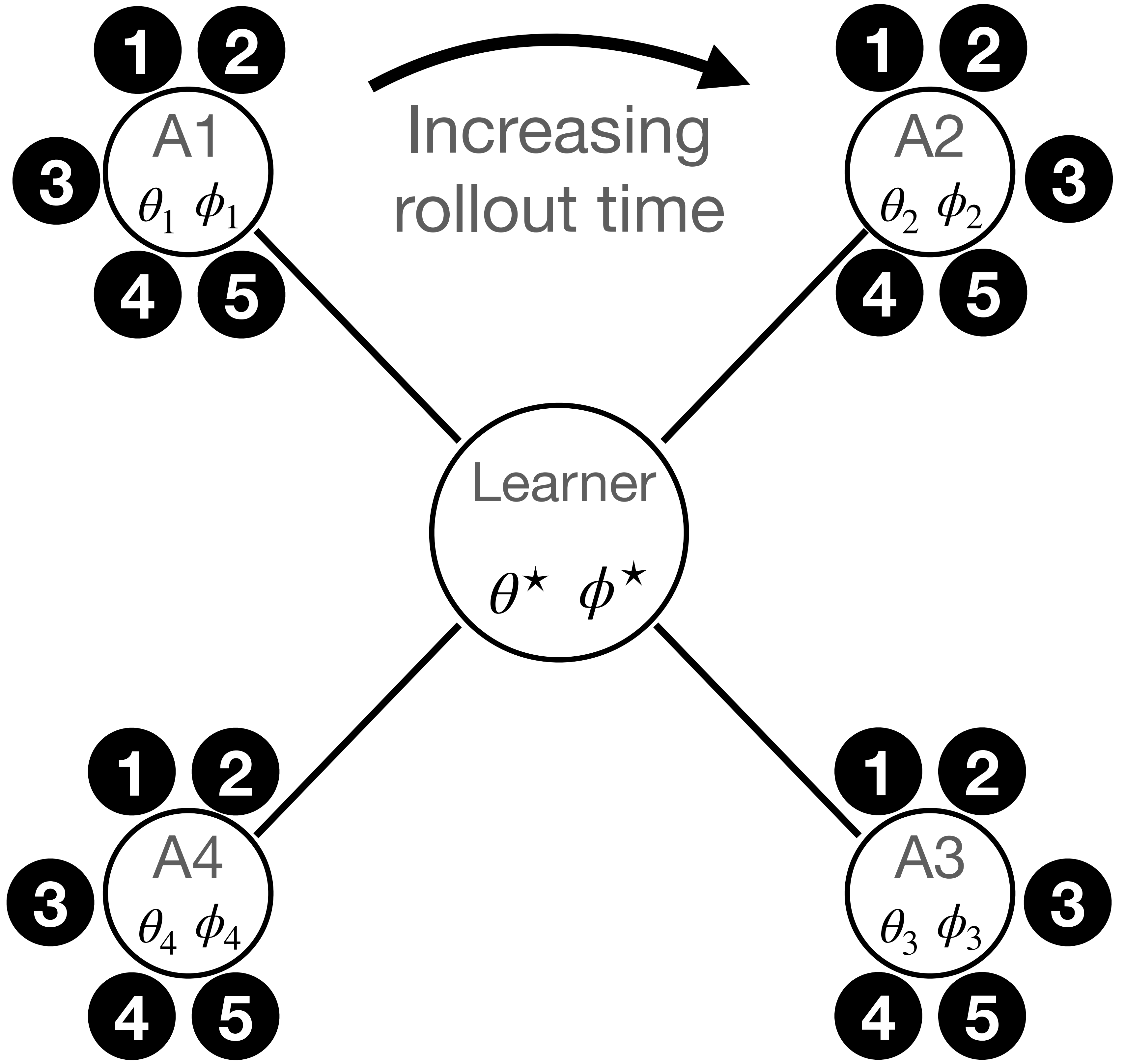
5 Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6 Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7 Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \qquad \forall i \in [1,4]$$
$$\phi_i \leftarrow \phi^\star$$

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

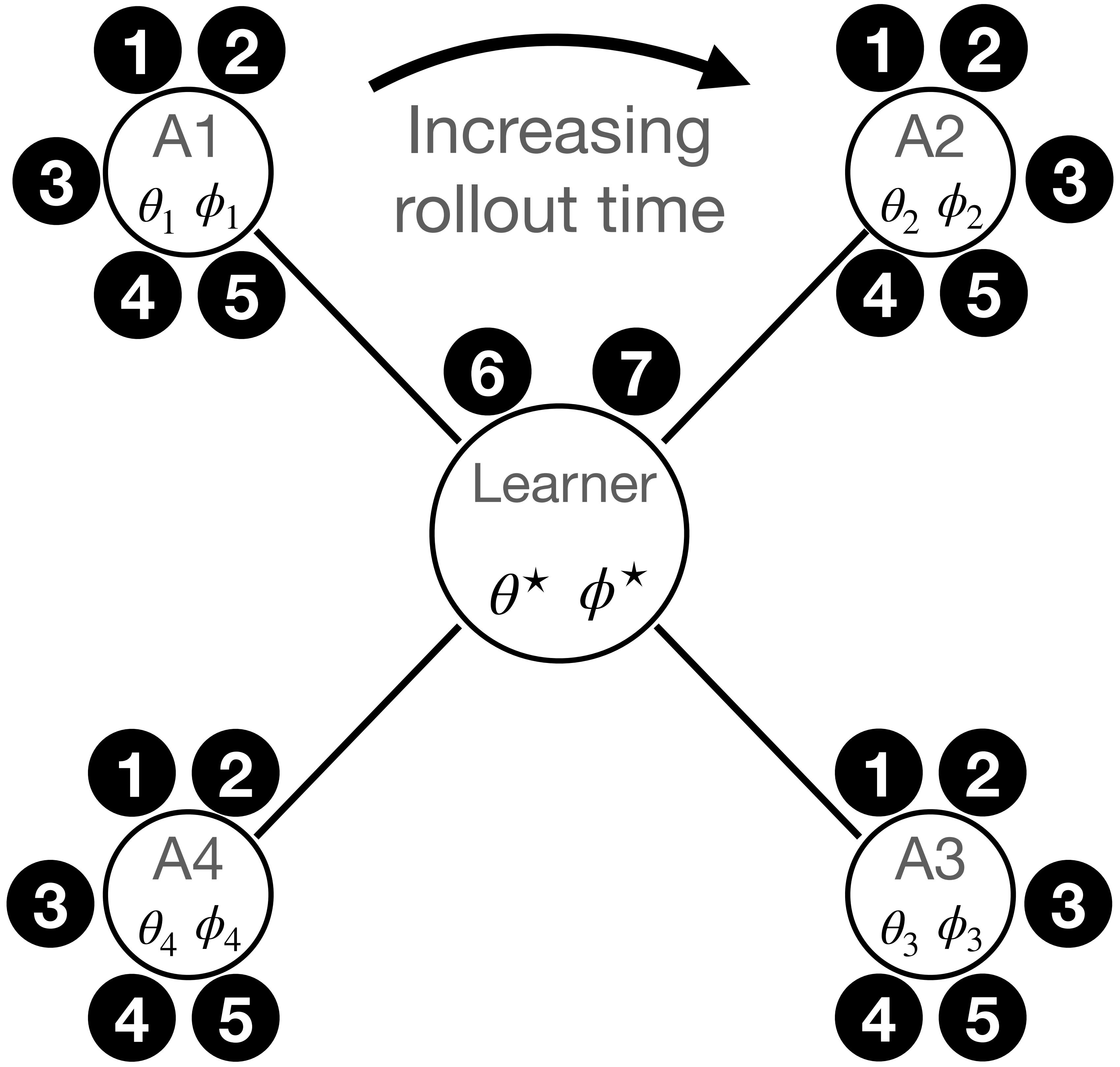
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \forall i \in [1,4]$$
$$\phi_i \leftarrow \phi^\star$$

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

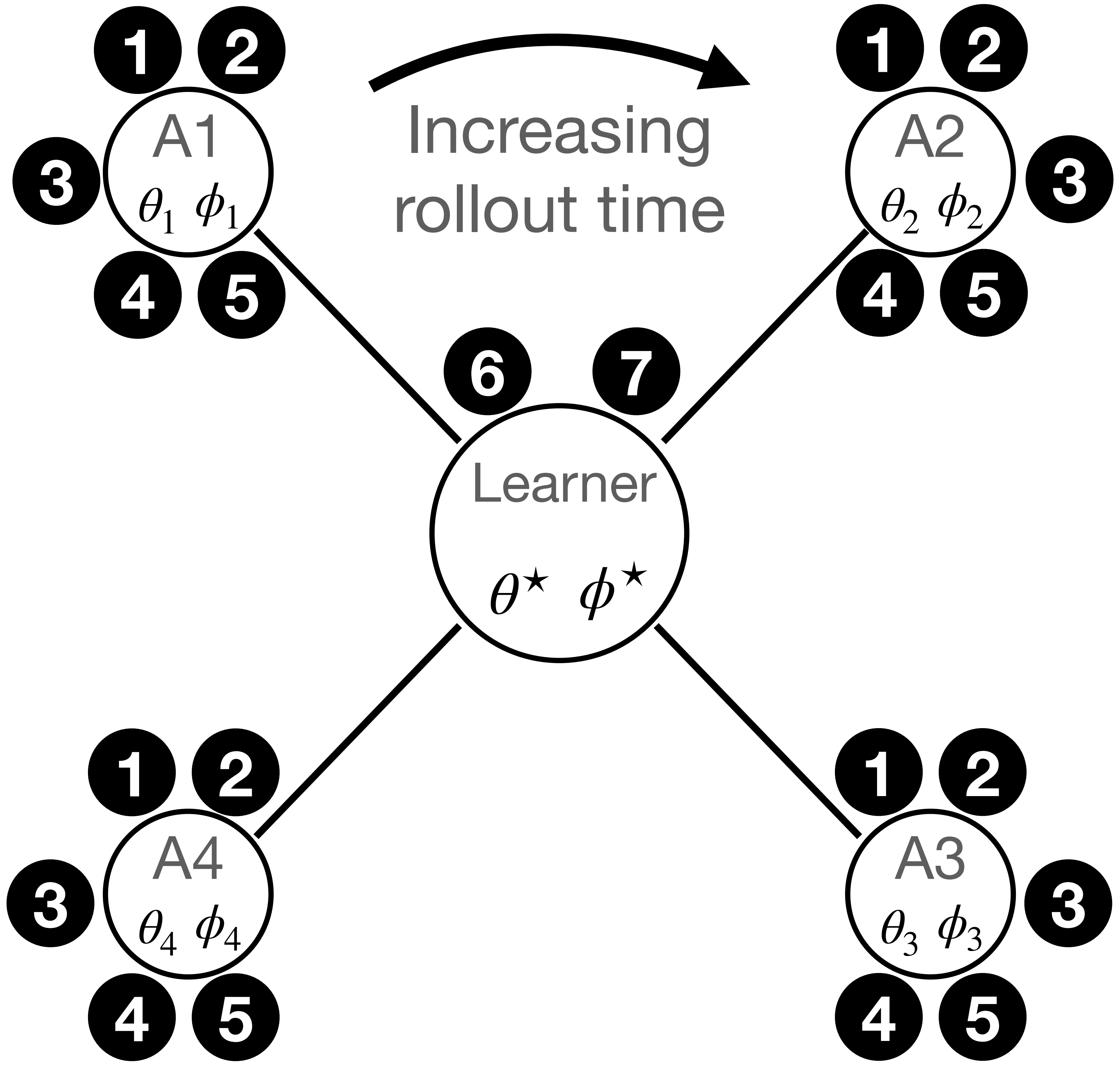
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \boxed{\delta_t \nabla_\phi V_\phi(s_t)}$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \boxed{\delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)}$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

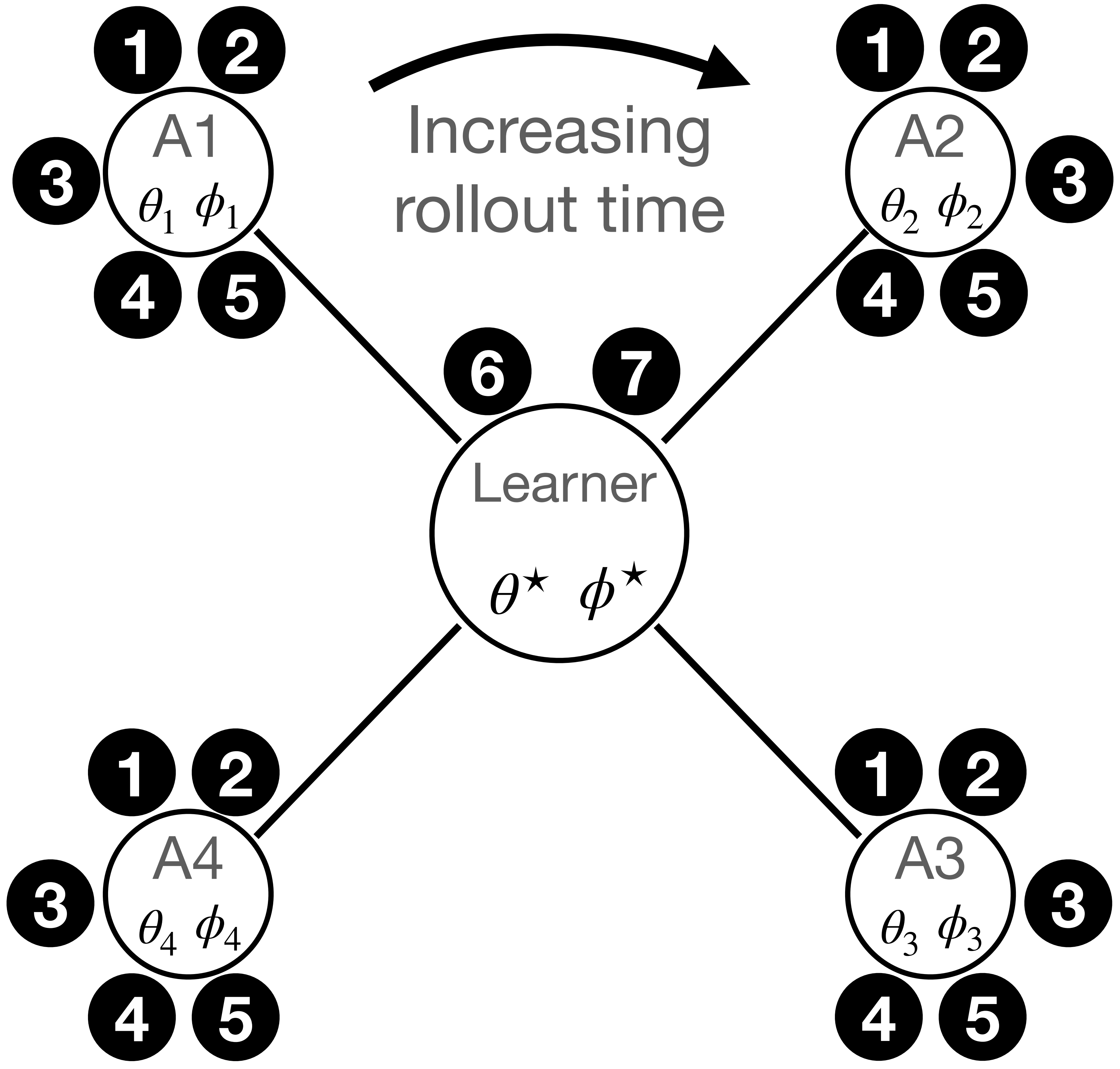
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \boxed{\delta_t \nabla_\phi V_\phi(s_t)}$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \boxed{\delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)}$$



$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_1, \theta_1}$$

Global update for A1

Initial

$$\begin{aligned}\theta_i &\leftarrow \theta^\star\\ \phi_i &\leftarrow \phi^\star\end{aligned} \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

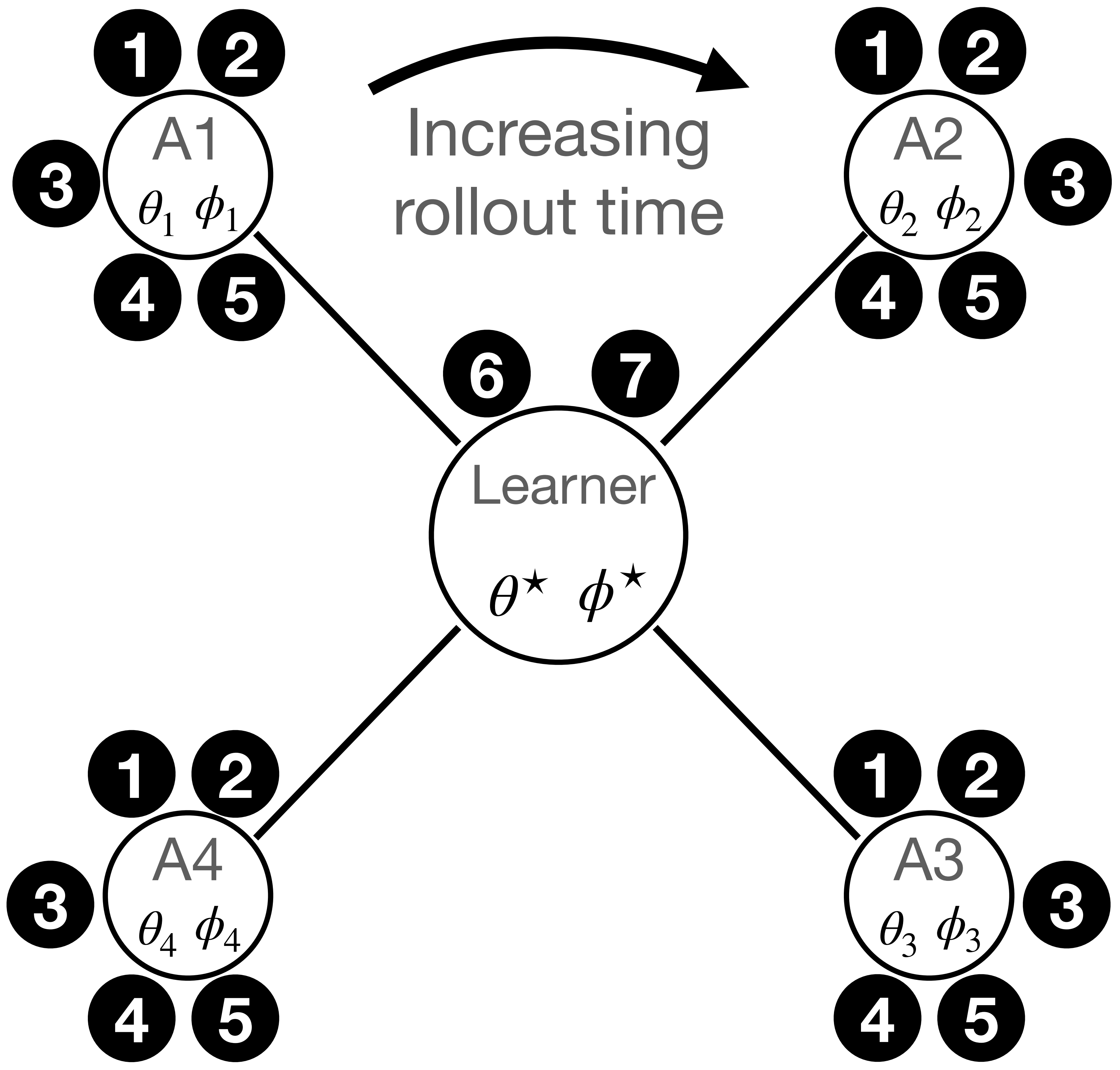
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \boxed{\delta_t \nabla_\phi V_\phi(s_t)}$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \boxed{\delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)}$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

$$\left.\begin{aligned} \phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t) \\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t) \end{aligned}\right|_{\phi_1, \theta_1}$$

Global update for A1

$$\left.\begin{aligned} \phi_2^\star &\leftarrow \phi_1^\star + \delta_t \nabla_\phi V_\phi(s_t) \\ \theta_2^\star &\leftarrow \theta_1^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t) \end{aligned}\right|_{\phi_2, \theta_2}$$

Global update for A2

$\cdots$

Motivation | Multi-Task RL | Asynchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

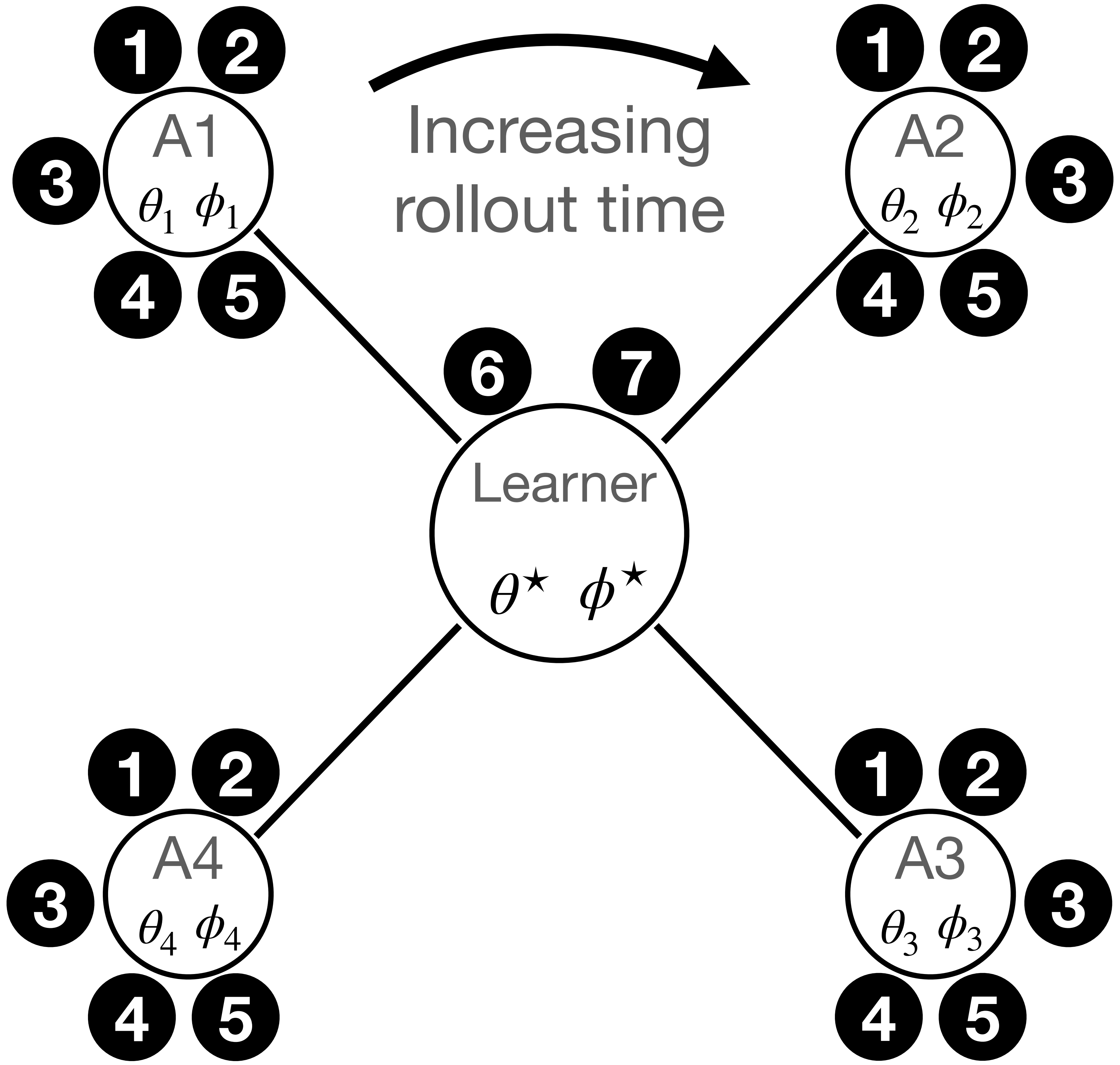
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \boxed{\delta_t \nabla_\phi V_\phi(s_t)}$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \boxed{\delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)}$$



A3C
(Mnih et al., 2016)

Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t) \\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_1, \theta_1}$$

Global update for A1

$$\left.\begin{aligned}\phi_2^\star &\leftarrow \phi_1^\star + \delta_t \nabla_\phi V_\phi(s_t) \\ \theta_2^\star &\leftarrow \theta_1^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_2, \theta_2} \quad \cdots$$

Global update for A2

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

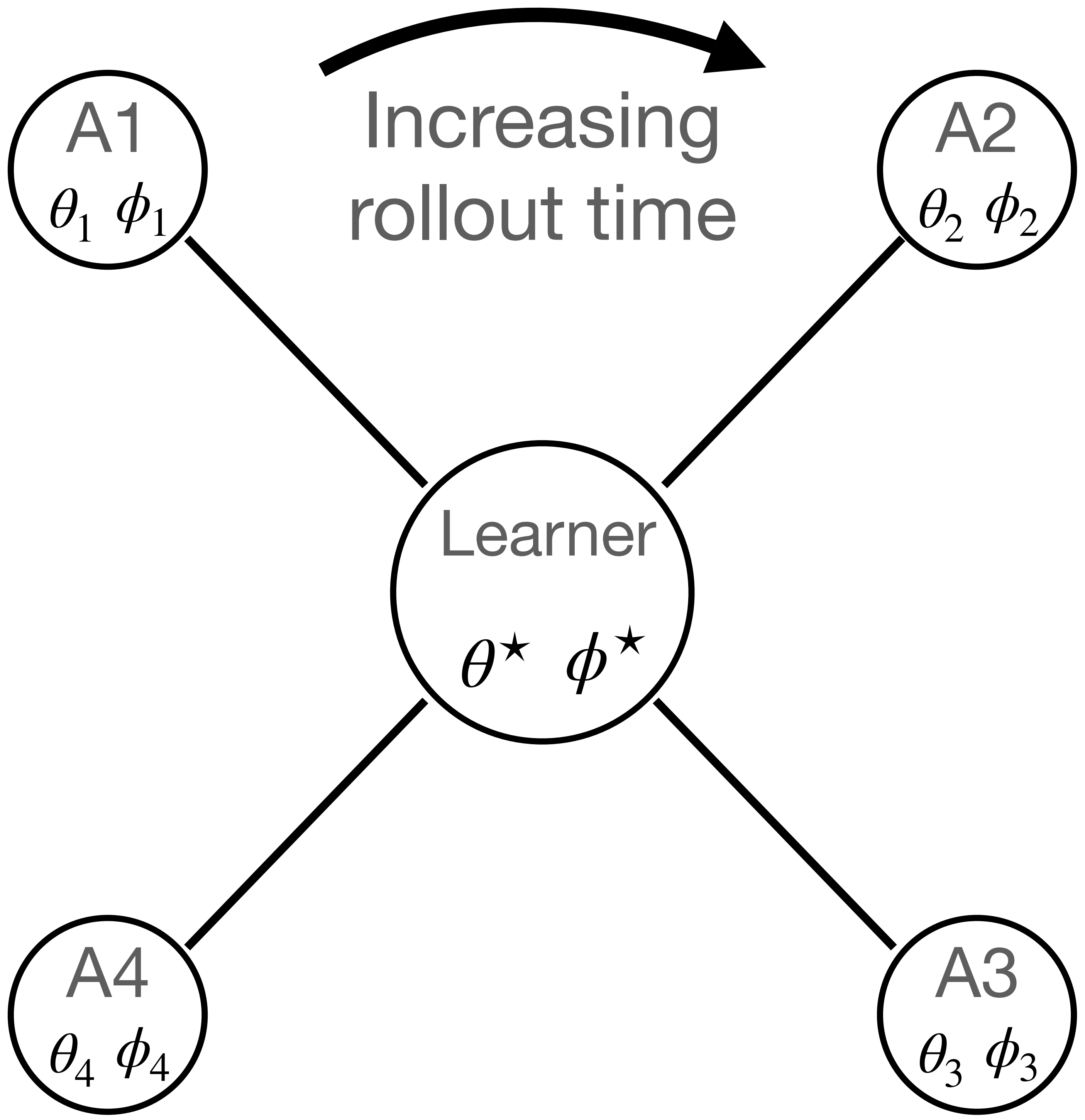
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

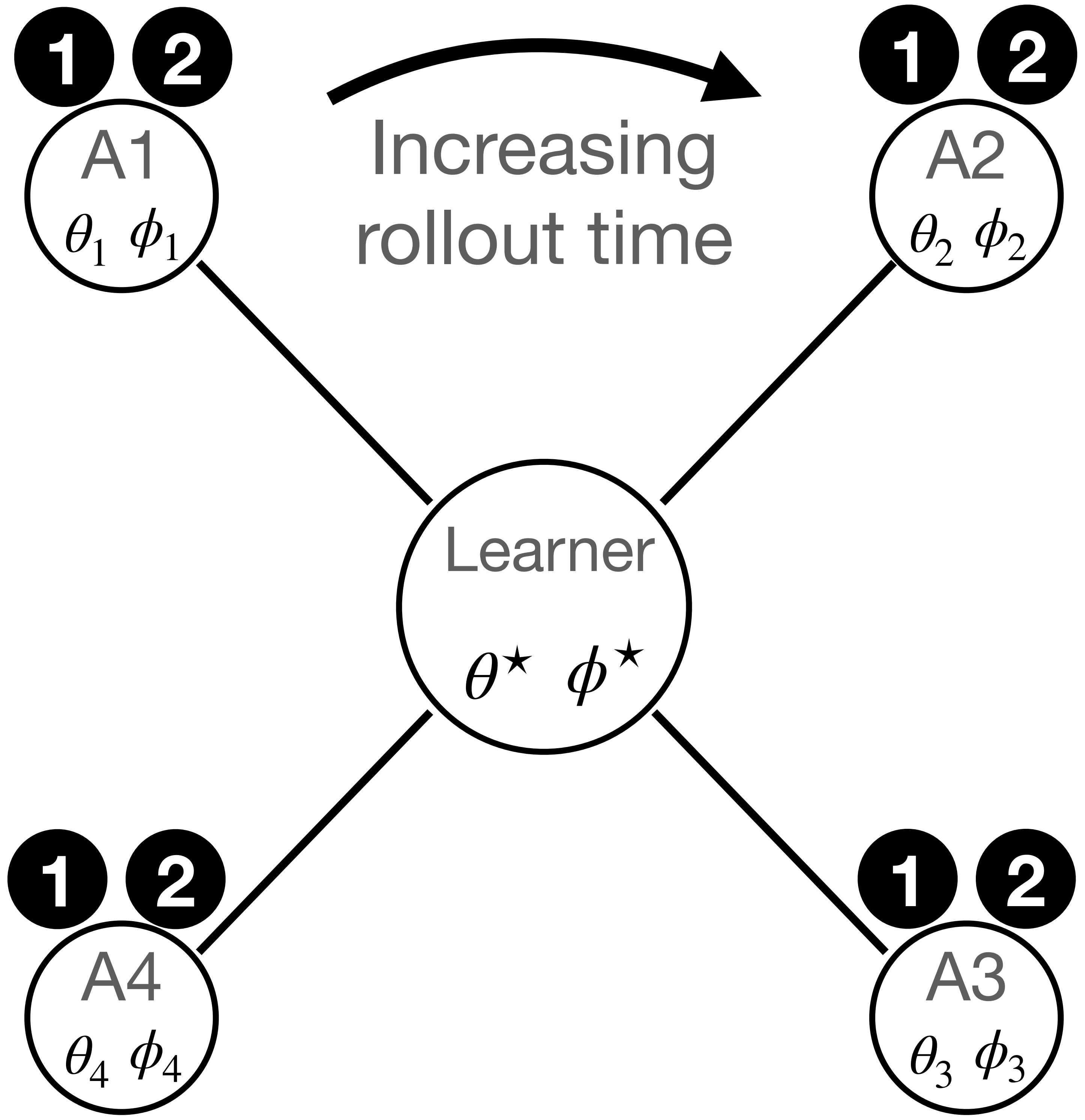
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\begin{aligned}\theta_i &\leftarrow \theta^\star \\ \phi_i &\leftarrow \phi^\star\end{aligned} \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\, s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

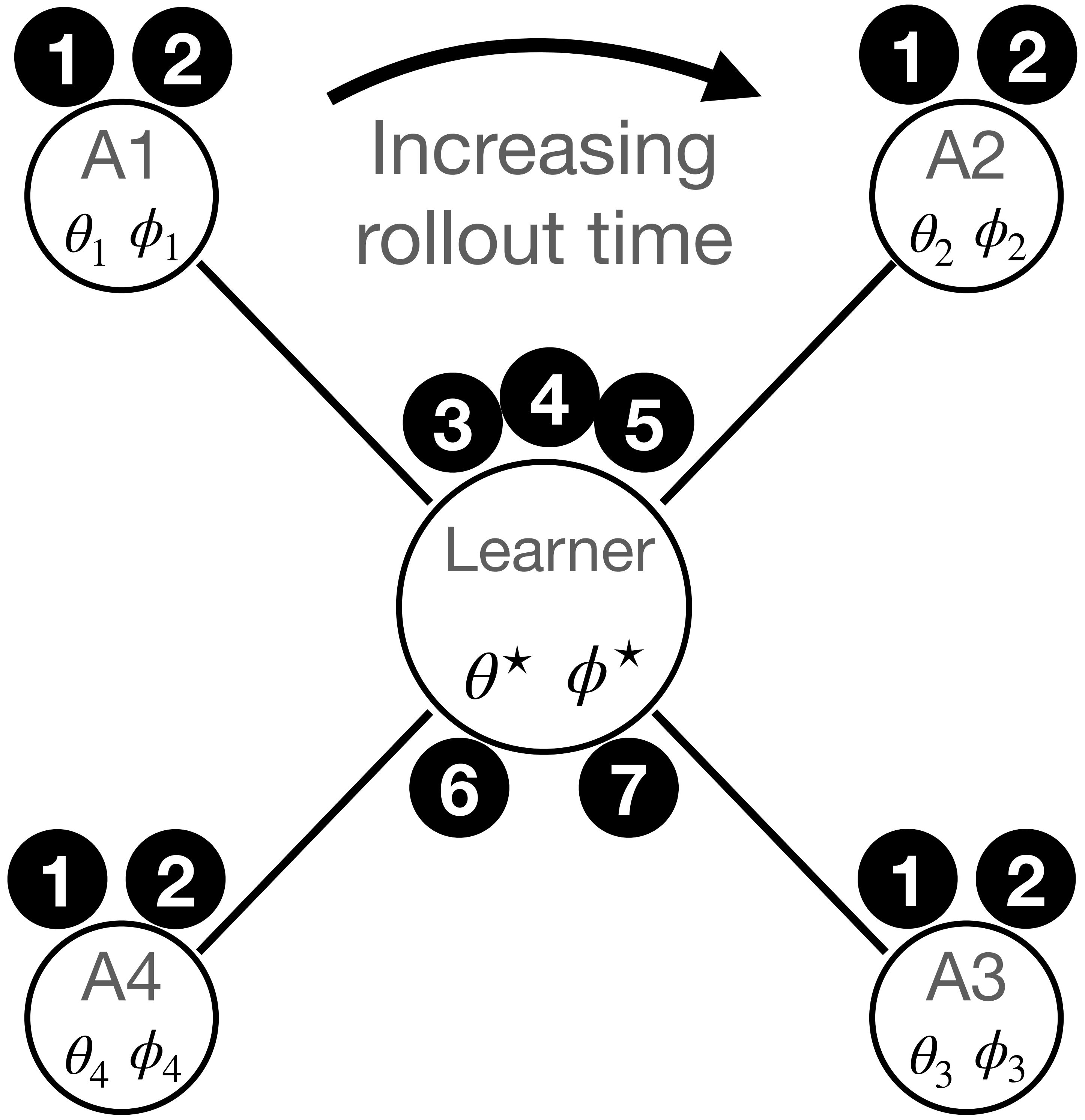
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \qquad \forall i \in [1,4]$$
$$\phi_i \leftarrow \phi^\star$$

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

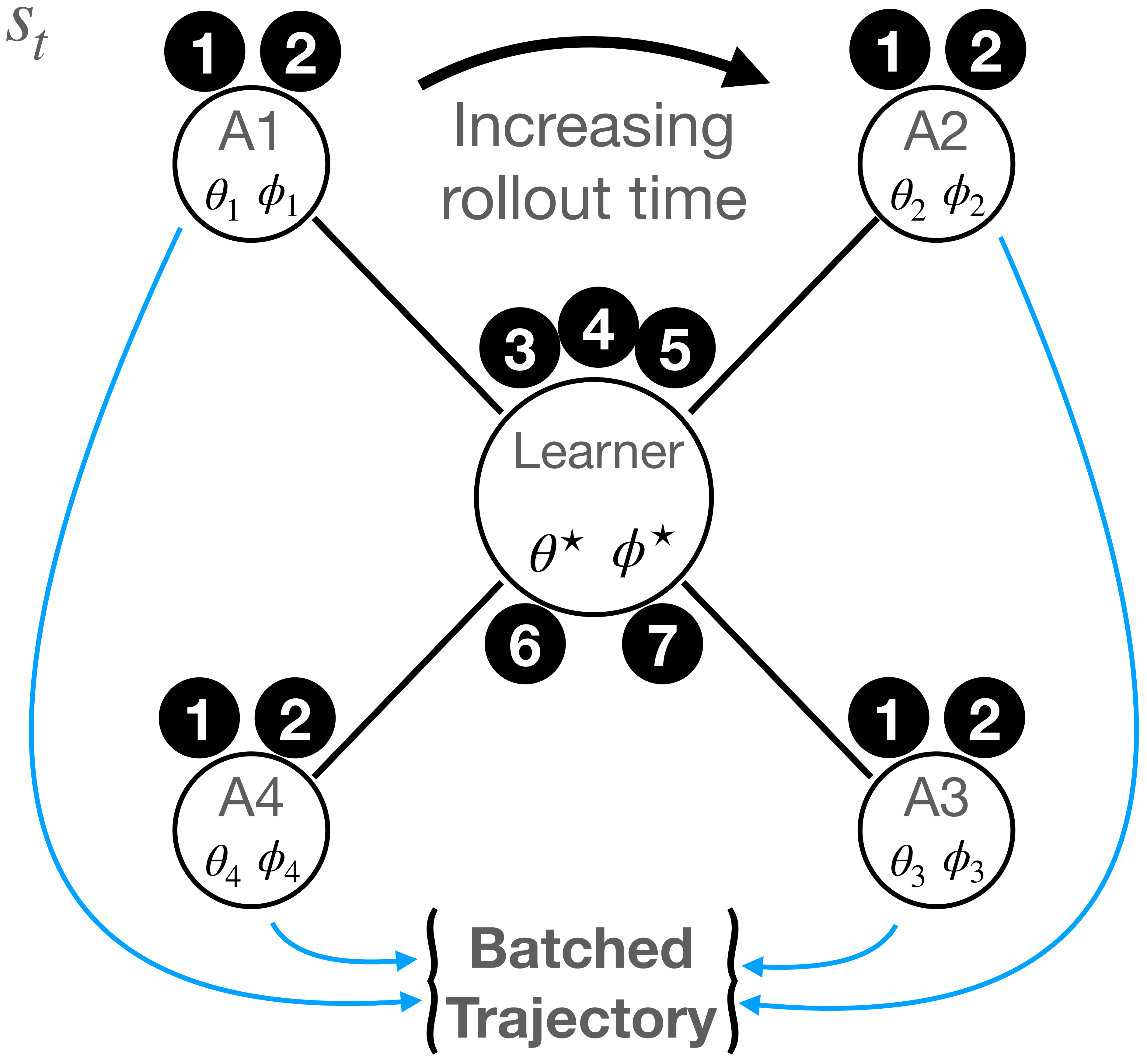
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \quad \phi_i \leftarrow \phi^\star \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

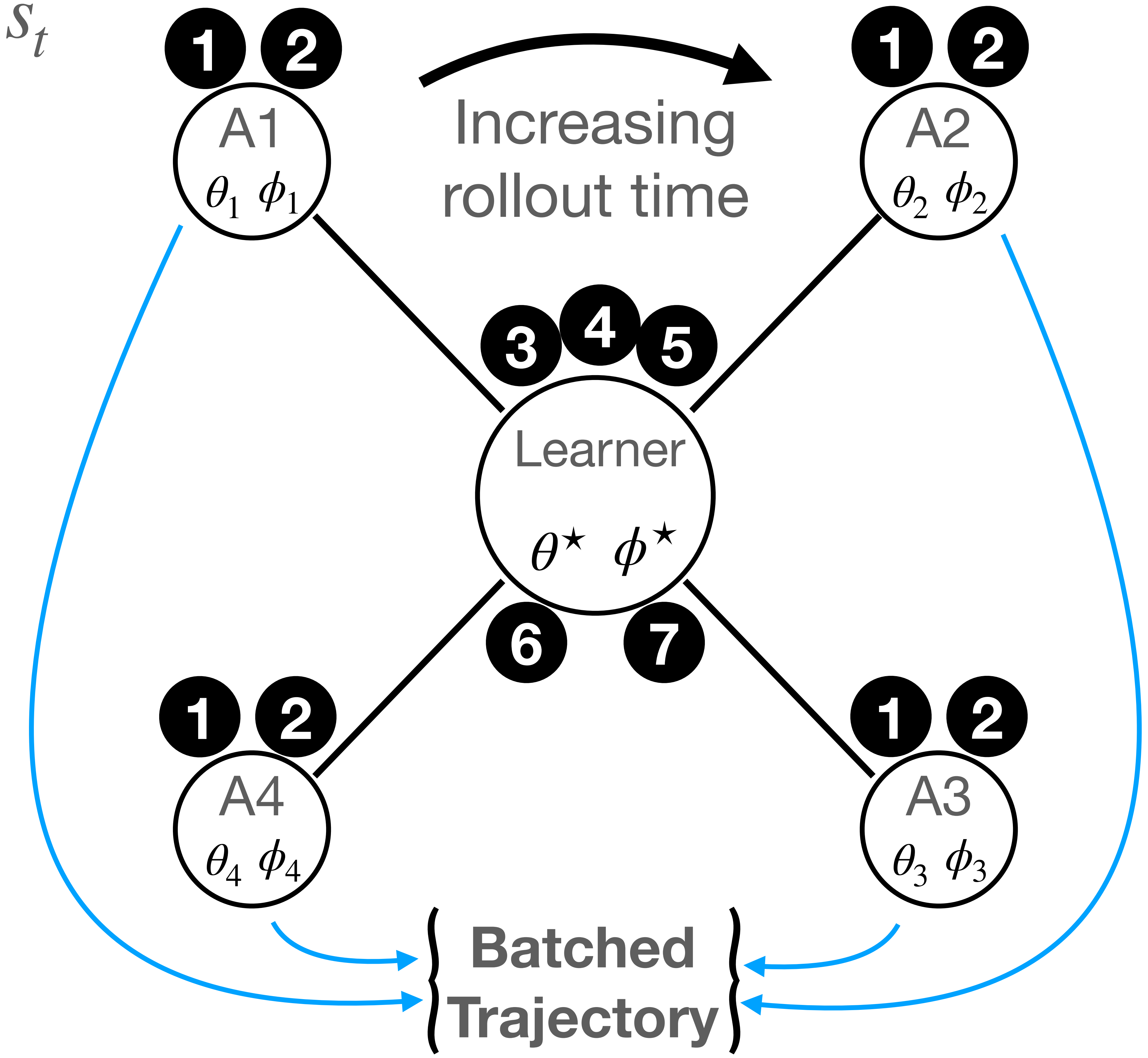
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Initial

$$\theta_i \leftarrow \theta^\star \qquad \forall i \in [1,4]$$
$$\phi_i \leftarrow \phi^\star$$

$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi^\star,\,\theta^\star}$$

Single Global Update

Motivation | Multi-Task RL | Synchronous Update

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

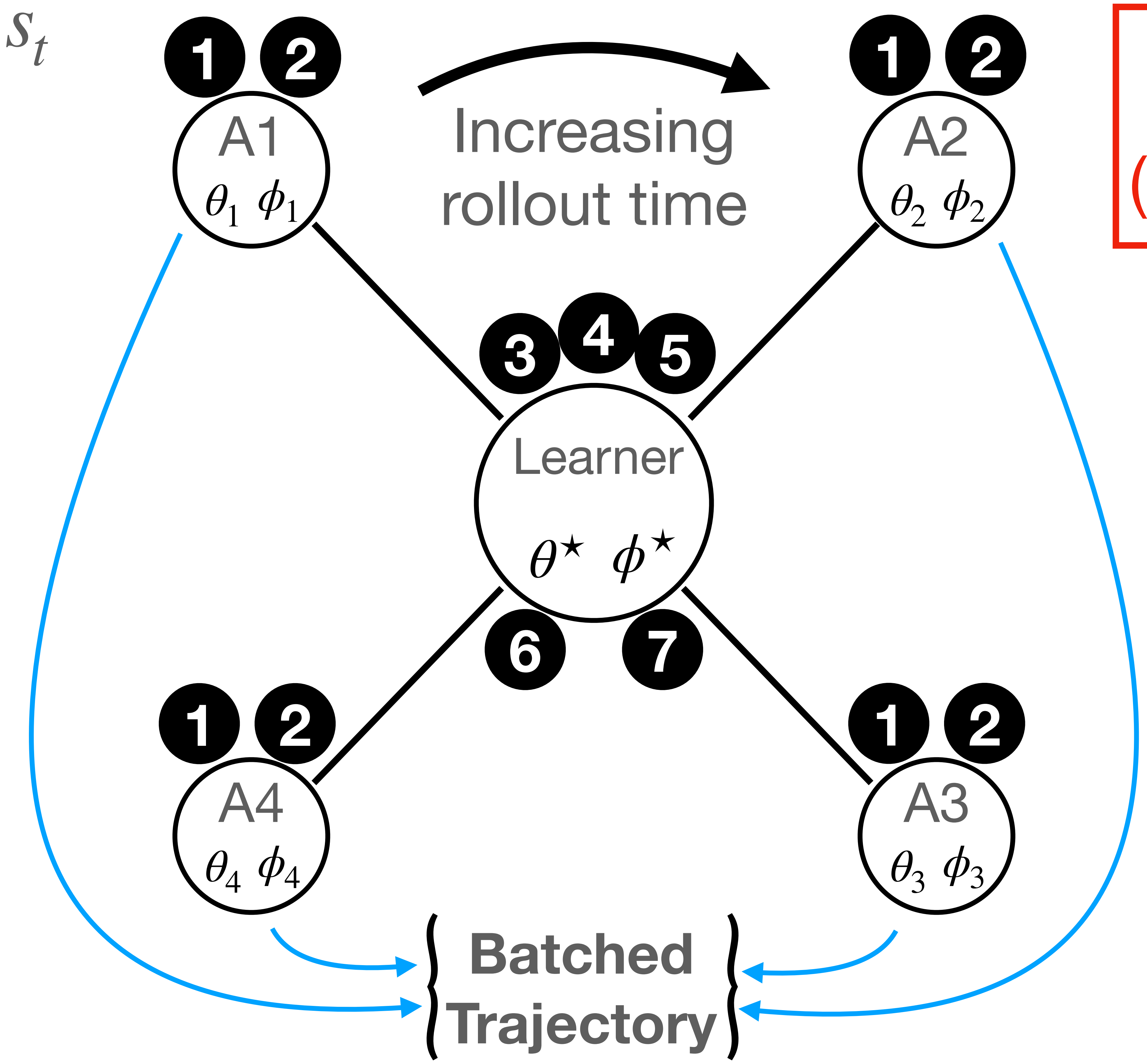
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi^\star, \theta^\star}$$

Single Global Update

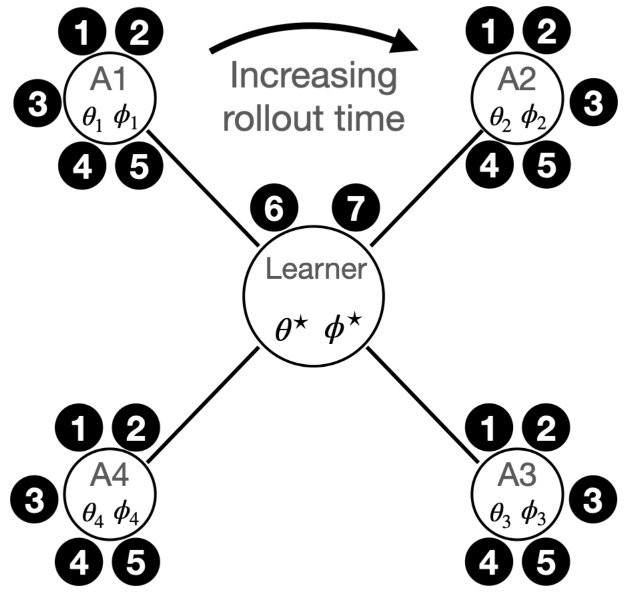
A2C
(Clemente et al., 2017)

Initial

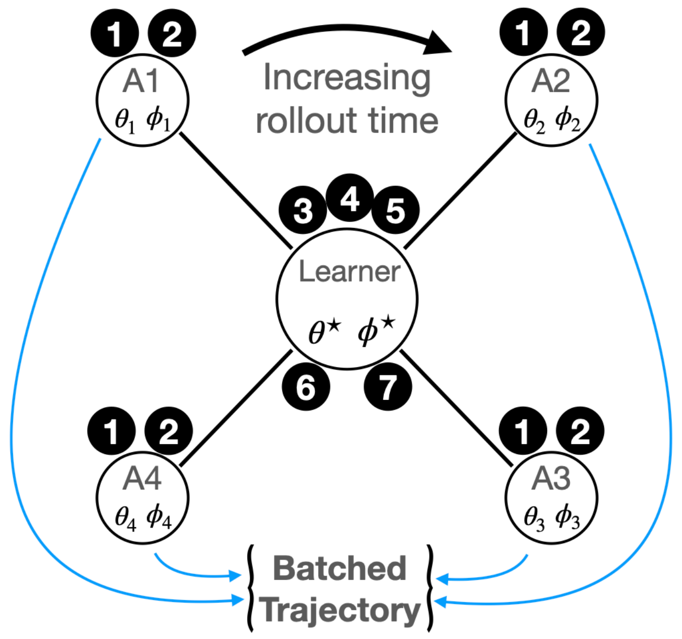
$$\begin{aligned}\theta_i &\leftarrow \theta^\star\\ \phi_i &\leftarrow \phi^\star\end{aligned} \quad \forall i \in [1,4]$$

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)

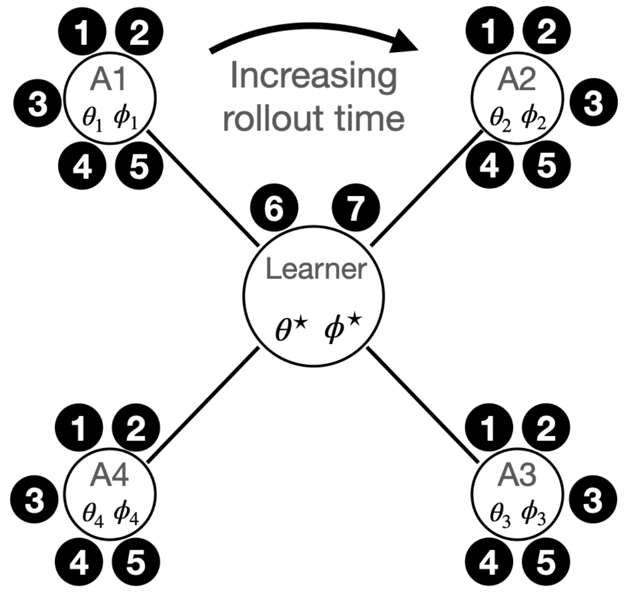


A2C (Synchronous)

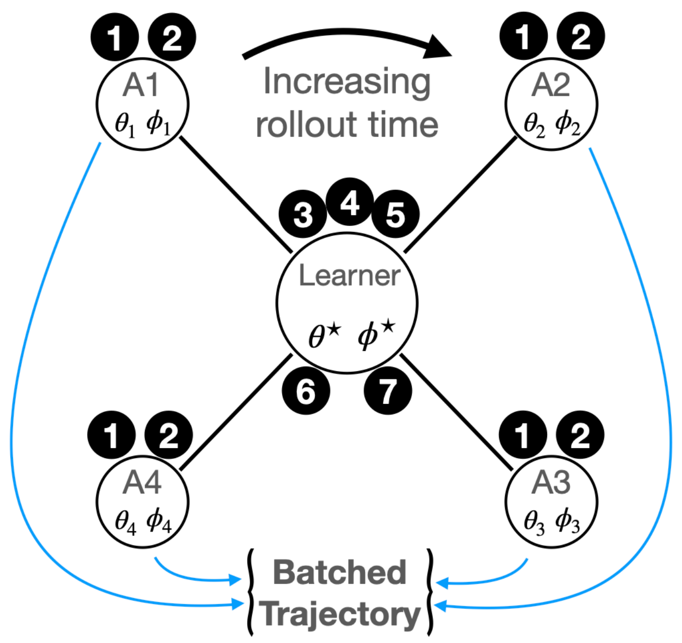


Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



A2C (Synchronous)

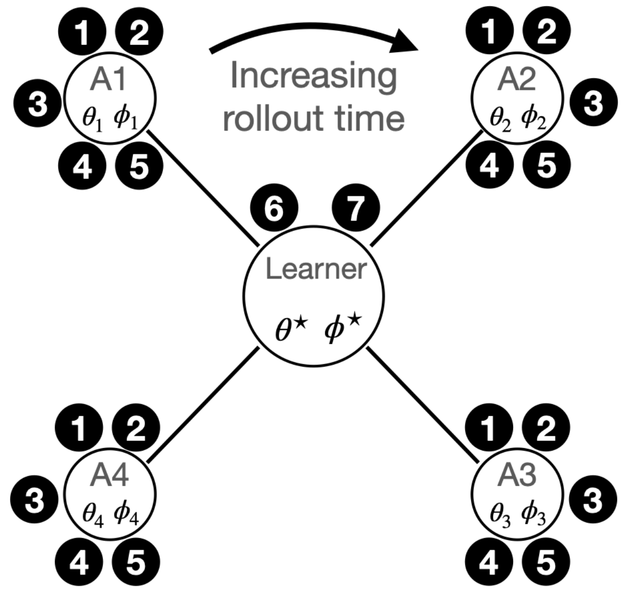


✓ Global update per actor rollout

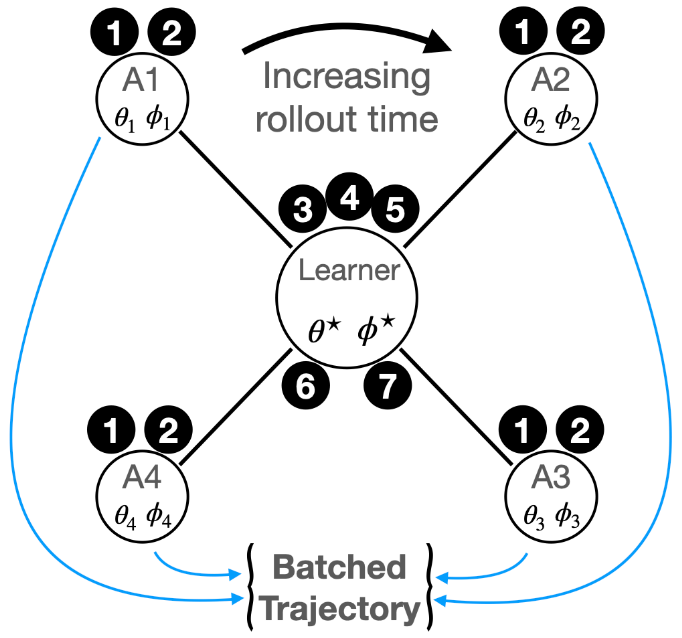
✗ Global update bottlenecked by slowest actor

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



A2C (Synchronous)

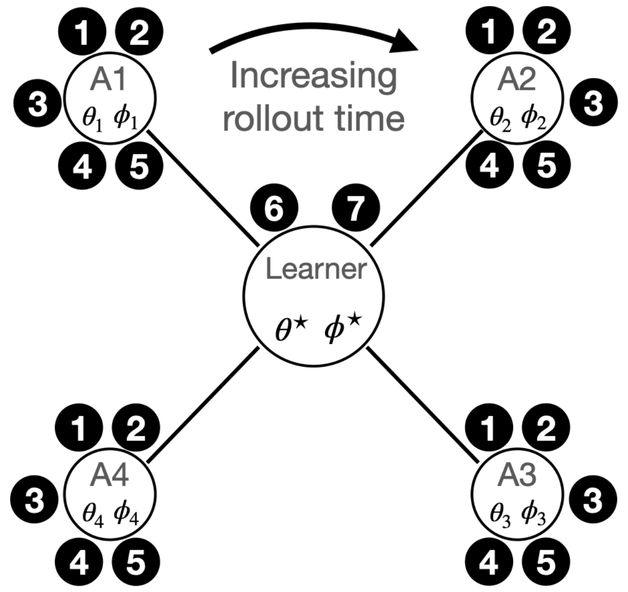


- ✓ Global update per actor rollout
- ✗ Local gradient computation - GPU underutilized

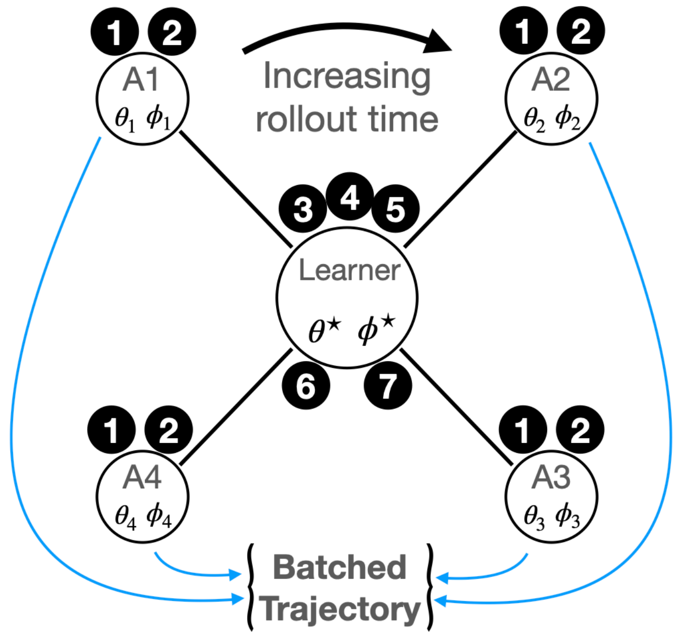
- ✗ Global update bottlenecked by slowest actor
- ✓ Batched gradient computation - GPU scalable

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



A2C (Synchronous)

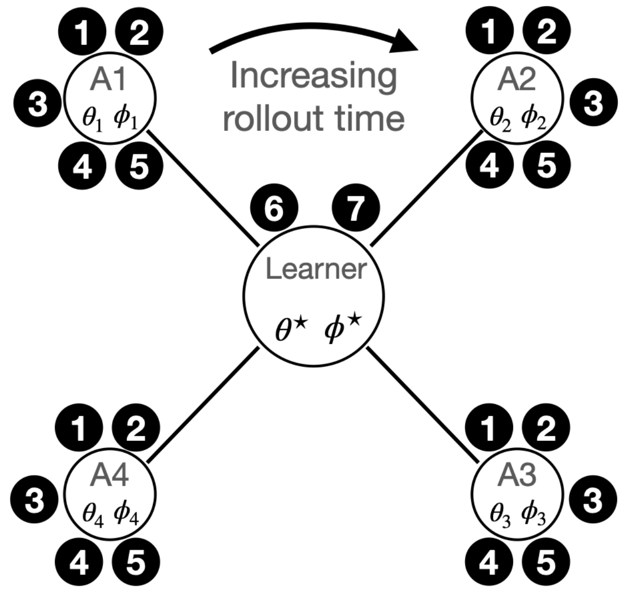


- ✓ Global update per actor rollout
- ✗ Local gradient computation - GPU underutilized
- ✗ Bandwidth Limited - Transfer massive gradients

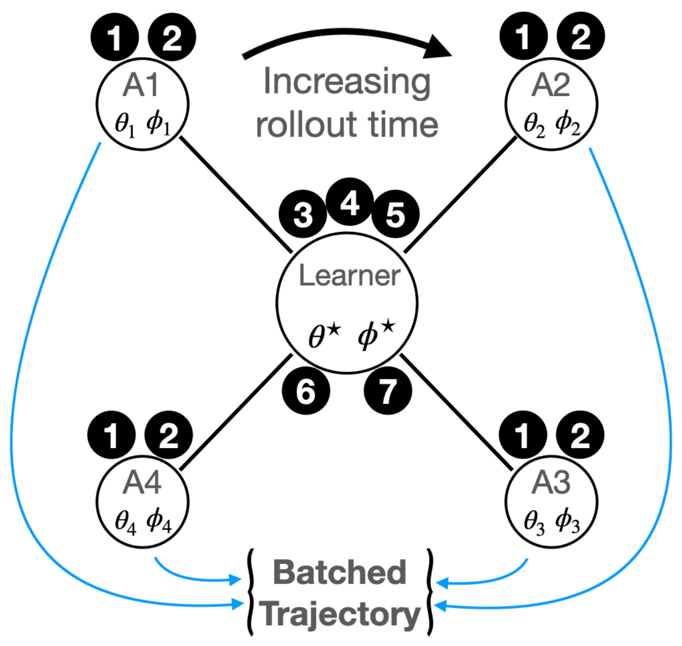
- ✗ Global update bottlenecked by slowest actor
- ✓ Batched gradient computation - GPU scalable
- ✓ Better bandwidth utilization - Transfer trajectories

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



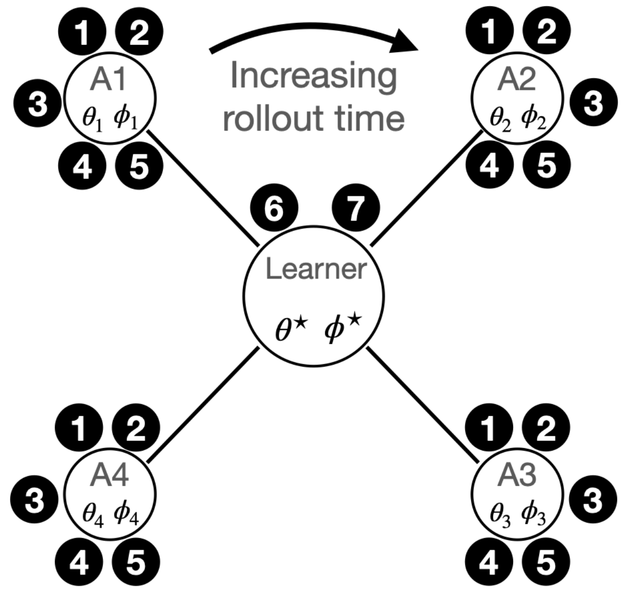
A2C (Synchronous)



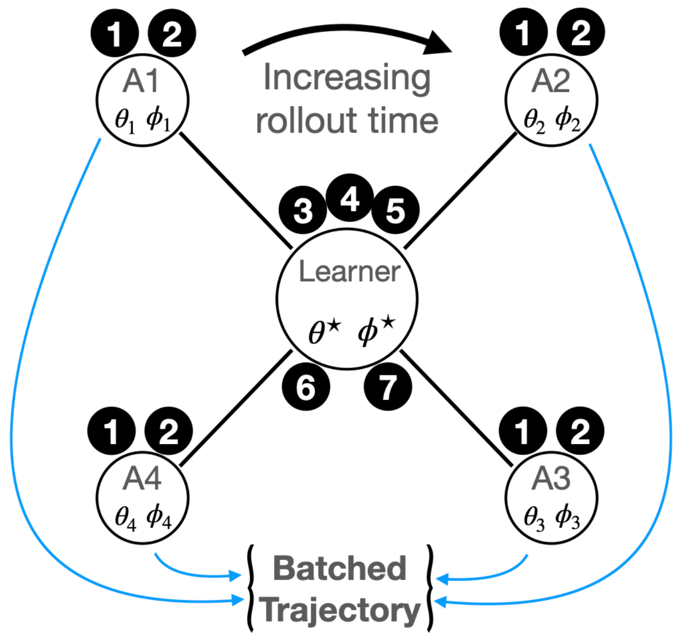
A3C (Asynchronous)	A2C (Synchronous)
✓ Global update per actor rollout	✗ Global update bottlenecked by slowest actor
✗ Local gradient computation - GPU underutilized	✓ Batched gradient computation - GPU scalable
✗ Bandwidth Limited - Transfer massive gradients	✓ Better bandwidth utilization - Transfer trajectories
✗ Unstable - Policy lag due to asynchronous updates	✓ Stable - No Policy Lag

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



A2C (Synchronous)



A3C (Asynchronous)	A2C (Synchronous)
✓ Global update per actor rollout	✗ Global update bottlenecked by slowest actor
✗ Local gradient computation - GPU underutilized	✓ Batched gradient computation - GPU scalable
✗ Bandwidth Limited - Transfer massive gradients	✓ Better bandwidth utilization - Transfer trajectories
✗ Unstable - Policy lag due to asynchronous updates	✓ Stable - No Policy Lag

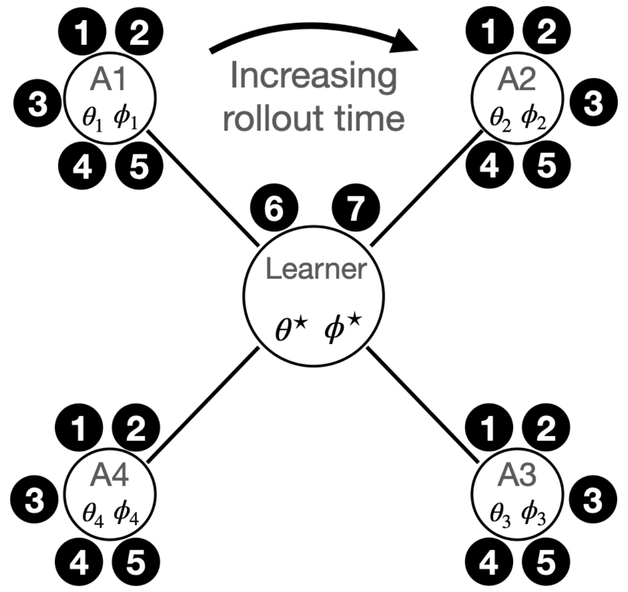
$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_1, \theta_1} \qquad \left.\begin{aligned}\phi_2^\star &\leftarrow \phi_1^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_2^\star &\leftarrow \theta_1^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_2, \theta_2} \quad \cdots$$

Global update for A1

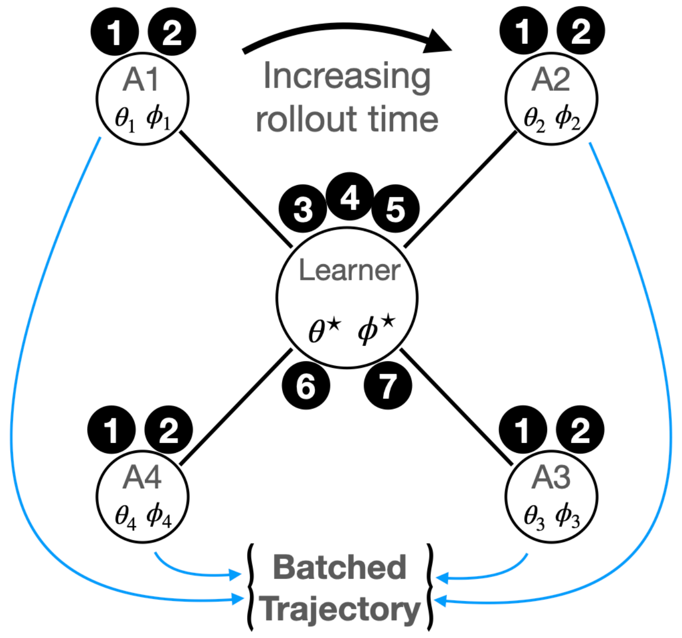
Global update for A2

Motivation | Multi-Task RL | Trade-Offs

A3C (Asynchronous)



A2C (Synchronous)



A3C (Asynchronous)	A2C (Synchronous)
✔ Global update per actor rollout	✘ Global update bottlenecked by slowest actor
✘ Local gradient computation - GPU underutilized	✔ Batched gradient computation - GPU scalable
✘ Bandwidth Limited - Transfer massive gradients	✔ Better bandwidth utilization - Transfer trajectories
✘ Unstable - Policy lag due to asynchronous updates	✔ Stable - No Policy Lag

$$\left.\begin{aligned}\phi_1^\star &\leftarrow \phi^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_1^\star &\leftarrow \theta^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_1, \theta_1}$$

Global update for A1

$$\left.\begin{aligned}\phi_2^\star &\leftarrow \phi_1^\star + \delta_t \nabla_\phi V_\phi(s_t)\\ \theta_2^\star &\leftarrow \theta_1^\star + \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\end{aligned}\right|_{\phi_2, \theta_2} \quad \cdots$$

Global update for A2

Gradients computed for $\phi^\star, \theta^\star$

But update applied to $\phi_1^\star, \theta_1^\star$

IMPALA | V-Trace

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

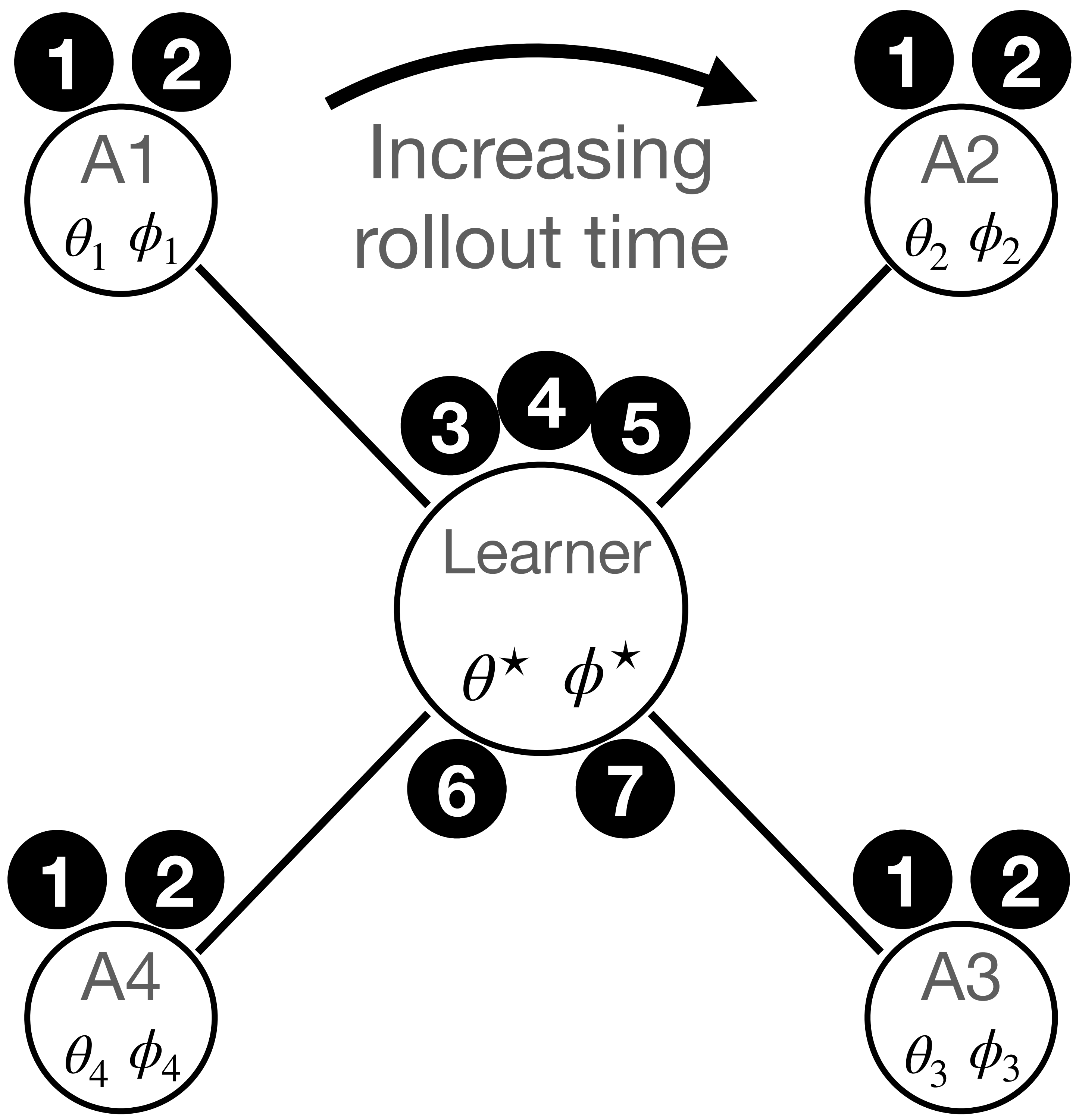
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



IMPALA | V-Trace

❶ Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

❷ Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

❸ Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

❹ Value Gradient $\nabla_\phi V_\phi(s_t)$

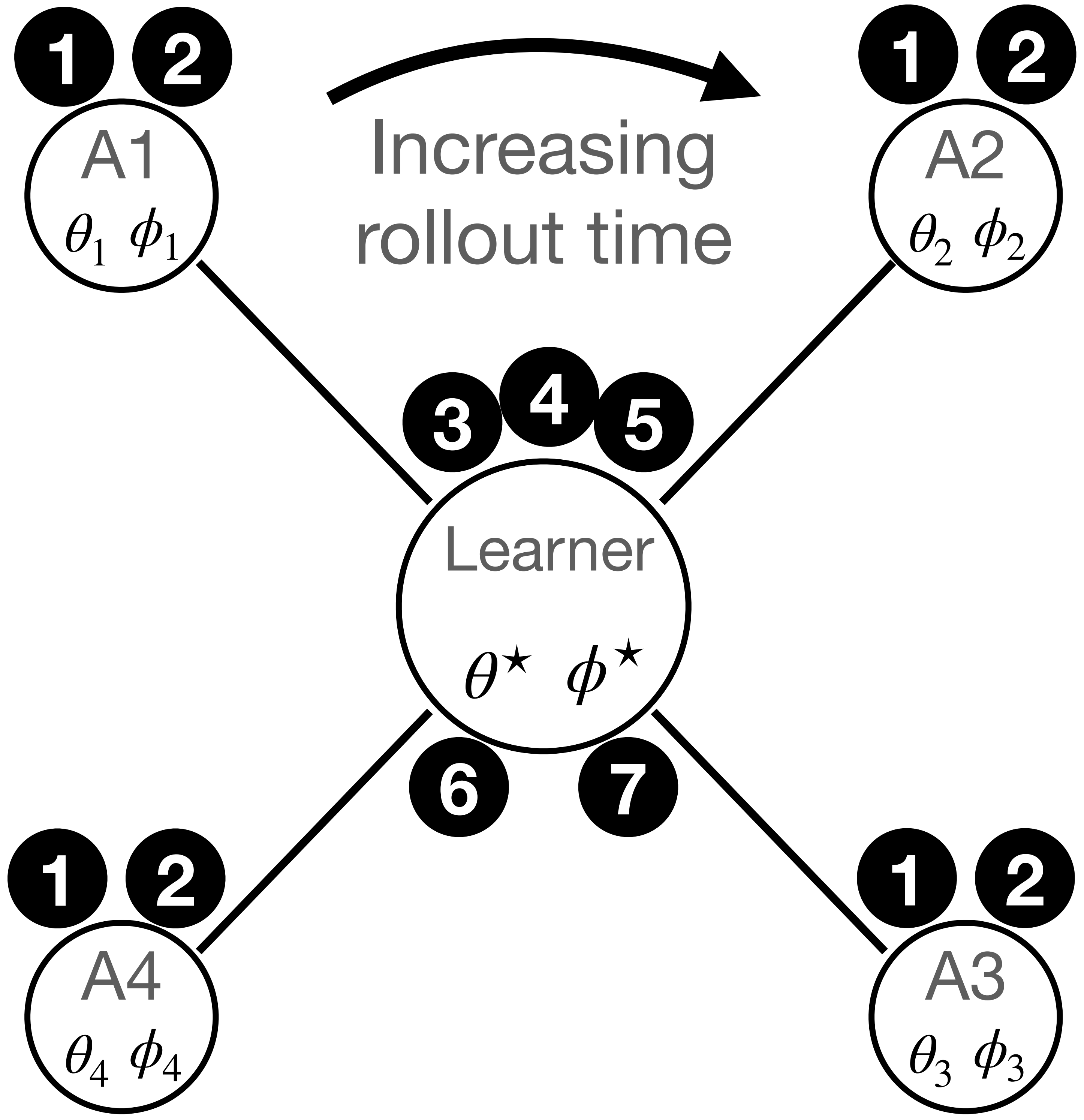
❺ Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

❻ Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

❼ Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Looks similar to A2C but…..

$\theta_i \neq \theta^\star$ $\phi_i \neq \phi^\star$

IMPALA | V-Trace

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

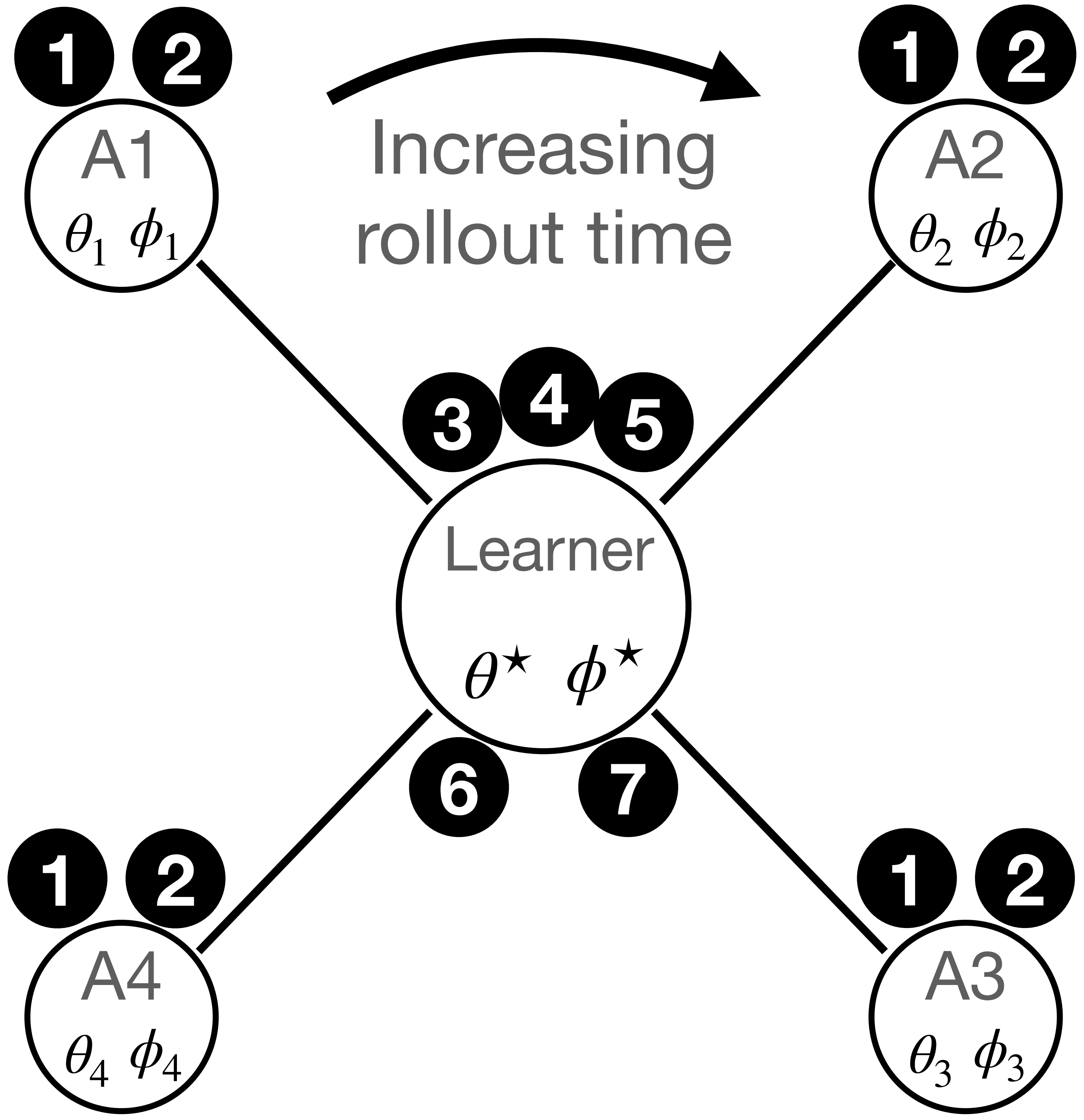
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Looks similar to A2C but…..

$\theta_i \neq \theta^\star$ $\phi_i \neq \phi^\star$

We have trajectory from θ_i but we need to update $\theta^\star$

IMPALA | V-Trace

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

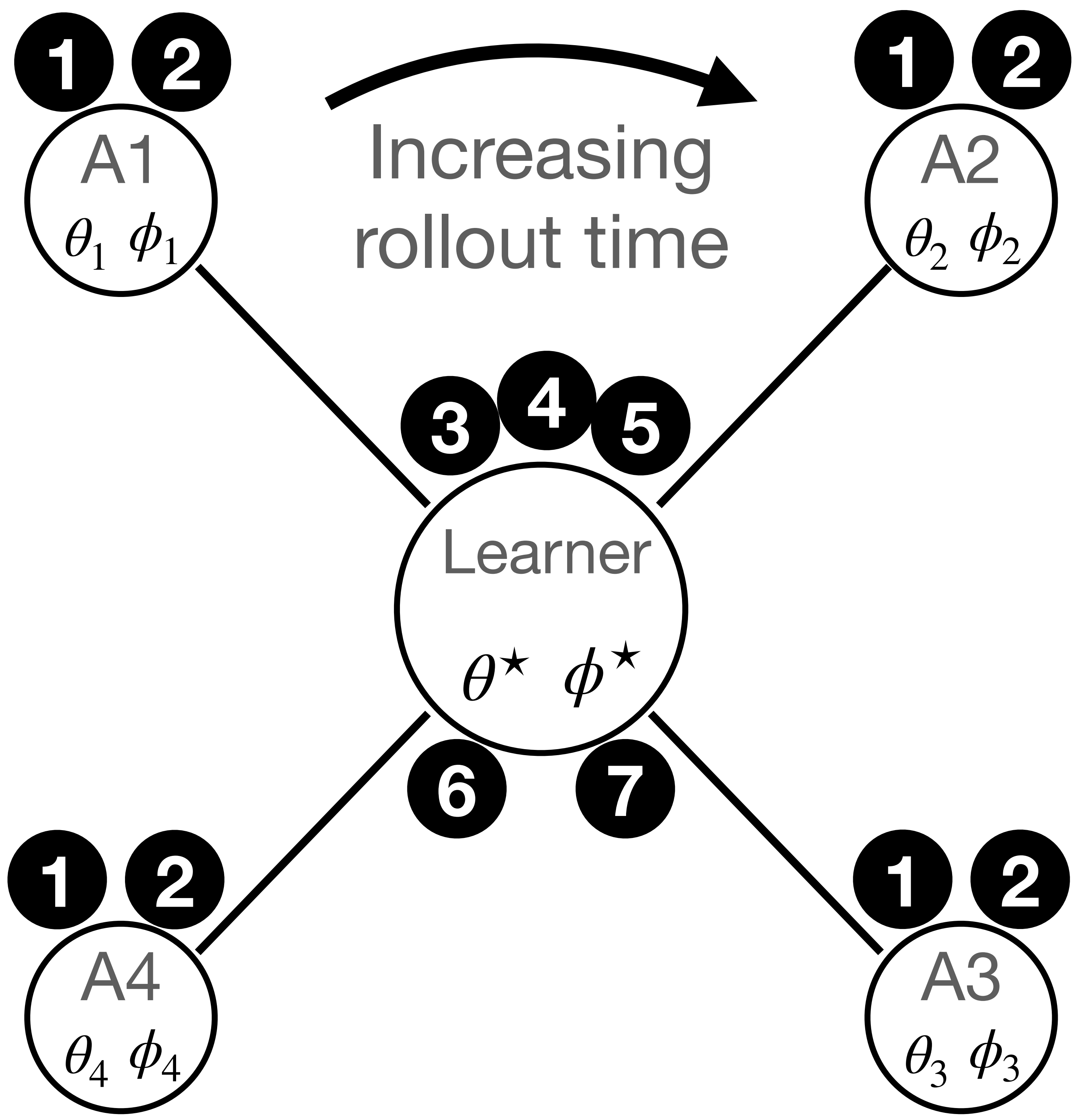
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Looks similar to A2C but…..

$\theta_i \neq \theta^\star$ $\phi_i \neq \phi^\star$

We have trajectory from θ_i but we need to update $\theta^\star$

How do we do this?

IMPALA | V-Trace

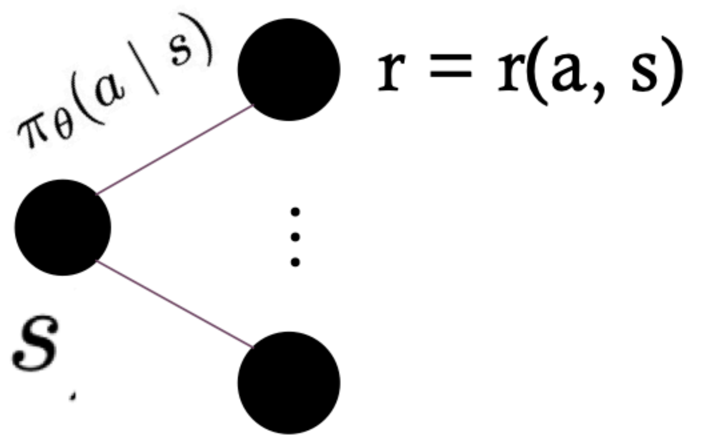
CHANGE OF VARIABLE – IMPORTANCE SAMPLING

Given two densities p and q:

$$\mathbb{E}_{x\sim p}[f(x)] = \sum_{x\in\mathcal{X}} p(x)f(x) = \sum_{x\in\mathcal{X}} q(x)\frac{p(x)}{q(x)}f(x) = \mathbb{E}_{\boxed{x\sim q}}\left[\frac{p(x)}{q(x)}f(x)\right]$$

Example: Single transition following π_θ, but sampled from π_θ:

$$J(\pi_{\theta'}) = \mathbb{E}_{(s,a,r)\sim\pi_{\theta'}}[r(\cdot,s)|s] = \sum_{a\in\mathcal{A}} \pi_{\theta'}(a|s)r(a,s) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s)\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)}r(a,s) = \mathbb{E}_{\boxed{(s,a,r)\sim\pi_\theta}}[\frac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)}r(\cdot,s)|s]$$



Lecture 16 - Slide 24

How do we do this?

$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$

IMPALA | V-Trace

CHANGE OF VARIABLE – IMPORTANCE SAMPLING

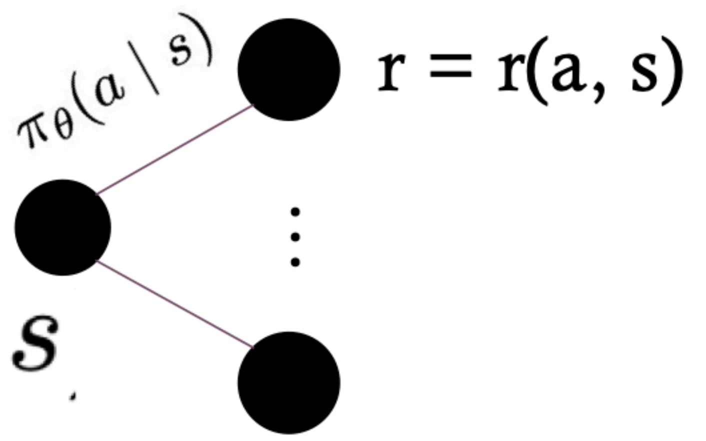
Given two densities p and q:

$$\mathbb{E}_{x\sim p}[f(x)] = \sum_{x\in\mathcal{X}} p(x)f(x) = \sum_{x\in\mathcal{X}} q(x)\frac{p(x)}{q(x)}f(x) = \mathbb{E}_{\boxed{x\sim q}}\left[\frac{p(x)}{q(x)}f(x)\right]$$

We can borrow **Importance Sampling** from TRPO and PPO (Kahn et al., 1953)

Example: Single transition following π_θ, but sampled from π_θ:

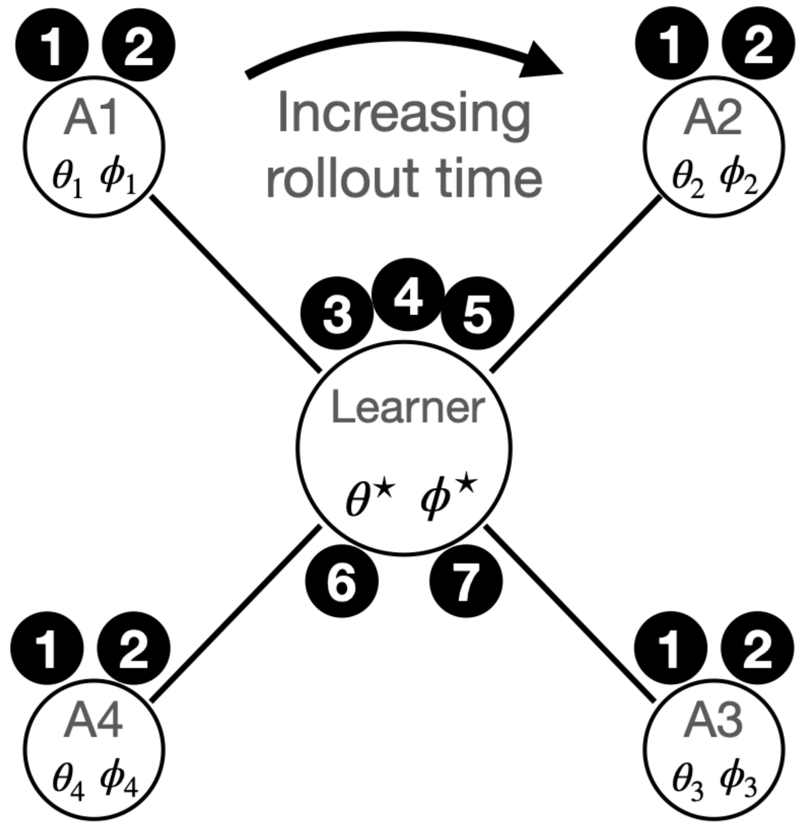
$$J(\pi_{\theta'}) = \mathbb{E}_{(s,a,r)\sim\pi_{\theta'}}[r(\cdot,s)|s] = \sum_{a\in\mathcal{A}} \pi_{\theta'}(a|s)r(a,s) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s)\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)}r(a,s) = \mathbb{E}_{\boxed{(s,a,r)\sim\pi_\theta}}[\frac{\pi_{\theta'}(\cdot|s)}{\pi_\theta(\cdot|s)}r(\cdot,s)|s]$$



Lecture 16 - Slide 24

How do we do this?

IMPALA | V-Trace | TD(0)



1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

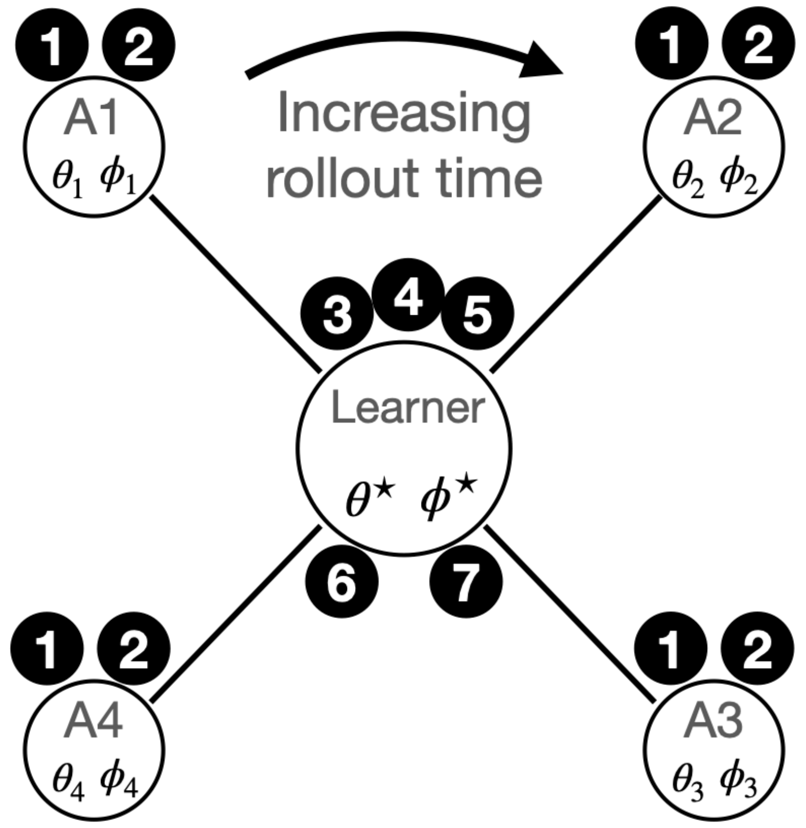
6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace | TD(0)



1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

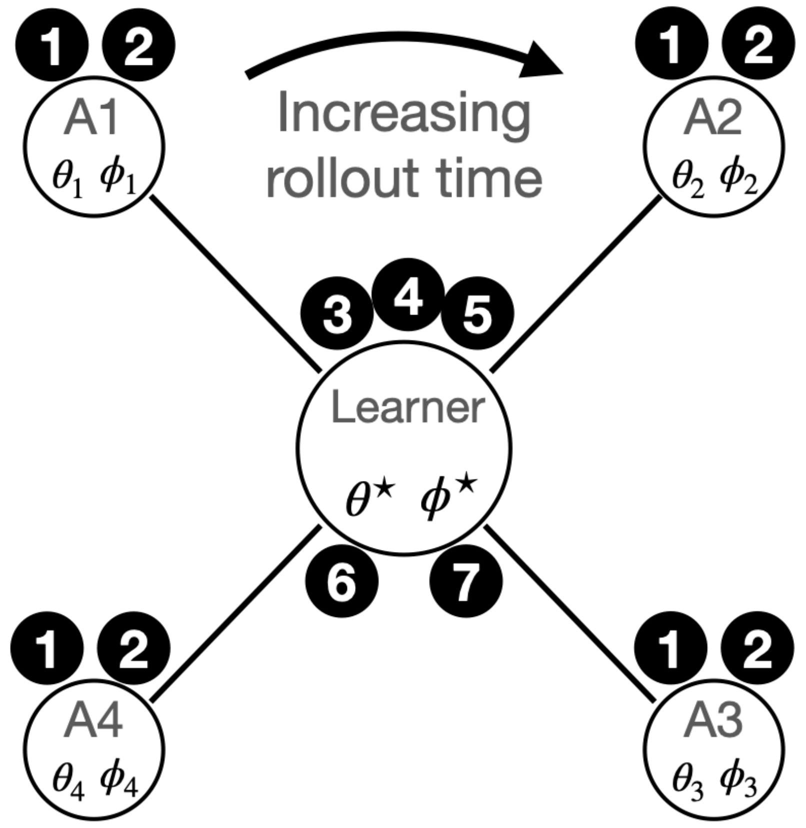
7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Bellman update

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

IMPALA | V-Trace | TD(0)



1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

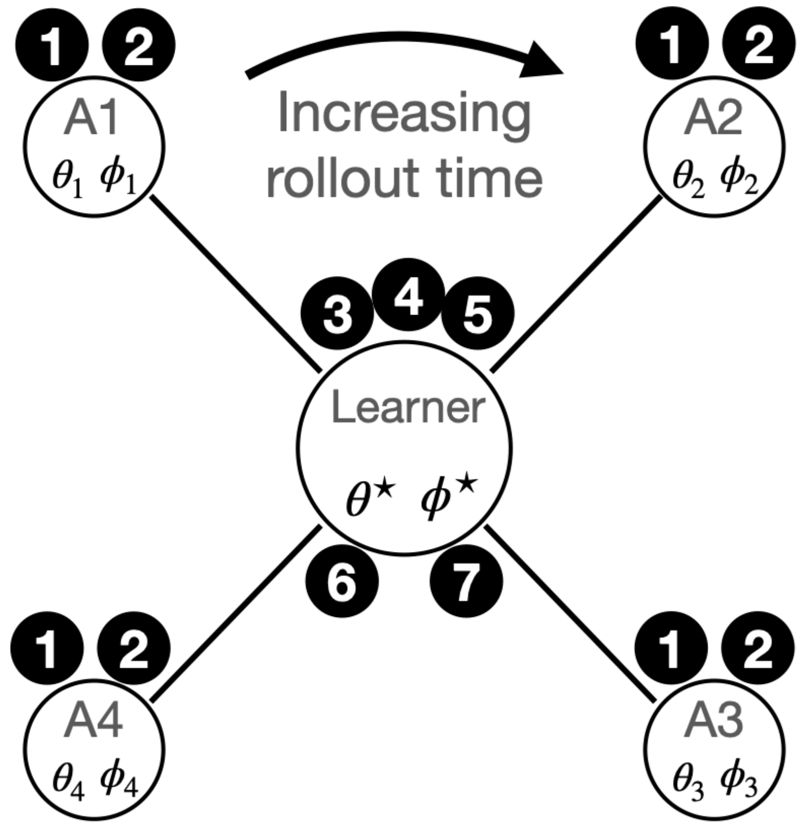
$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Bellman update

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

Trajectory obtained from θ_1

IMPALA | V-Trace | TD(0)



1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

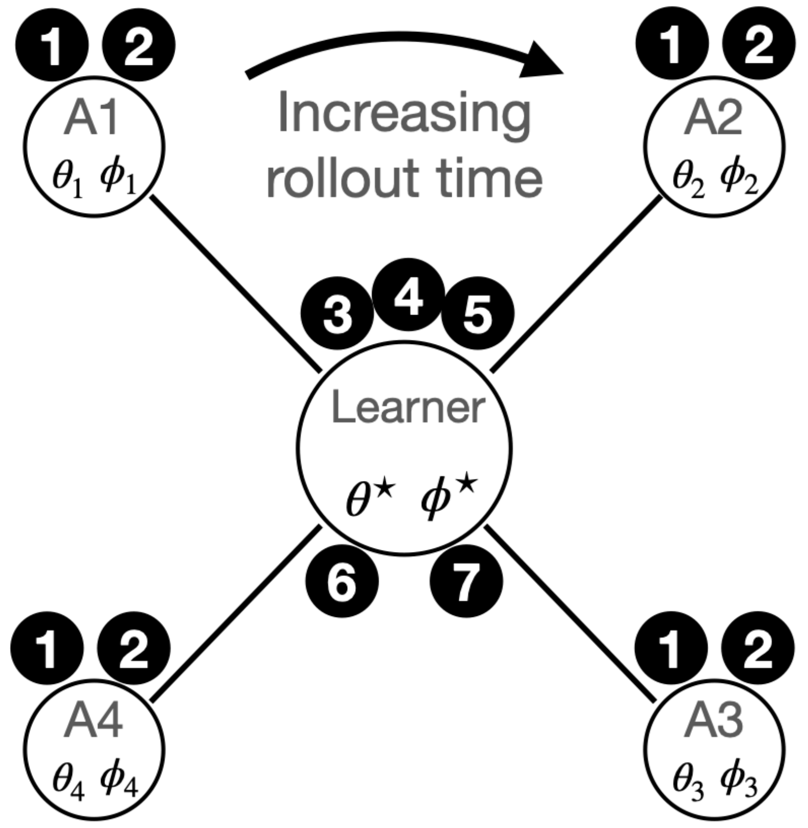
Bellman update

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

Trajectory obtained from θ_1

$$\rho_t \triangleq min(1, \frac{\pi_{\theta^\star}(a_t \mid s_t)}{\pi_{\theta_1}(a_t \mid s_t)})$$

IMPALA | V-Trace | TD(0)



1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Bellman update

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

Trajectory obtained from θ_1

$$\rho_t \triangleq min(1, \frac{\pi_{\theta^\star}(a_t \mid s_t)}{\pi_{\theta_1}(a_t \mid s_t)})$$

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times \rho_t \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

IMPALA | V-Trace | TD(0)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot\,|\,s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

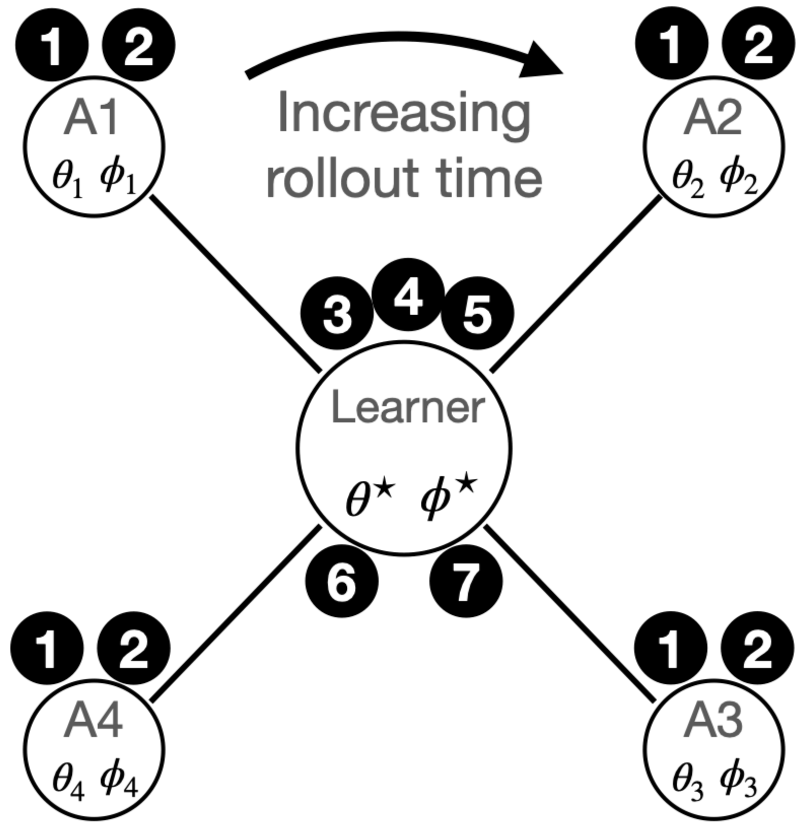
5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$



Bellman update

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

Trajectory obtained from θ_1

$$\rho_t \triangleq min(1, \frac{\pi_{\theta^\star}(a_t \mid s_t)}{\pi_{\theta_1}(a_t \mid s_t)})$$

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times \rho_t \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

$$\delta_t = \rho_t \times (r_{t+1} + \gamma V_{\phi^\star}(s_{t+1}) - V_{\phi^\star}(s_t))$$

V-Trace correction for TD(0)

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\,\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env\,.\,step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times \sum_{k=t}^{t+n} \gamma^{k-t} \times \rho_k \times (r_{k+1} + \gamma V_{\phi^\star}(s_{k+1}) - V_{\phi^\star}(s_k))$$

Naive V-Trace Bellman update for TD(n)

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times \sum_{k=t}^{t+n} \gamma^{k-t} \times \rho_k \times (r_{k+1} + \gamma V_{\phi^\star}(s_{k+1}) - V_{\phi^\star}(s_k))$$

Naive V-Trace Bellman update for TD(n)

$$\begin{aligned} V_{\phi^\star}(s_0) \leftarrow V_{\phi^\star}(s_0) + \alpha_{\phi^\star} \times (&\rho_0(r_1 + \gamma V_{\phi^\star}(s_1) - V_{\phi^\star}(s_0)) \\ &+\rho_1(r_2 + \gamma V_{\phi^\star}(s_2) - V_{\phi^\star}(s_1))) \end{aligned}$$

IMPALA | V-Trace | TD(n)

1 Sa

2 G

3 C

4 Va

5 P

6 U

Example

Bellman: $V(S0) = \boxed{p0{*}(r0 + (p1{*}r1 + p2{*}r2))}$

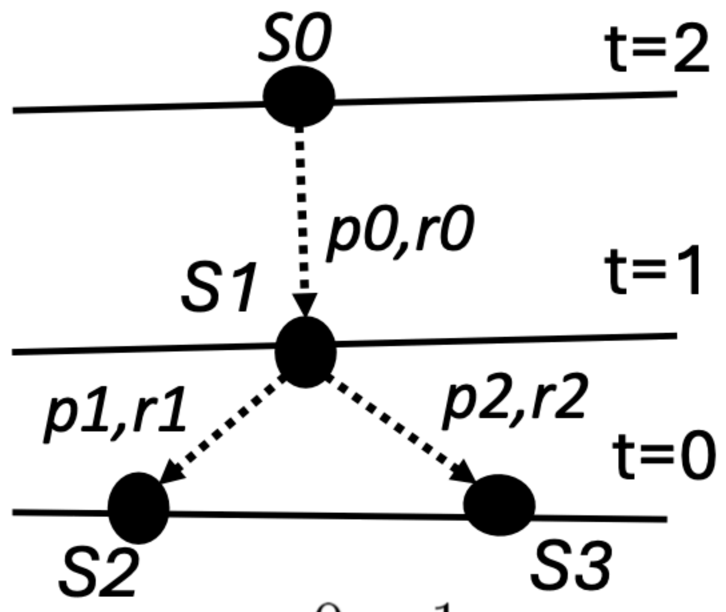
$$= p0{*}r0 + p0{*}p1{*}r1 + p0{*}p2{*}r2$$

Trajectories: $V(S0) = \boxed{p0{*}p1{*}(r0{+}r1) + p0{*}p2{*}(r0{+}r2)}$

$$= p0{*}p1{*}r0 + p0{*}p1{*}r1 + p0{*}p2{*}r0 + p0{*}p2{*}r2$$

$$= p0{*}r0{*}\underbrace{(p1{+}p2)}_{=1} + p0{*}p1{*}r1 + p0{*}p2{*}r2$$

$$= p0{*}r0 + p0{*}p1{*}r1 + p0{*}p2{*}r2$$



$p0 = 1$

$p1 + p2 = 1$

Unrolled MDP for H = 2, policy π

Lecture 7 - Slide 24

Intuition: rooted DAG $\rightarrow$ set of paths starting at root

Contribution of each edge to expected reward at *S0* is distributed over trajectories that contain it

$V_{\phi^\star}(s_k))$

7 Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace | TD(n)

1 S

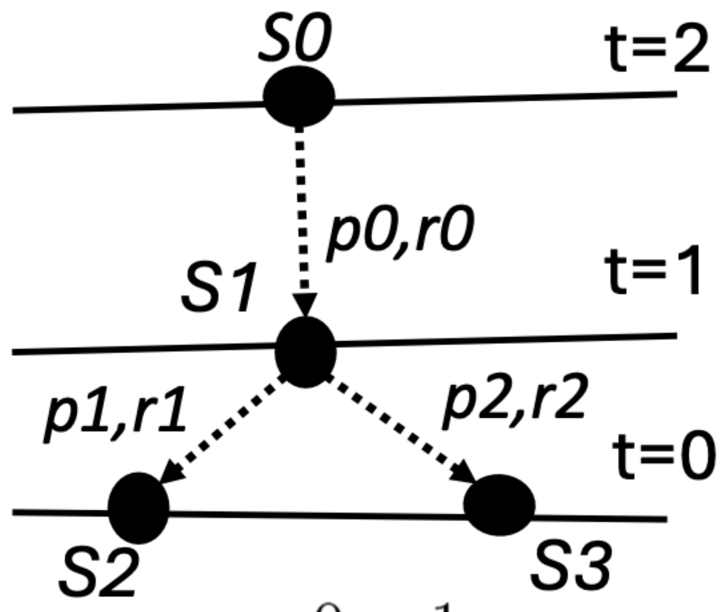
Example

Bellman:

$$V(S0) = \boxed{p0{*}(r0 + (p1{*}r1 + p2{*}r2))}$$
$$= p0{*}r0 + \boxed{p0{*}p1{*}r1} + p0{*}p2{*}r2$$

Trajectories:

$$V(S0) = \boxed{p0{*}p1{*}(r0{+}r1) + p0{*}p2{*}(r0{+}r2)}$$
$$= p0{*}p1{*}r0 + p0{*}p1{*}r1 + p0{*}p2{*}r0 + p0{*}p2{*}r2$$
$$= p0{*}r0{*}\underbrace{(p1{+}p2)}_{=1} + p0{*}p1{*}r1 + p0{*}p2{*}r2$$
$$= p0{*}r0 + p0{*}p1{*}r1 + p0{*}p2{*}r2$$



$p0 = 1$

$p1 + p2 = 1$

Unrolled MDP for H = 2, policy π

Reward contribution proportional to probability of its *Path*

We need to weight future samples (t + n) from t

Lecture 7 - Slide 24

Intuition: rooted DAG → set of paths starting at root

Contribution of each edge to expected reward at *S0* is distributed over trajectories that contain it

2 G

3 C

4 V

5 P

6 U

$V_{\phi^\star}(s_k))$

7 Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$\begin{aligned} V_{\phi^\star}(s_0) \leftarrow V_{\phi^\star}(s_0) + \alpha_{\phi^\star} \times (&\rho_0(r_1 + \gamma V_{\phi^\star}(s_1) - V_{\phi^\star}(s_0)) \\ + \quad &\rho_1(r_2 + \gamma V_{\phi^\star}(s_2) - V_{\phi^\star}(s_1))) \end{aligned}$$

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$V_{\phi^\star}(s_0) \leftarrow V_{\phi^\star}(s_0) + \alpha_{\phi^\star} \times (\rho_0(r_1 + \gamma V_{\phi^\star}(s_1) - V_{\phi^\star}(s_0)) + \rho_0 \rho_1(r_2 + \gamma V_{\phi^\star}(s_2) - V_{\phi^\star}(s_1)))$$

IMPALA | V-Trace | TD(n)

1. Sample an action with current policy for s_t

$$a_t \sim \pi_\theta(\cdot \mid s_t)$$

2. Get trajectory for sampled action a_t

$$s_t, a_t, s_{t+1}, r_{t+1} \leftarrow env.step(a_t, s_t)$$

3. Compute TD-Error δ_t

$$\delta_t = r_{t+1} + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$$

4. Value Gradient $\nabla_\phi V_\phi(s_t)$

5. Policy Gradient $\nabla_\theta \log \pi_\theta(a_t \mid s_t)$

6. Update Value parameters ϕ

$$\phi \leftarrow \phi + \alpha_\phi \delta_t \nabla_\phi V_\phi(s_t)$$

7. Update Policy parameters θ

$$\theta \leftarrow \theta + \alpha_\theta \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$V_{\phi^\star}(s_0) \leftarrow V_{\phi^\star}(s_0) + \alpha_{\phi^\star} \times (\rho_0(r_1 + \gamma V_{\phi^\star}(s_1) - V_{\phi^\star}(s_0)) + {\color{red}\rho_0}\rho_1(r_2 + \gamma V_{\phi^\star}(s_2) - V_{\phi^\star}(s_1)))$$

$$V_{\phi^\star}(s_t) \leftarrow V_{\phi^\star}(s_t) + \alpha_{\phi^\star} \times \sum_{k=t}^{t+n} \gamma^{k-t} \times (\prod_{i=t}^{k-1} \rho_i) \times \rho_k \times (r_{k+1} + \gamma V_{\phi^\star}(s_{k+1}) - V_{\phi^\star}(s_k))$$

Final V-Trace correction for TD(n)

IMPALA | Evaluation

Architecture	CPUs	GPUs[1]	FPS[2]	
Single-Machine			Task 1	Task 2
A3C 32 workers	64	0	6.5K	9K
Batched A2C (sync step)	48	0	9K	5K
Batched A2C (sync step)	48	1	13K	5.5K
Batched A2C (sync traj.)	48	0	16K	17.5K
Batched A2C (dyn. batch)	48	1	16K	13K
IMPALA 48 actors	48	0	17K	20.5K
IMPALA (dyn. batch) 48 actors[3]	48	1	21K	24K
Distributed				
A3C	200	0	46K	50K
IMPALA	150	1	80K	
IMPALA (optimised)	375	1	200K	
IMPALA (optimised) batch 128	500	1	250K	

[1] Nvidia P100 [2] In frames/sec (4 times the agent steps due to action repeat). [3] Limited by amount of rendering possible on a single machine.

Table 1. Throughput on `seekavoid_arena_01` (task 1) and `rooms_keys_doors_puzzle` (task 2) with the shallow model in Figure 3. The latter has variable length episodes and slow restarts. Batched A2C and IMPALA use batch size 32 if not otherwise mentioned.

Throughput Evaluation

	Task 1	Task 2	Task 3	Task 4	Task 5
Without Replay					
V-trace	46.8	32.9	**31.3**	**229.2**	**43.8**
1-Step	**51.8**	**35.9**	25.4	215.8	43.7
ε-correction	44.2	27.3	4.3	107.7	41.5
No-correction	40.3	29.1	5.0	94.9	16.1
With Replay					
V-trace	47.1	**35.8**	**34.5**	**250.8**	**46.9**
1-Step	**54.7**	34.4	26.4	204.8	41.6
ε-correction	30.4	30.2	3.9	101.5	37.6
No-correction	35.0	21.1	2.8	85.0	11.2

Tasks: `rooms_watermaze`, `rooms_keys_doors_puzzle`, `lasertag_three_opponents_small`, `explore_goal_locations_small`, `seekavoid_arena_01`

Table 2. Average final return over 3 best hyperparameters for different off-policy correction methods on 5 DeepMind Lab tasks. When the lag in policy is negligible both V-trace and 1-step importance sampling perform similarly well and better than ε-correction/No-correction. However, when the lag increases due to use of experience replay, V-trace performs better than all other methods in 4 out 5 tasks.

Policy Lag Correction Evaluation

IMPALA | Evaluation

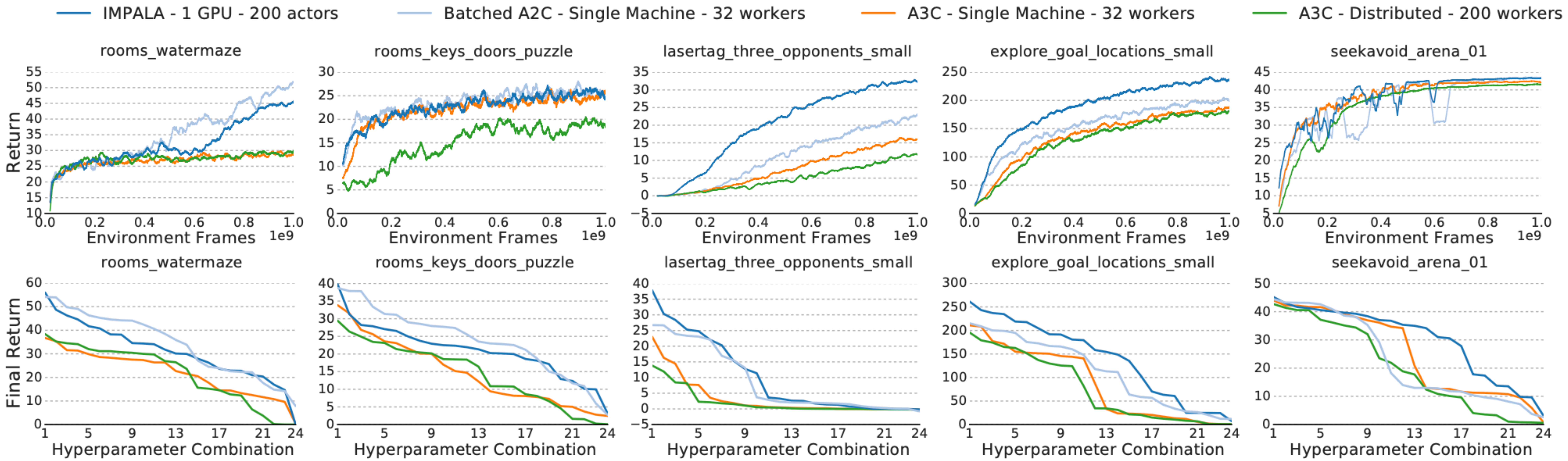


Figure 4. **Top Row:** Single task training on 5 DeepMind Lab tasks. Each curve is the mean of the best 3 runs based on final return. IMPALA achieves better performance than A3C. **Bottom Row:** Stability across hyperparameter combinations sorted by the final performance across different hyperparameter combinations. IMPALA is consistently more stable than A3C.

Stability Evaluation