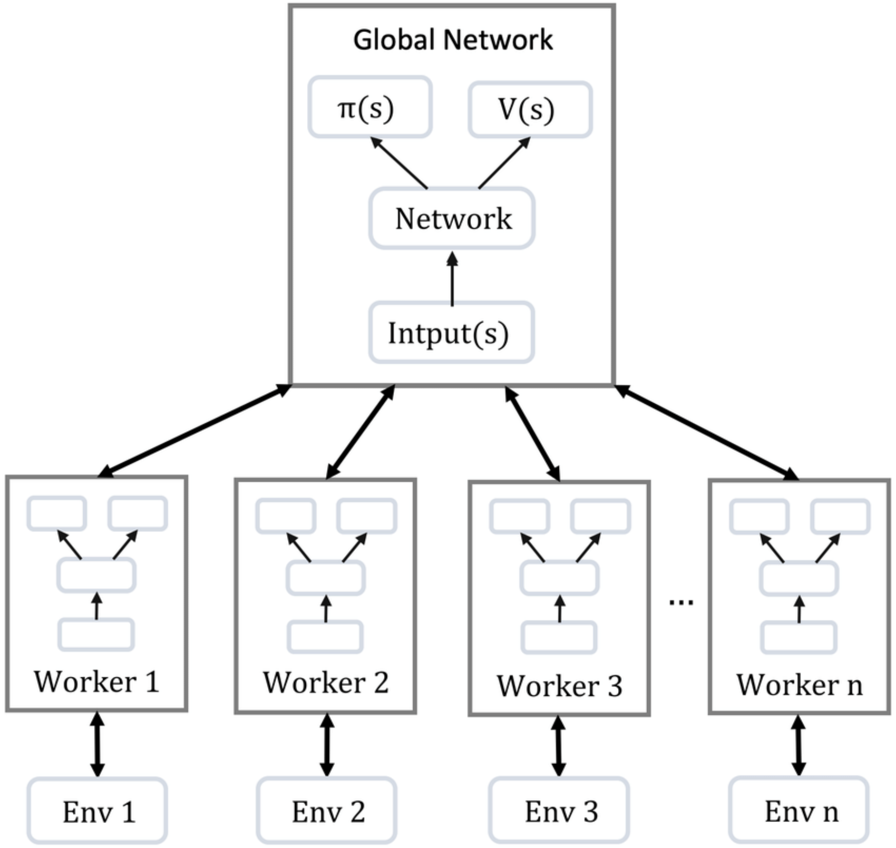
A3C 1
Reinforcement Learning Review

Based on Prof. Oh's Reinforcement Learning Lectures

Suinne Lee

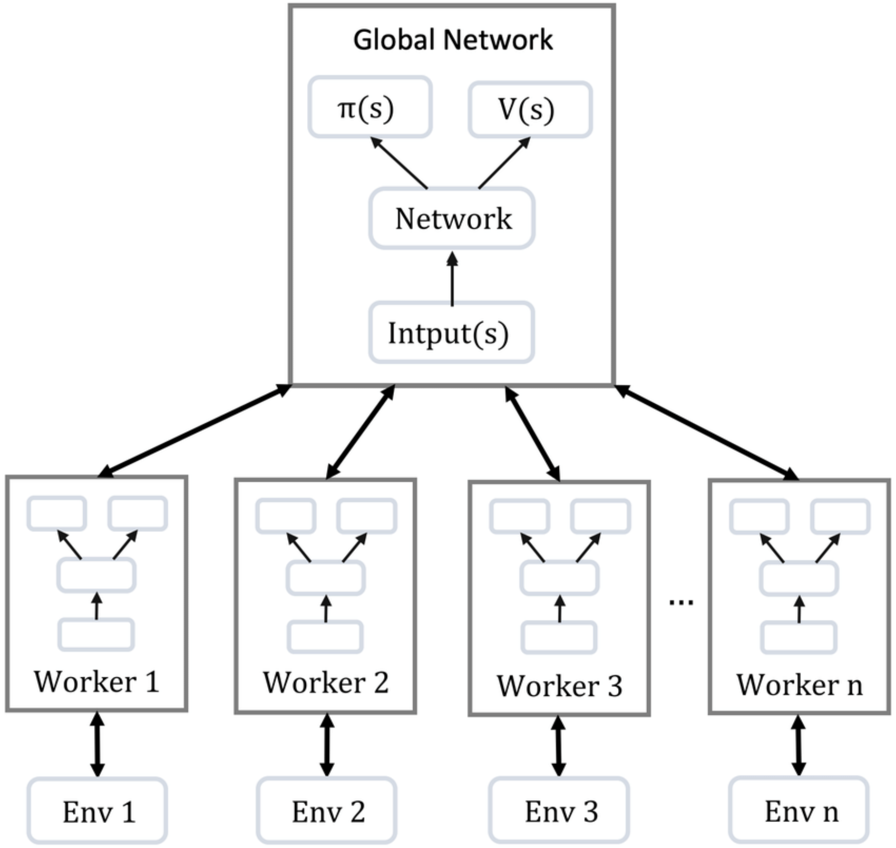
A3C (Asynchronous Advantage Actor-Critic)

- Utilizes multiple networks: global + multiple worker agents.
- The multiple networks share the same architecture. They compute on their own then incorporate everything into the global network.
- Each agent acts in its own state and acquires its own trajectory throughout time. → Multiple instances.
- (Same network, same environment, different instances!)



A3C (Asynchronous Advantage Actor-Critic)

- Parameters are updated on the global network… ASYNCHRONOUSLY!!



Synchronous vs Asynchronous?

- Synchronous update: each agent learns with their own policy. The parameters are updated simultaneously for all agents.
- Asynchronous: They are updated individually at the end of each agent’s minibatch gradient descent

A3C In a Nutshell

- Multiple networks (namely global and worker networks) with its own copy of the environment (each other actor-critic networks)
- Asynchronous update

A3C - Advantages

- No need for experience replay (the multiple agent system reduces overall temporal correlation)
- Recall: actor – policy, critic – value function.
  - We can use V(s), Q(s,a) and A(s,a) for the value function!
- In order to evaluate the advantage we need two networks: Q(s,a) and V(s).
- But we instead use n-step return G_t^n in place of Q. (Doesn't require a network, it is automatically computed from the trajectory)
- $A(s,a) \approx G_t^n - V(s)$

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$$

To estimate $q_\pi(s,a)$, at first/every time-step t that the state s was visited and the action a was selected in an episode,

- increment count: $n(s,a) \leftarrow n(s,a) + 1$ (for all episodes experienced)
- increment total return: $S(s,a) \leftarrow S(s,a) + G_t$
- value of (s,a) is estimated by mean return: $Q(s,a) = \frac{S(s,a)}{n(s,a)}$

A3C - Advantages

- Out of the two very important DRL algorithms, A3C also works with a continuous action space

	DQN	A3C
state space	disc/conti	disc/conti
action	disc	disc/conti

- Also…. is fast.
- Runs on a CPU (doesn’t need a GPU)
- On-policy RL is possible.

On-Policy: target policy = behavior policy

Off-Policy: target policy $\pi \neq$ behavior policy μ

A3C – Advantages in a nutshell

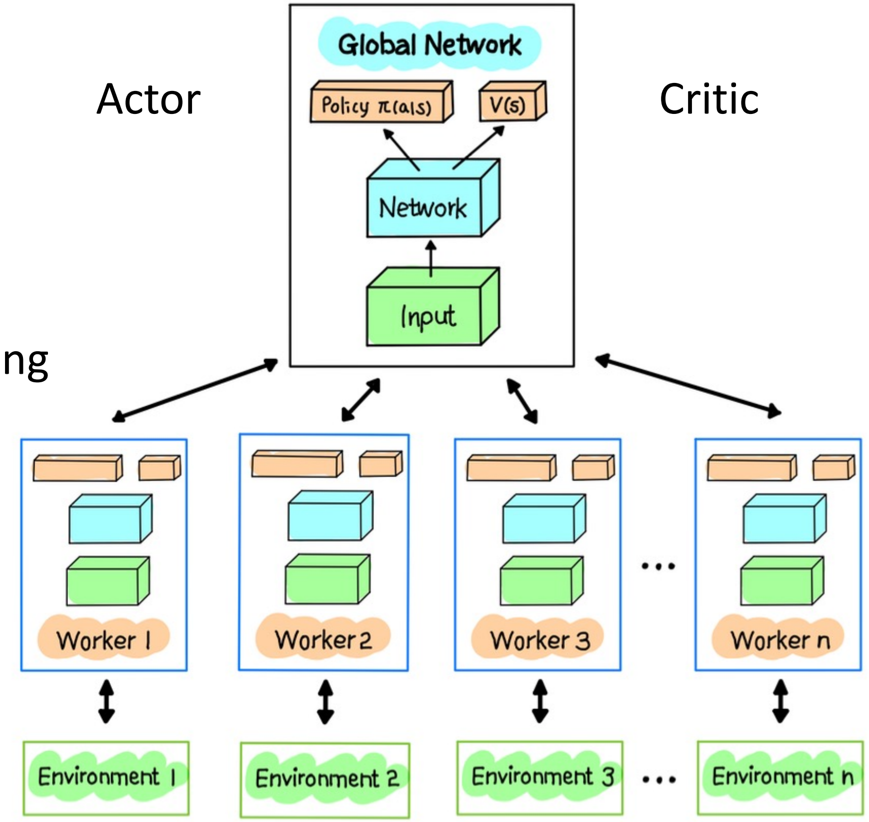
- Less temporal correlation (no need for experience replay)
- Using G-V for advantage allows us to perform with only one network for V!
- Works for a continous action space
- Fast, efficient

A3C – how does it work?

Actor

Critic

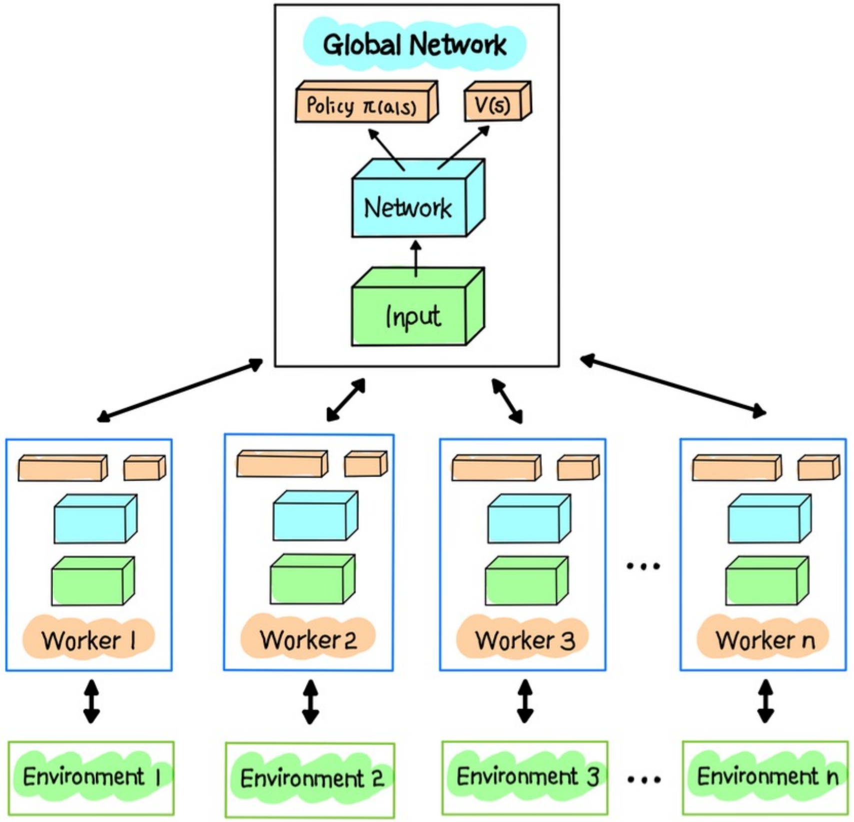
Worker networks: only used in training

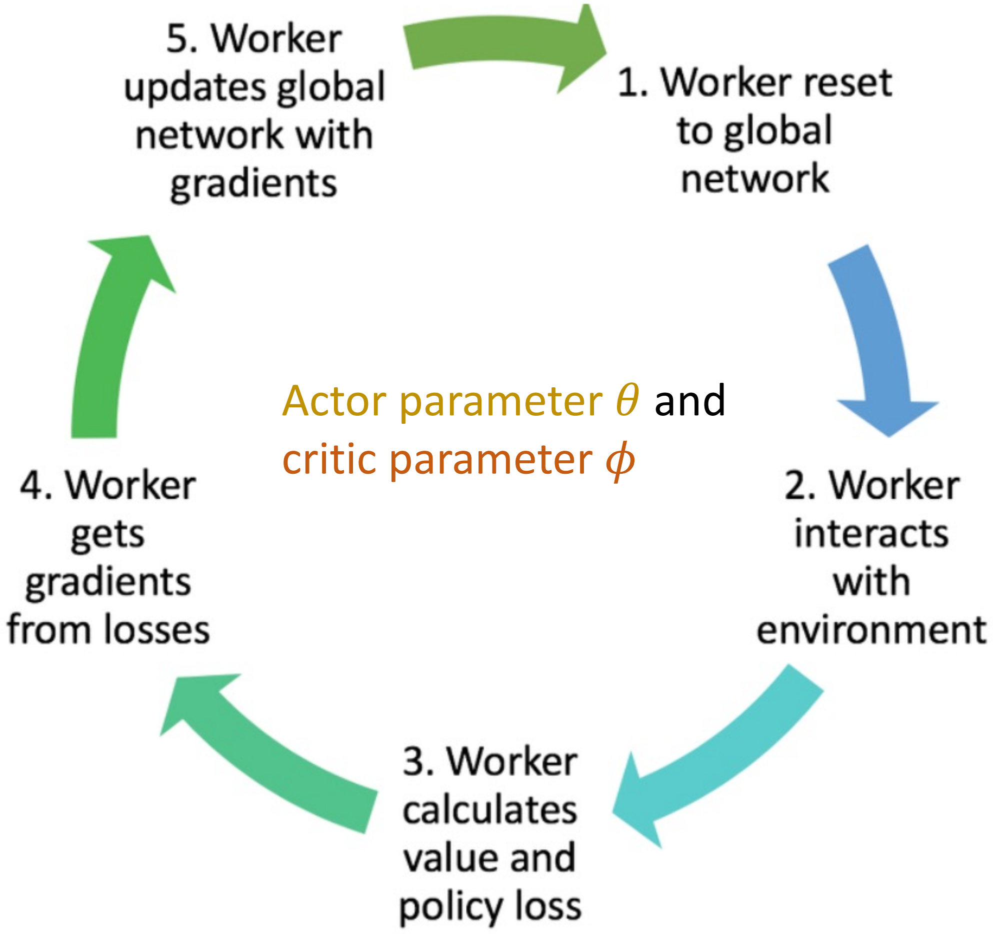


Each worker interacts with its copy of the environment!

Worker networks accululate gradients and then update (asynchronously)

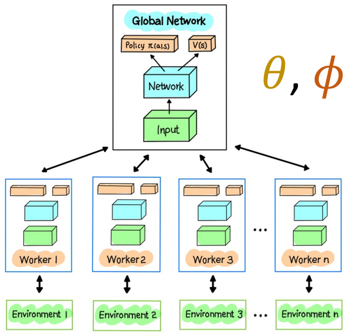
A3C – how does it work?



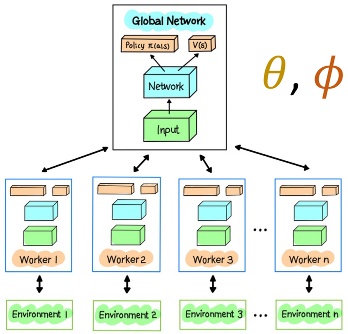


A3C – how does it work?

Actor parameter θ and critic parameter ϕ



θ, ϕ



θ, ϕ

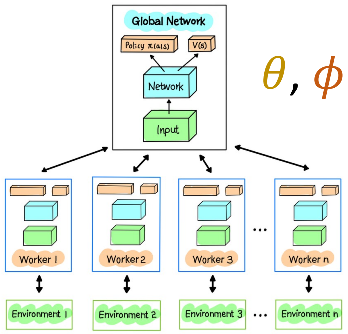
θ_1, ϕ_1 θ_2, ϕ_2 θ_3, ϕ_3 θ_4, ϕ_4

- During $t_{\max}$ steps, each agent computes an accumulated gradient (in its own process) and finally updates the shared parameters.

  Agent Actor: $\Delta\theta \leftarrow \Delta\theta + (G_t^{(n)} - V_{\phi'}(s_t))\, \nabla_{\theta'} \log \pi_{\theta'}(a_t \mid s_t)$

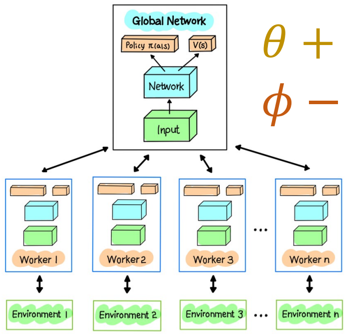
  Agent Critic: $\Delta\phi \leftarrow \Delta\phi - (G_t^{(n)} - V_{\phi'}(s_t))\, \nabla_{\phi'} V_{\phi'}(s_t)$

  Global network: $\theta \leftarrow \theta + \alpha\Delta\theta$ and $\phi \leftarrow \phi - \beta\Delta\phi$



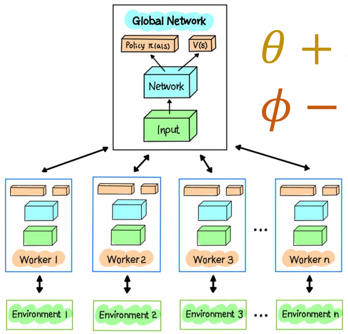
θ, ϕ

$\theta_1 + \alpha\nabla\theta_1,$ $\phi_1 - \beta\nabla\phi_1$ θ_2, ϕ_2 θ_3, ϕ_3 θ_4, ϕ_4



$\theta + \alpha\nabla\theta_1,$ $\phi - \beta\nabla\phi_1$

$\theta_1 + \alpha\nabla\theta_1,$ $\phi_1 - \beta\nabla\phi_1$ θ_2, ϕ_2 θ_3, ϕ_3 θ_4, ϕ_4



$\theta + \alpha\nabla\theta_1,$ $\phi - \beta\nabla\phi_1$

$\boldsymbol{\theta} + \alpha\nabla\theta_1,$ $\boldsymbol{\phi} - \beta\nabla\phi_1$ θ_2, ϕ_2 θ_3, ϕ_3 θ_4, ϕ_4

A3C – how does it work?