Estimating Policy Functions in Payments Systems using Deep Reinforcement Learning

Ajit Desai[1], Han Du[1], Rodney Garratt[2], Francisco Rivadeneyra[1], Pablo Samuel Castro[3]

[1]Bank of Canada, [2]University of California Santa Barbara, [3]Google Research, Brain Team

Bank of Canada

Banque du Canada

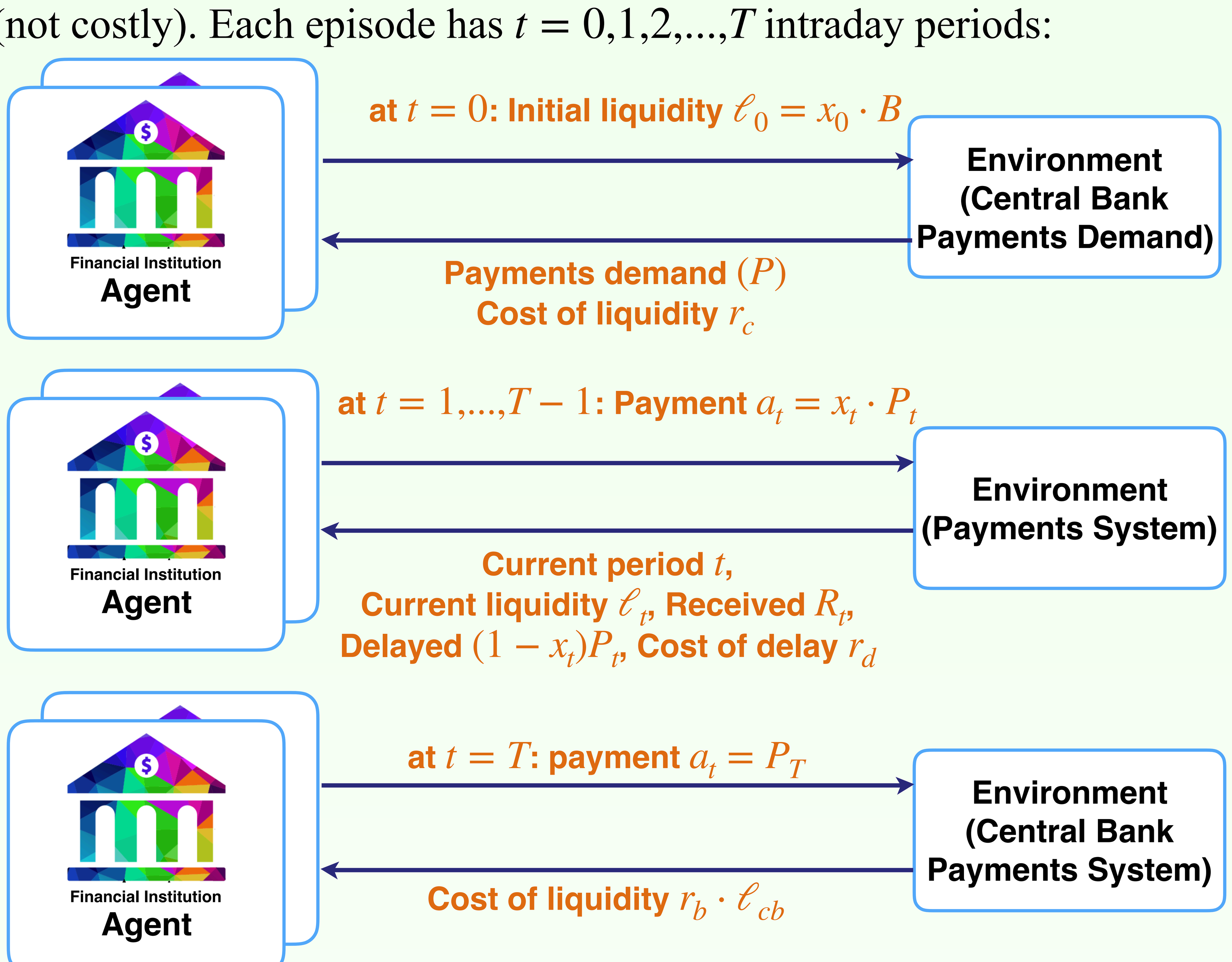
We demonstrate the applicability of deep reinforcement learning (DRL) to estimate best-response functions in a high-value payments system (HVPS), a real-world strategic game.

Objective: Approximate the liquidity management rules of Canada's HVPS participants using DRL. The objective of the agents is to learn:

1. The optimal initial liquidity policy
2. The optimal intraday payments policy

Environment

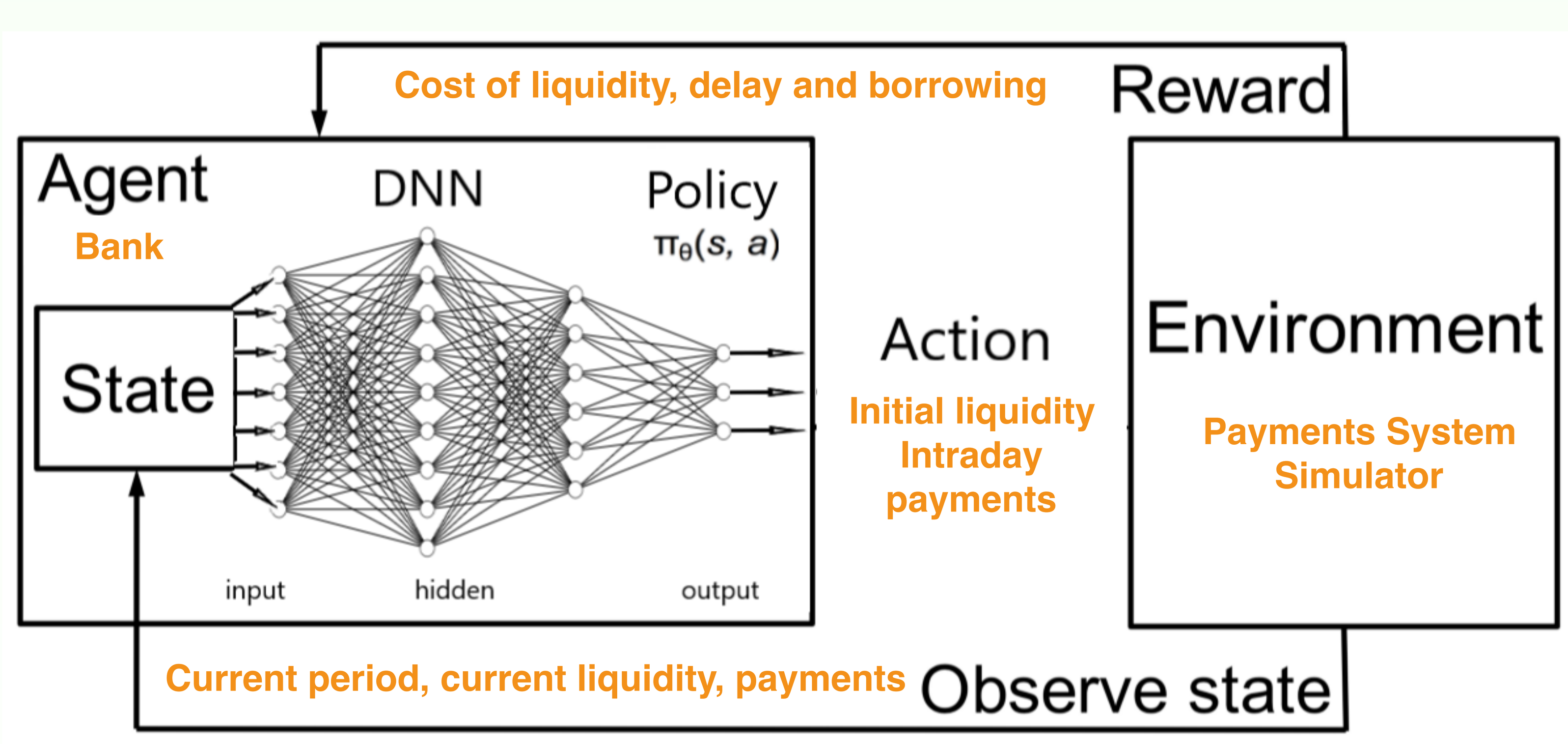
Every day (episode) the agent's objective is to satisfy exogenous payments demand using initial liquidity (costly) or received payments (not costly). Each episode has $t = 0,1,2,...,T$ intraday periods:



Reward $\mathscr{R} = r_c\ell_0 + \sum_{t=1}^{T-1} P_t(1-x_t)\cdot r_d + r_b\ell_{cb}$

Methodology

Deep Reinforcement Learning: train agents, parameterized using a deep neural network, to behave optimally in a sequential decision task through interaction with the environment.



We employ the **vanilla policy gradient** (VPG) method.

A policy π_θ with parameters θ, $\mathscr{R}(s_t, a_t)$ the cost of executing the action a_t from the state s_t and τ is trajectory of state-action-reward tuple. The function $J(\pi_\theta)$ the expected finite-horizon undiscounted cost.

- The gradient of the cost function:

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\mathscr{R}(s_t, a_t)$$

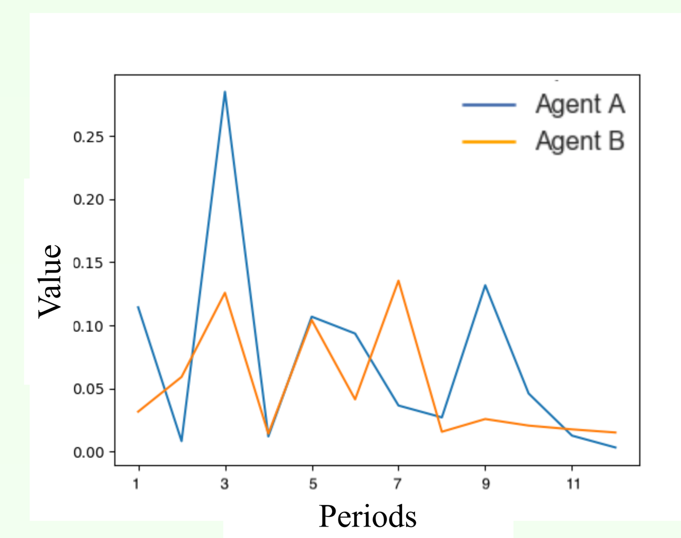
- The policy parameters θ are updated after each episode e:

$$\theta_{e+1} = \theta_e + \alpha \nabla_\theta J(\pi_\theta)$$

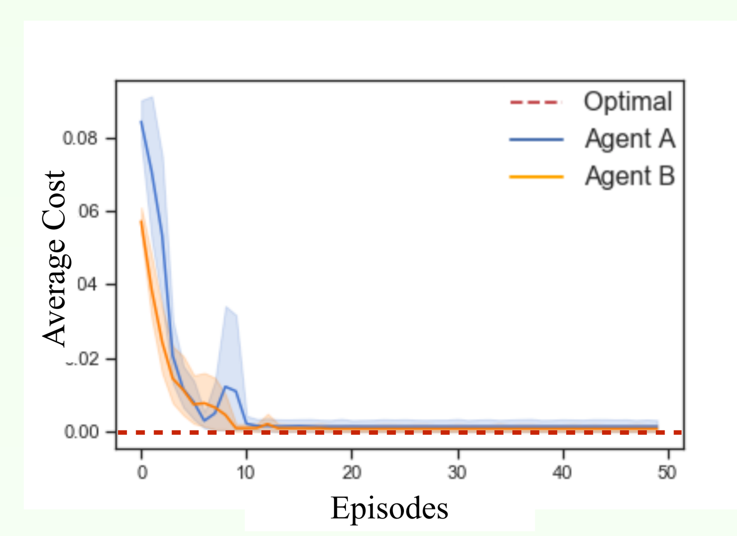
Results

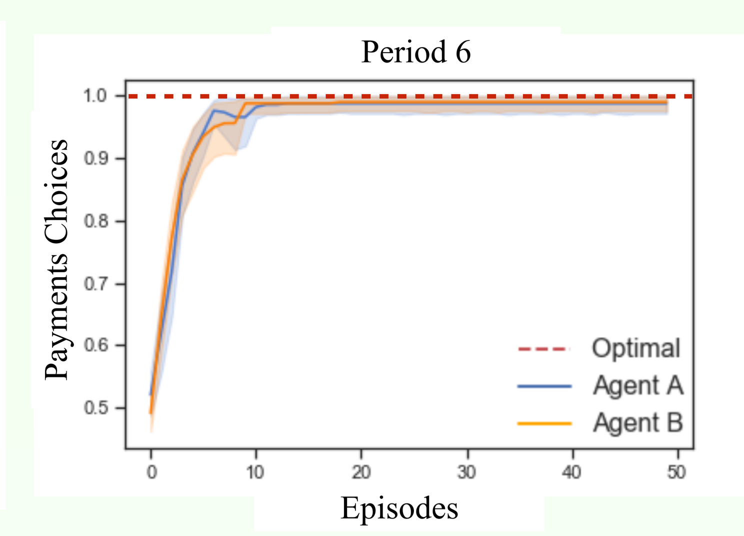
1: **Intraday payment decision**: Without incentives for the delay, an agent learns to send as much as it can in every period.

- Unlimited free initial liquidity
- State space = [current period (t), initial liquidity (ℓ_0), current period payments demand (P_t) and liquidity (ℓ_t)]
- Action space is the payment choice ($x_t \cdot P_t$), where $x_t \in [0,1]$ discretized into 11 buckets
- Per-episode total cost: $\mathscr{R} = \sum_t^{T-1} P_t(1-x_t)\cdot r_d$



Payments demands used in the simulation

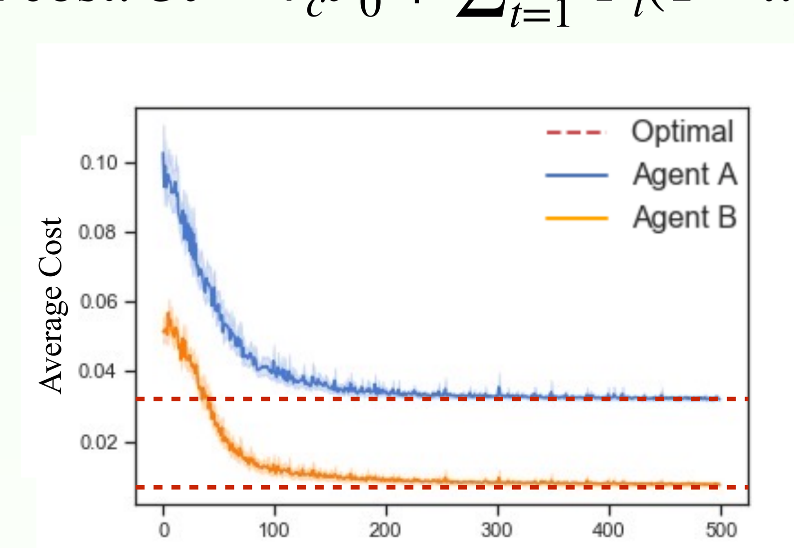




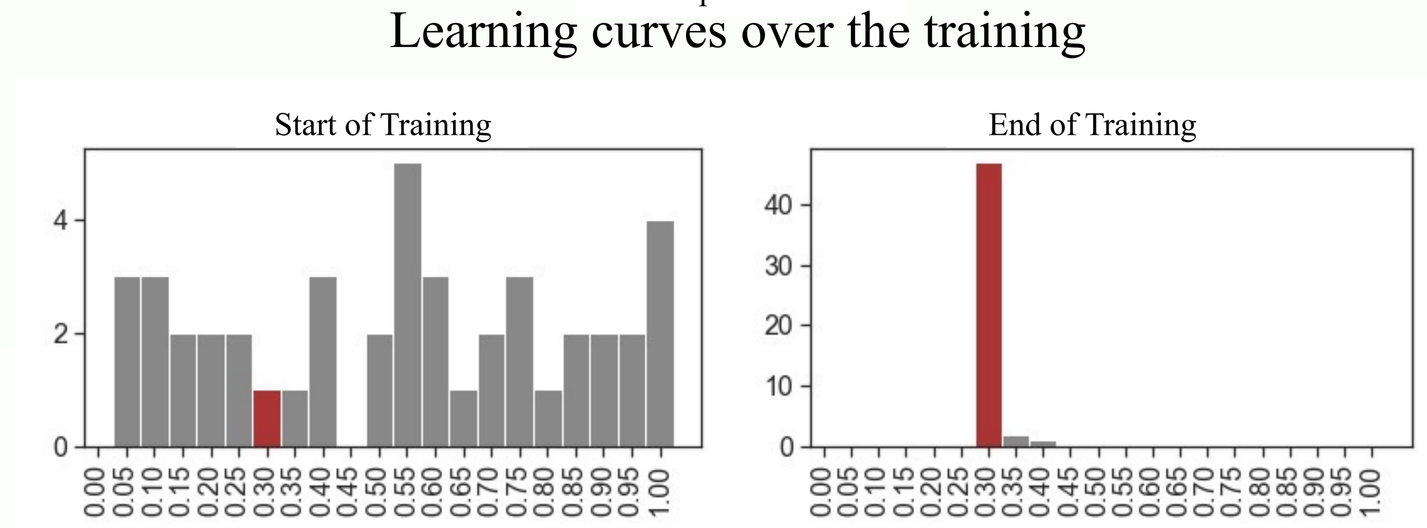
Learning curves over the training | Learned actions in a given period

2: **Initial liquidity decision**: Given above intraday policy, an agent learns to choose the optimal initial liquidity.

- Intraday payment policy is to send as much as possible
- Sate space = $[P_1, P_2, \ldots, P_T]$
- Action space the liquidity choice ($x_0 \cdot B$), where $x_0 \in [0,1]$ discretized into 21 buckets
- Per-episode total cost: $\mathscr{R} = r_c\ell_0 + \sum_{t=1}^{T-1} P_t(1-x_t)\cdot r_d + r_b\ell_{cb}$



Learning curves over the training



Agent A's histograms of learned actions over the course of training

Conclusions

- DRL agents demonstrate optimal learning behaviour in both intraday payments and initial liquidity problems.
- Our results demonstrate the applicability of DRL to high-value payments system, a real-world large stakes strategic game.