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]

October 21, 2021 (Paper Link)

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

Economics of Payments X - Virtual Conference

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

Introduction

Liquidity Management in High Value Payments Systems

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

Problems

1. For banks: managing liquidity is costly and can be challenging
2. For the central bank: ensure the safety and efficiency of the system

Questions

1. Can machine learning (ML) 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

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

Challenges: many participants, complex strategic interaction, partial information, many unobservable parameters, and indivisible payments

Reinforcement Learning (RL): flexible environment—multi agents, large state space, stochastic; learn from partial observations; can solve complex decision-making tasks

Problem Setup and Key Lessons

RL agents interact in a simplified payment system to learn policy functions to reduce cost of processing their payments by choosing (separately and simultaneously):

- The amount of initial liquidity allocation
- The rate at which to pay intraday payments

Key lessons:

1. Agents trained with RL learn the optimal policy which minimizes the cost of processing their individual payments
2. Estimated functions can help understand unobservable quantities like delay costs
3. Applying ML requires careful specification of environment and learning algorithm

Reinforcement Learning

Reinforcement Learning

Agent

Parameterized actor and learner

Environment

What the agent interacts with

State (s)

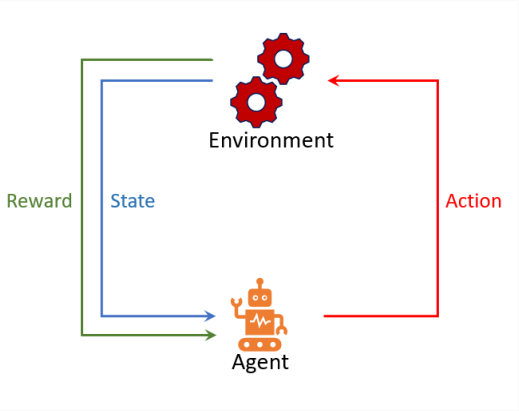
Variables that the agent observes

Action (a)

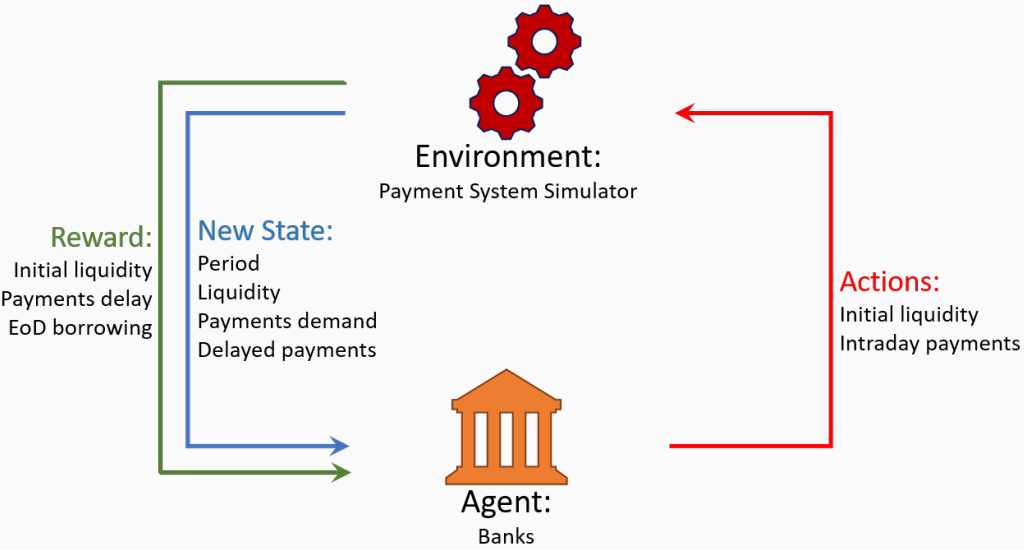
Choice made by agent

Reward ($\mathcal{R}$)

Immediate cost of performing action



RL in the Context of a Payments System



How Does a RL Agent Learn: value function estimation

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

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

– Given a start state s_0 and policy parameters θ, we can define: $J(\theta) := V^{\pi_\theta}(s_0)$

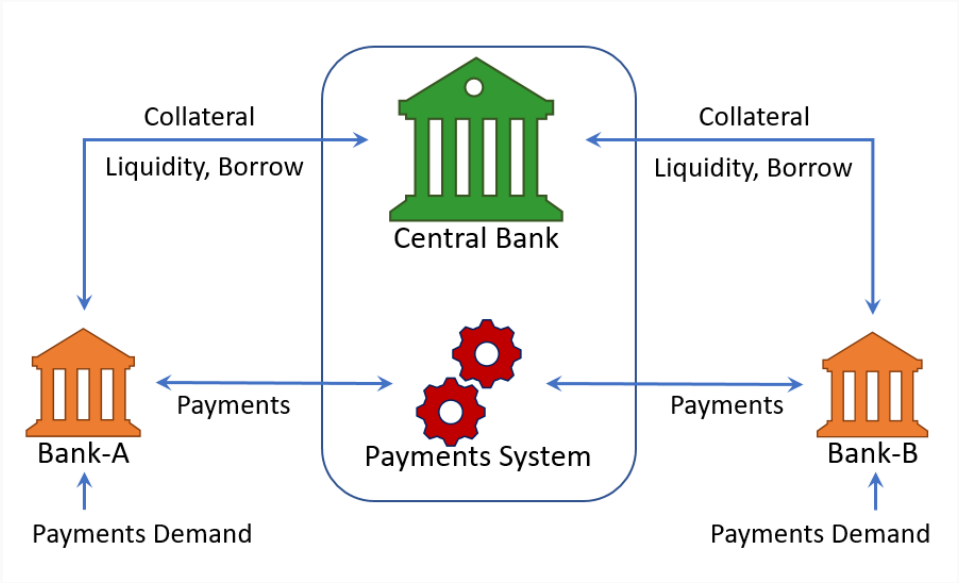
– Using **stochastic gradient descent** to update parameters: $\theta \leftarrow \theta + \alpha \nabla J(\theta)$

– We use **REINFORCE** algorithm: $\nabla J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta} \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\mathcal{R}(s_t, a_t)$

– Agent wants to find **optimal** $\pi^* := \arg\max_\pi V^\pi$

Payments System Environment

A Real-Time Gross Settlement (RTGS) System with Two Agents



Timeline and Decisions of the Agent

Beginning of the day, $t = 0$, available collateral $\mathcal{B}$	
Initial liquidity decision:	$x_0 \in [0, 1]$
Liquidity allocation:	$\ell_0 = x_0 \cdot \mathcal{B}$
Cost of initial liquidity:	$r_c \cdot \ell_0$
Intraday periods, $t = 1, ..., T-1$, Payment demand P_t	
Decision to send:	$x_t \in [0, 1]$
Liquidity constraint:	$P_t x_t \leq \ell_{t-1}$
Evolution of liquidity:	$\ell_t = \ell_{t-1} - P_t x_t + R_t$
Cost of delay:	$r_d \cdot P_t(1 - x_t)$
End-of-day borrowing, $t = T$, Borrowing from CB ℓ_{cb}	
Cost of end-of-day borrowing:	$r_b \cdot \ell_{cb}$

The total cost per day (episode): $\mathcal{R} = r_c \cdot \ell_0 + \sum_{t=1}^{T-1} P_t(1 - x_t) \cdot r_d + r_b \cdot \ell_b$

Learning Setup & Results

Learning Setup

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

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

Separate action learning with two agents:

– Initial liquidity decision

– Intraday payment decision

Joint action learning with only one agent:

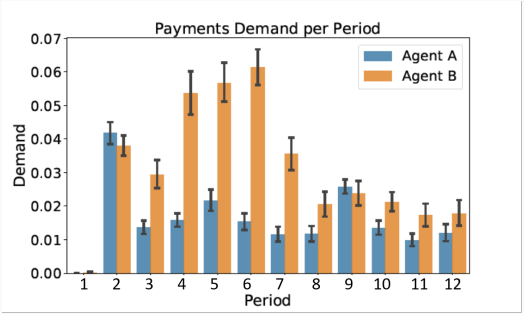
– Initial liquidity and intraday payment decisions with **divisible** payments

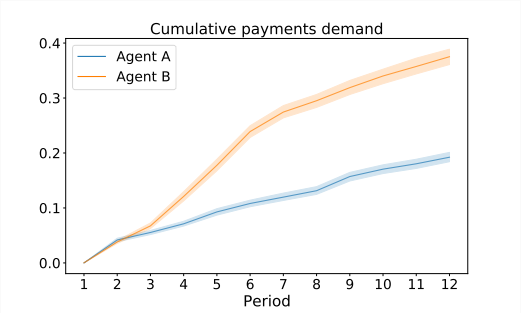
– Initial liquidity and intraday payment decisions with **lumpy** payments

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

Separate Action Learning

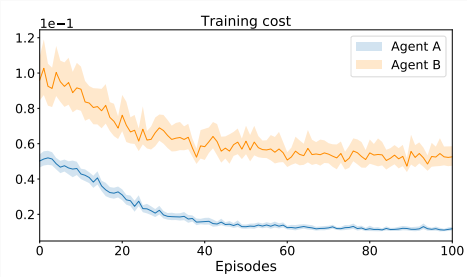
Learning Initial Liquidity Decision

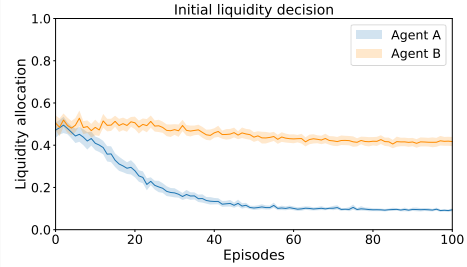
- **State space**: set of intraday payments
- **Action space**: $x_0 \in \{0,\ 0.05,\ ...,\ 1\}$, a fraction of collateral ($x_0 \cdot B$)
- **Intraday action**: send as much as possible
- **Total cost**:

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

where $r_c < r_d < r_b$

$r_c = 0.1, r_d = 0.2, r_b = 0.4$

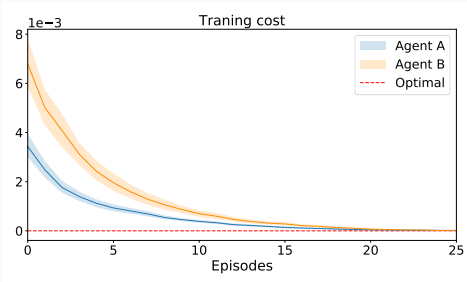


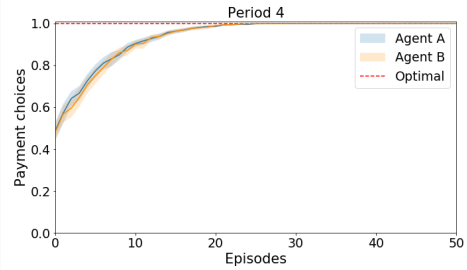


Learning Intraday Payment Decision

- **Initial liquidity**: provide enough liquidity at no cost
- **State space**: period, liquidity, new payments demand, delayed payments
- **Action space**: $x_t = \{0,\ 0.05,\ ...,\ 1\}$, fraction of payments demand $(x_t \cdot P_t)$
- **Total cost**:

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





Joint Action Learning

Joint Action with Only One Agent

Setup:

- Three period ($T = 3$) with dummy payments profile
- Only Agent A is learning both actions (liquidity x_0 and payments x_t)
- Agent B provided enough liquidity and no delay in payments (non-strategic)
- Compare outcomes with and without lumpy payments

Only divisible payments:

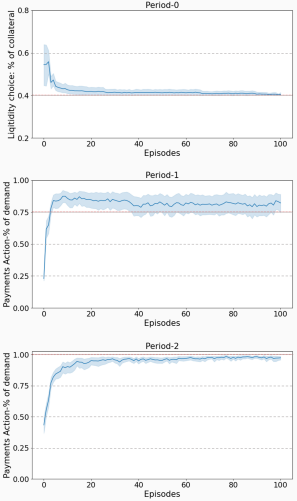
- Joint action learning generate liquidity vs delay trade-off
- Agent A should learn to allocate enough liquidity to send everything in each period

With lumpy payments:

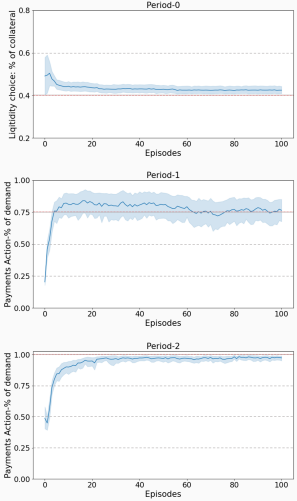
- In period 2 a portion of demand is indivisible
- Additional trade-off: delay in period 1 vs delay in period 2
- Lumpy payment in period 2 generates incentive to delay in period 1

Joint Learning of the Initial Liquidity and Intraday Decisions (one agent only)

Only divisible payments



With lumpy payment



$t = 0$

Initial liquidity is higher in lumpy (wider CI)

$t = 1$

With lumpy agent show precautionary behavior: learns to conserve liquidity

$t = 2$

Agent makes right choices of x_0 & x_1 to be able to send lumpy & divisible payments

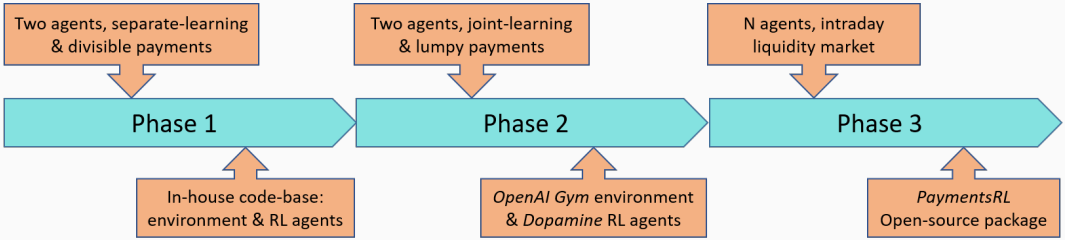
Conclusions & Future Plan

ML can Find Solution to Liquidity Management Problem

Main result:

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

Project Roadmap:



Thank You!