Estimating Policy Functions in Payments Systems using Reinforcement Learning*

P. S. Castro[1] A. Desai[2] H. Du[2] R. Garratt[3] F. Rivadeneyra[2]

June 18, 2021

*The opinions here are of the authors and do not necessarily reflect the ones of the Bank of Canada.

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

Liquidity Management in High Value Payments Systems

High-value payments systems are part of the core financial infrastructure; settle transactions between large financial institutions

Problem:

For banks: managing liquidity is costly and can be challenging

For the central bank: ensure the safety and efficiency of the system

Questions

1. Can machine learning find solutions to the liquidity management problem?
2. Could these solutions be a guide for financial institutions and the central bank?

Machine Learning and Liquidity Management

Objective: approximate the policy rules of banks participating in a HVPS using Reinforcement Learning (RL)

- We consider the problem of approximating the best-response functions of banks interacting in a high-value payments system to model their behavior
- Understanding the behaviour of HVPS participants can assist us in two ways:
  1. Ensuring safety and efficiency of payments systems.
  2. Help designing new payments systems

Method: Reinforcement learning (RL)

RL is a computational approach to automate learning from interacting with the environment

- RL train payment system participants to behave optimally in sequential decision tasks mapping observations of the environment to action choices
- In our environment RL agents interact in the payment system to learn policy functions to reduce cost of processing their payments by choosing:
  1. The amount of initial liquidity
  2. The rate at which to pay intraday as the demands arrive from clients

Key result

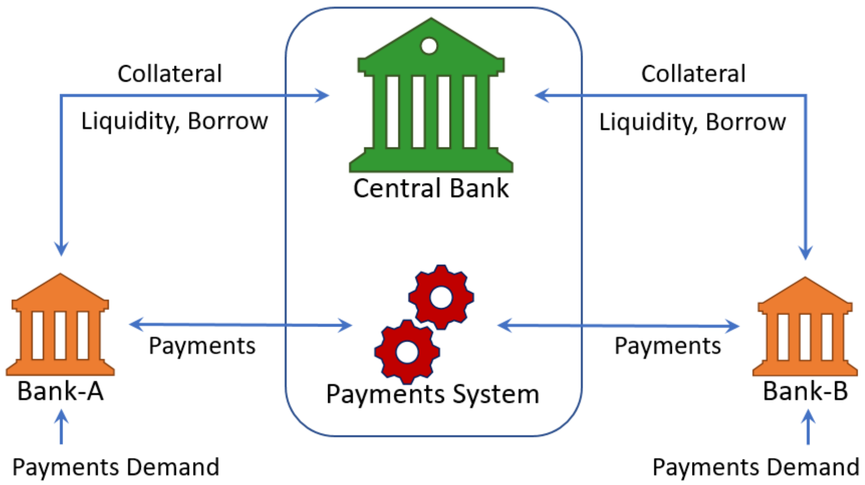
Agents trained with RL learn the optimal policy which minimizes the cost of processing their individual payments

Outline

1. Payments System Environment
2. Reinforcement Learning
3. Learning Setup & Results

Payments System Environment

Environment: Real-time gross settlement system (RTGS)



Environment: Beginning of the day

At $t = 0$: Available collateral B

Decision

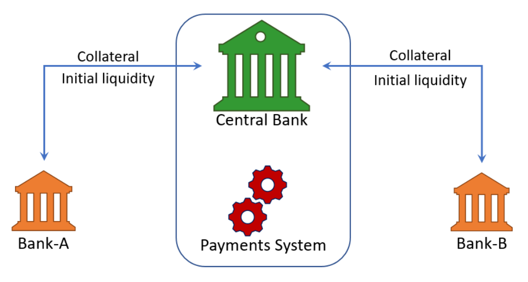
allocate share $x_0 \in [0, 1]$

Liquidity allocation

$\ell_0 = x_0 \cdot B$

Cost of initial liquidity

$r_c \cdot \ell_0$



Environment: Intraday

From $t = 1, \ldots, T-1$: Agent receives payment demands P_t from clients

Decisions

send share $x_t \in [0, 1]$

Liquidity constraint

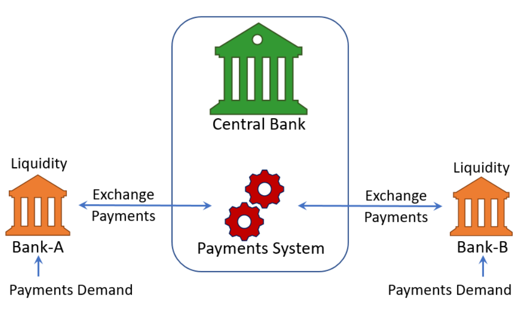
$P_t x_t \leq \ell_{t-1}$

Liquidity evolves

$\ell_t = \ell_{t-1} - P_t x_t + R_t$

Cost of delay

$P_t (1 - x_t) \cdot r_d$



Environment: End-of-day

At $t = T$: Borrow from central bank if necessary

Payment demand

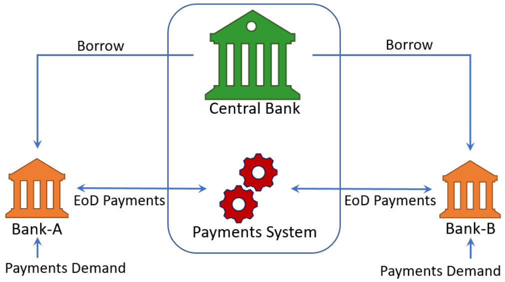
P_T

End-of-day shortage

$\ell_b = P_T - \ell_{T-1}$

Cost of borrowing

$r_b \cdot \ell_b$

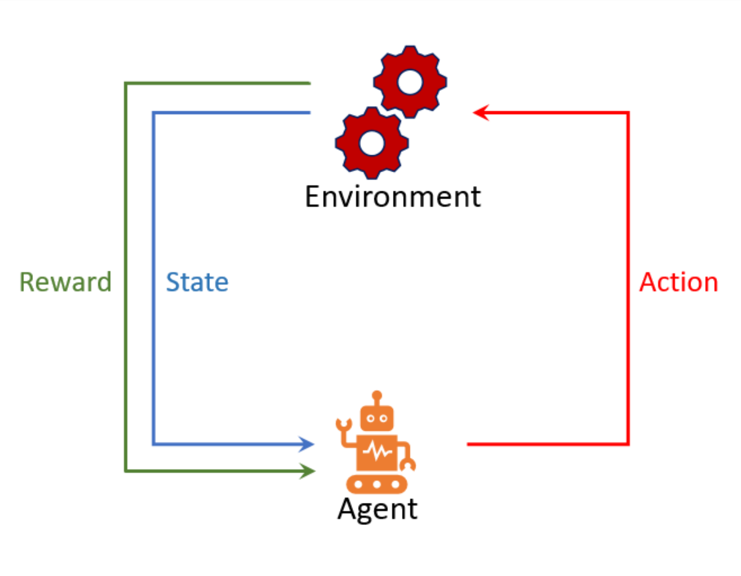


The total cost per episode:

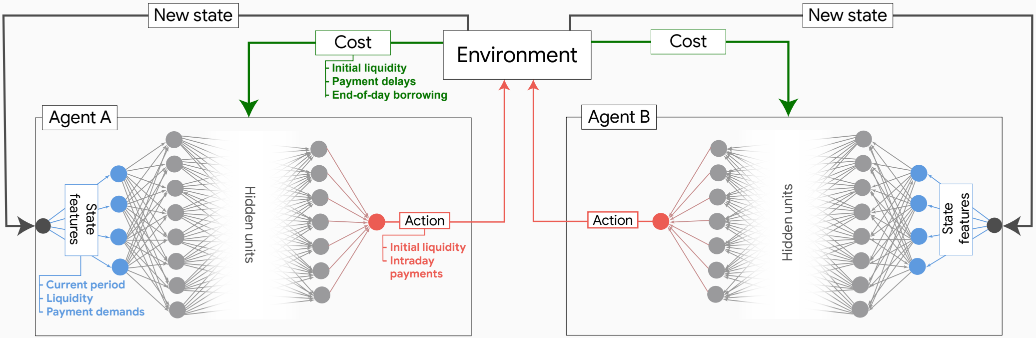
$$\mathcal{R} = r_c \cdot \ell_0 + \sum_{t=1}^{T-1} P_t(1 - x_t) \cdot r_d + r_b \cdot \ell_b$$

Reinforcement Learning

Reinforcement Learning



RL: In the context of payments system



RL: Value functions

is formalized via *policies* π:

$$\pi : \mathcal{S} \to \Delta(\mathcal{A})$$

The *value* of being at state s when following policy π:

$$V^\pi(s) = \mathbb{E}_{a \sim \pi(s)} \left[\underbrace{\mathcal{R}(s, a)}_{\text{cost}} + \underbrace{\gamma}_{\text{discount factor}} \mathbb{E}_{\underbrace{s' \sim \mathcal{P}(s, a)}_{\text{Next-state distribution}}} V^\pi(s') \right]$$

Agent wants to find π^*:

$$\pi^* := \arg\max_\pi V^\pi$$

RL: REINFORCE

Given a start state s_0 and policy parameters θ, we can define:

$$J(\theta) := V^{\pi_\theta}(s_0)$$

and update parameters using stochastic gradient descent:

$$\theta \leftarrow \theta + \alpha \nabla J(\theta)$$

We can sample trajectories $\tau := \langle s_0, a_0, \ldots, s_{T-1}, a_{T-1} \rangle$ from π_θ and use the **policy gradient theorem**:

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

Learning Setup & Results

Learning setup

Objective of the agent is to minimize the cost of processing payments:

$$\mathcal{R} = \text{collateral opportunity cost} + \text{delay cost} + \text{borrowing cost from central bank}$$

Two separate training exercises:

– Learn the initial liquidity decision

– Train the intraday payment decision

Two experiments:

– 2-period scenario to check solution (think morning/afternoon payment cycles)

– 12-period scenario with real data (think hourly cycles)

Learning Setup: Initial liquidity decision

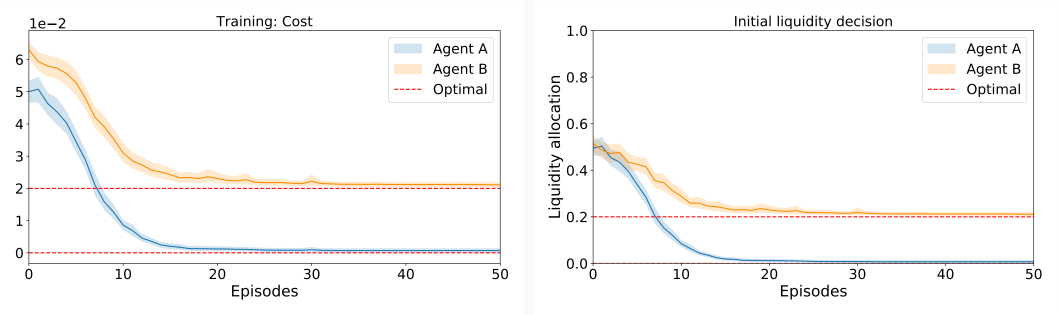
- **State space**: Agent observes the entire vector of intraday payments demands
- **Action space**: $x_t \in \{0, 0.05, 0.1, ..., 1\}$, a fraction of available collateral $(x_t \cdot B)$
- **Intraday action**: Send as much as possible
- **Total cost**:

$$\mathcal{R} = r_c \cdot \ell_0 + \sum_{t=1}^{T-1} P_t(1 - x_t) \cdot r_d + r_b \cdot \ell_b$$

We choose parameters with the relationship: $r_c < r_d < r_b$,
where $r_c = 0.1, r_d = 0.2, r_b = 0.4$

Results: 2-period initial liquidity decision

Dummy payment demands: $P^A = [0,\ 0.15], \quad P^B = [0.15,\ 0.05]$

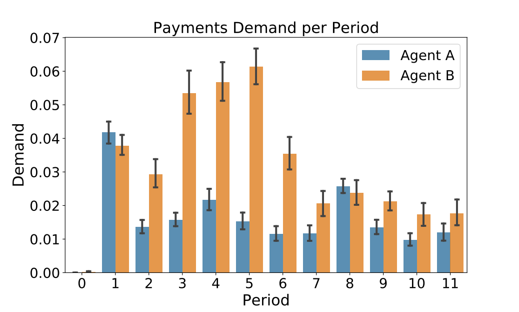


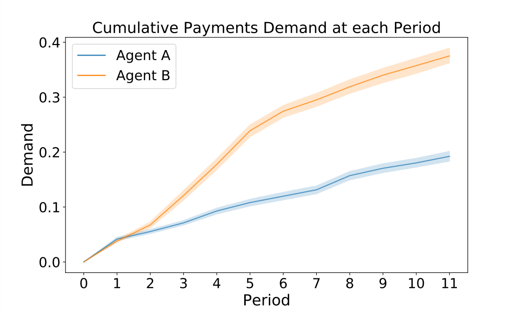
Agents learn the optimal liquidity choices

Payments demand from LVTS

Description of real data:

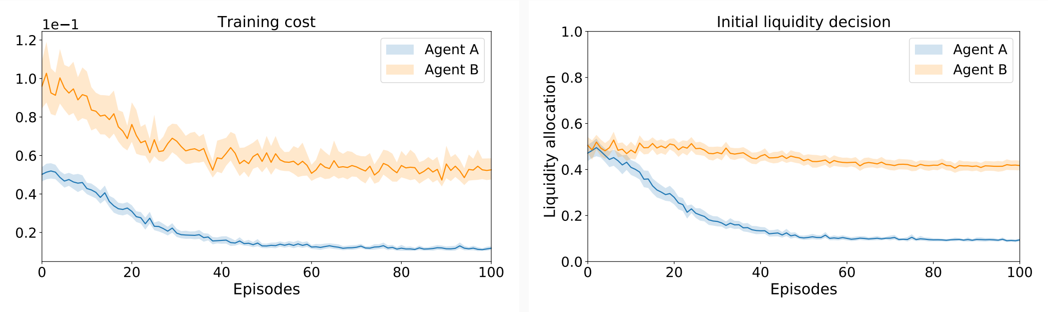
- Normalized hourly aggregate payments observed between two LVTS participants
- Sample size: 380 business days between January 02, 2018 and August 30, 2019





LVTS: Large-value transfer system

Results: 12-period initial liquidity decision



Learning is more gradual but agents learn to reduce their costs

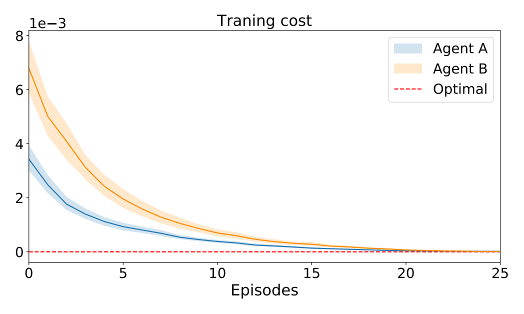
Learning setup: Intraday payment decision

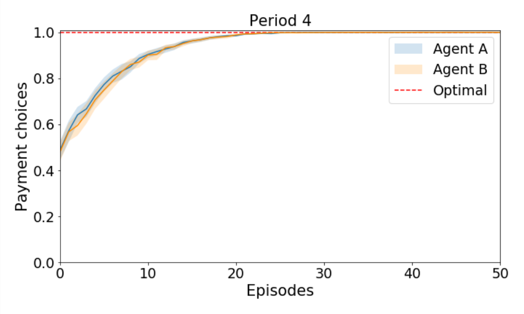
12-Period scenario with known analytical solution:

- **Initial liquidity**: Provide enough liquidity —at no cost— to settle all demand
- **State space**: Period, liquidity, new payments demand, total payments demand
- **Action space**: $x_t = \{0,\ 0.05,\ 0.1,\ ...,\ 1\}$, fraction of payments demand $(x_t P_t)$
- **Total cost**: Processing cost per-episode

$$\mathcal{R} = \sum_{t=1}^{T-1} P_t(1 - x_t) \cdot r_d, \qquad r_d = 0.2$$

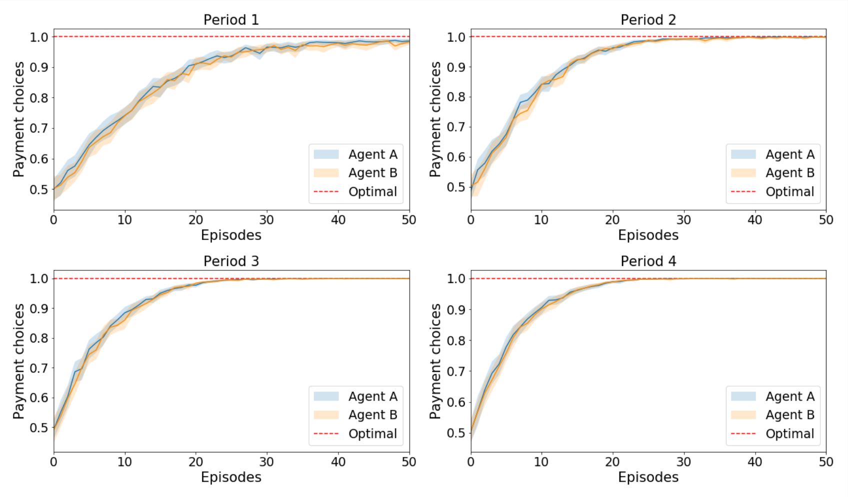
Results: Intraday payment decision





Evolution of cost and action choices incurred during training and testing. The solid lines are average cost for 50 independent training exercises with 99% CI bands.

Results: intraday payment decision

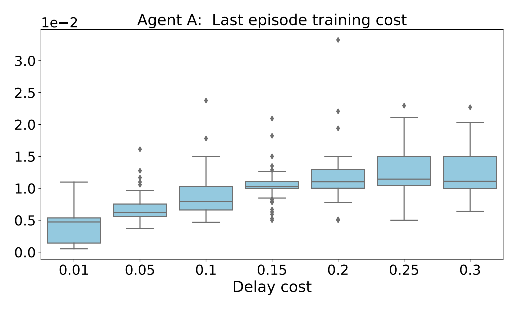


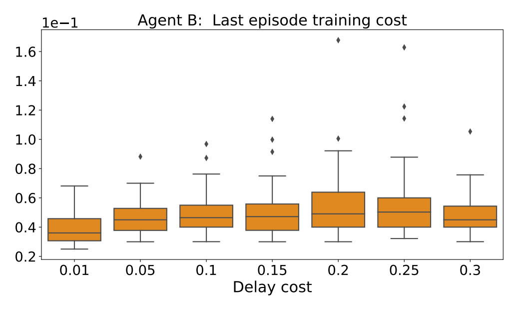
Evolution over the training process of the intraday payment choices x_t (first 4-periods)

Robustness

Learning is robust to several variations in training setup

1. Learning rates, network setup and batch sizes
2. Different payment profiles
3. Costs, in particular delay cost:





Conclusions

Main result:

RL agents learn policies that minimize/reduce the cost of processing payments, promising to explain behaviour and design future payments systems

Next steps:

1. Joint training of the initial liquidity and intraday payment decision
2. Indivisible payments: motive for strategic delay
3. Intraday liquidity market: additional decision rule
4. Simultaneous training of larger number of agents

Thank You!