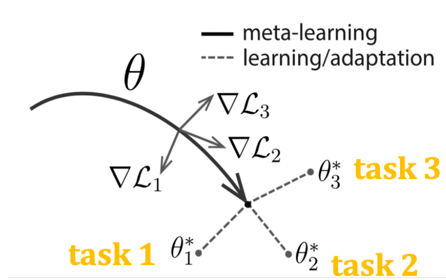
1 Multi-agent Markov Decision Process

The package routing problem can be formulated as a multi-agent Markov decision process. Let n denote the number of routers in the network. A multi-agent Markov decision process is characterized by a tuple $M =< S, A^i, P, R >$ where S is the global state space shared by all the routers, A^i is the set of actions that router i can execute. Let $A = \Pi_{i=1}^n A^i$ be the joint action space of all the routers. $P : S \times A \times S \rightarrow [0, 1]$ is a state transition probability and $R : S \times A \rightarrow R$ is the shared reward function. Each router has a policy π_i representing the probability of choosing action a^i at state s_t. The policy π_i is approximated by a parameterized function $\pi^i_{\theta_i}$. Let $\pi_\theta = \Pi_{i=1}^n \pi^i_{\theta_i}$ be the joint policy of all the routers. The objective of the routers is to find the optimal policy parameters that maximize the expected reward, which is given by $J(\theta)$

$$J(\pi_\theta) = E_{x\sim d_{\pi_\theta}, a_i \sim \pi^i_{\theta_i}} \left[\pi(s, a_1, \cdots, a_n) R(x, a)\right], \tag{1}$$

where d_{π_θ} is the stationary distribution of the Markov chain under policy π_θ.

In summary, both state and reward are global and shared by all the routers. Action is private. Is it possible to only use local state or reward?



2 Algorithm

2.1 MAML

Model-agnostic meta reinforcement learning is one kind of meta RL algorithm which achieves good generalization performance on new tasks by updating policy model parameters. As shown in Figure 1, a good policy model parameter θ should be close to all the optimal parameters of the tasks which makes θ the best parameters initialization that can quickly adapt to different new tasks. When we have, for example, three different new tasks 1, 2 and 3, a gradient step is taken for each task (the gray lines). We can see that the parameters are close to all the 3 optimal parameters of task 1, 2, and 3 which makes θ the best parameters initialization that can quickly adapt to different new tasks.

As a result, only a small change in the parameters θ will lead to an optimal minimization of the loss function of any task.

2.2 Trust Region Policy Optimization

TRPO algorithm limits the parameter changes (policy changes).

$$\begin{aligned} \max \quad & E_{s\sim d_{\theta_{\text{old}}}, a_i \sim \pi^i_{\theta_{i\text{old}}}} \left[\frac{\pi_\theta(a_1, \cdots, a_n|s)}{\pi_{\theta_{\text{old}}}(a_1, \cdots, a_n|s)} A_{\pi_{\theta_{\text{old}}}}(s, a_1, \cdots, a_n) \right] \\ \text{s.t.} \quad & E_{s\sim d_{\theta_{\text{old}}}} \left[D_{KL}(\pi^i_{\theta_{i\text{old}}}(\cdot|s) || \pi^i_{\theta_i}(\cdot|s)) \right] \leq \delta, \quad \forall i \in [1, n], \end{aligned} \tag{2}$$

where $\pi_\theta(a_1, \cdots, a_n|s) = \Pi_{i=1}^n \pi^i_{\theta_i}(a_i|s)$. The solution of Equation (2) can be found in Appendix C in "Trust Region Policy Optimization".

2.3 Multi-agent MAML

Algorithm 1: Multi-agent MAML algorithm

Require: $p(\tau)$: distributions of tasks;
Require: α: step size hyper-parameter;
randomly initialize $\theta_i, i \in [1, n]$;
while *not done* **do**
  sample batch of tasks $\tau_i \sim p(\tau)$;
  for *all* τ_i **do**
    Sample K trajectories $D = \{(x_1, a_1, r_1, x_2, ..., x_H)\}$ using $\pi_{\theta_i}, i \in [1, n]$ in τ_i;
    Evaluate $\nabla_{\theta_i} J(\pi_\theta)$ using D and $J(\pi_\theta)$ in Equation (1);
    Compute adapted parameters with gradient descent: $\theta'_i = \theta_i - \alpha \nabla_{\theta_i} J(\pi_\theta)$;
    Sample K trajectories $D'_i = \{(x_1, a_1, r_1, x_2, ..., x_H)\}$ using $\pi^i_{\theta'_i}, i \in [1, n]$ in τ_i;
  end
  Update θ according to D'_i and Equation (2)
end