REAL-TIME DATA STREAM PERFORMANCE IMPROVEMENT WITH REINFORCEMENT LEARNING

Y.G.J. Yagoda.

209399D

Degree of Master of Science

Department of Computer Science & Engineering
Faculty of Engineering

University of Moratuwa
Sri Lanka

August 2024

REAL-TIME DATA STREAM PERFORMANCE IM-PROVEMENT WITH REINFORCEMENT LEARNING

Y.G.J. Yagoda.

209399D

Thesis submitted in partial fulfillment of the requirements for the degree.

Master of Science in Computer Science

Department of Computer Science & Engineering

Faculty of Engineering

University of Moratuwa

Sri Lanka

August 2024

DECLARATION

I declare that this is my work and this thesis/dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:19/08/2024

The above candidate has carried out research for the PhD/MPhil/Masters thesis/dissertation under my supervision. I confirm that the declaration made above by the student is true and correct.

Name of Supervisor:

Signature of the Supervisor: Date: 20/08/2024

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor, Prof. Indika Perera of the Department of Computer Science and Engineering at the University of Moratuwa for the guidance and support throughout the project. The project would not have been successful without his valuable insights, immense knowledge, motivation, patience, and understanding. I am also thankful to all the non-academic and technical staff of the Department of Computer Science & Engineering for the support given in numerous ways.

I would also like to thank my parents and family members for supporting and encouraging us throughout the project completion and beyond. This would not have been possible without their tremendous support extended in many different ways.

Finally, a big warm thanks goes to all the friends who have supported me in this journey.

ABSTRACT

Real-time data streaming has become a fundamental aspect of modern data-centric applications, including a wide range of areas such as financial analytics and the Internet of Things (IoT). As real-time data streams become integral to these applications, optimizing their performance is paramount. This thesis aims to improve the efficiency of real-time data streams by employing a reinforcement learning (RL) technique to reduce the latency, especially in the widely used real-time data streaming platform, Kafka. Using reinforcement learning (RL), the system adjusts Kafka settings in real time to minimise latency while remaining within the resource limitation.

The main breakthrough comes in the RL agent's capacity to acquire knowledge from previous encounters and input to make intelligent, immediate choices on parameter modifications. The agent engages with the environment, overseeing system metrics and latency feedback, and constantly adjusts configuration parameters to get optimal performance. This adaptive approach guarantees that the data stream remains highly efficient and responsive, even when faced with fluctuating workload intensities and resource constraints.

The efficiency of the RL-based technique in improving real-time data stream performance across various workloads and circumstances is demonstrated through experimental assessments. The study examines the compromises between reducing latency and optimising throughput, thoroughly examining how varying levels of workload intensity and resource limitations affect performance.

The results of this study provide a substantial contribution to the field of real-time data stream processing. This study utilises reinforcement learning techniques to provide a data-driven and adaptable solution that improves the efficiency and responsiveness of Kafka-based streaming applications in many scenarios. This methodology not only enhances our comprehension of optimising data streams in real time but also offers valuable guidance for implementing reinforcement learning in dynamic contexts with limited resources.

Keywords: Real-Time data stream, Reinforcement Learning, Benchmarking process for real-time data streams, Actor-Critic Method, Markov Decision Process, Policy function, Value function, Agent, Environment.

TABLE OF CONTENTS

DECLARATION **i**

ACKNOWLEDGEMENT **ii**

ABSTRACT **iii**

CHAPTER 1 **1**

INTRODUCTION **1**

1.1 Introduction 1

1.2 Background and Motivation 2

1.3 Problem in Brief 2

1.4 Aim and Objectives 2

1.4.1 Aim 2

1.4.2 Objectives 2

1.4.3 Solution 3

1.5 Thesis Structure 3

CHAPTER 2 **5**

LITERATURE REVIEW **5**

2.1 Introduction 5

2.2 Real-time Data Streams 5

2.2.1 Data Source 5

2.2.2 Data Consumer 6

2.2.3 Data Streaming Platform 6

2.2.4 Distributed event streaming platform – Kafka 6

2.2.5 Design Principals 6

2.2.6 Topics and Partitions 7

2.2.7 Message Delivery in the Kafka System 8

2.2.8 Performance Metrics for real-time data stream 8

2.3 Heuristic approaches for dynamically adjusting parameters to optimise performance in real-time data streams. 8

2.3.1 Blackbox Optimization 9

2.3.2 Hill-Climbing Algorithm (HC) 10

2.3.3 Modified Hill-Climbing Algorithm (mHC) 11

2.3.4 Rule-Based Approach (RB) 14

2.3.5 Comparison-Based Model (CBM) 15

2.3.6 Weighted Latin Hypercube Sampling (LHS) 16

2.3.7 Exploration and Exploitation (E&E) process 16

2.3.8 Bound-and-Search (BS) Algorithm: 16

2.4 Benchmarking Distributed Stream Data Processing Systems. 16
2.4.1 Benchmark Design Decisions 16

2.5 Benchmarking Distributed Stream Processing Frameworks for Real-Time Classical Machine Learning Applications 18
2.5.1 Design of the System 18
2.5.2 Input Parameters 19
2.5.3 Output Parameters 19

2.6 Reinforcement Learning for Optimizing Real T-me Data Stream Performance 19
2.6.1 Reinforcement Learning 20
2.6.2 Reward Types 21
2.6.3 Policies and Value Function 21
2.6.4 State Value Function 21
2.6.5 Optimal Policy 22
2.6.6 Actor-Critic Method 22
2.6.7 Actor critic algorithm pseudo code 24
2.6.8 Markov Decision Processes (MDPs) 24
2.6.9 Usage of Actor-Critic methods to improve performance of real-time stream. 30
2.6.10 Reinforcement Learning Algorithms 31

Summary 37

CHAPTER 3 38

METHODOLOGY 38

3.1 Introduction 38

3.2 Understanding Kafka Behaviours and Configurations 38
3.2.1 Topics 38
3.2.2 Brokers 39
3.2.3 Partition 39
3.2.4 Batch Size 39
3.2.5 Linger Milliseconds(linger.ms) 40

3.3 Data Pipeline Setup and Data Streaming 40
3.3.1 Data Collection 41
3.3.2 Data Streaming 41
3.3.3 Providing latency feedback for optimizing data streaming latency. 43

3.4 Usage of Actor-Critic networks to optimize the latency. 44
3.4.1 Process of using Actor-critic 44
3.4.2 State dimension. 46
3.4.3 Action dimension. 47
3.4.4 Reward Function 48
3.4.5 A2C Agent 48
3.4.6 Latency Optimization Process Utilizing the Acro-critic Algorithm 48

3.4.7 Exploration vs. Exploitation 49
3.4.8 Streaming setup with a Heuristic approach 50
3.4.9 Challenges and Considerations 50

CHAPTER 4 **51**

EXPERIMENTS AND EVALUATION **51**

4.1 Comparison of Latency Variation: Regular Streaming, A2C, and mHC Approaches 51
4.1.1 Experimental Scenarios for Kafka Producer Configurations 52
4.1.2 Experiments Associated with Increased Data Rates 53
4.1.3 Experiments Associated with Lower Data Rates 76

4.2 Results and Discussion 96

CHAPTER 5 **99**

CONCLUSION **99**

5.1 Conclusion 99

5.2 Research Limitations 100

5.3 Future Work 101

LIST OF FIGURES

Figure 2-1	Components involved in real-time data stream[8]	5
Figure 2-2	Kafka Architecture[10]	7
Figure 2-3	Anatomy of a Topic[10]	7
Figure 2-4	Representation of performance using Blackbox optimization[19]	9
Figure 2-5	Parameter range for parameter xi	9
Figure 2-6	Parameter tunning problem of achieving best performance.	10
Figure 2-7	Rule-Based approach code[18]	15
Figure 2-8	Data Generator[23]	17
Figure 2-9	Merging different data sources[23]	17
Figure 2-10	Benchmark design for classical ML approach	18
Figure 2-11	Reinforcement Learning Framework[30]	20
Figure 2-12	Probabilities summation of transitions of next states[30]	21
Figure 2-13	Cumulative Reward[31]	21
Figure 2-14	State value function	22
Figure 2-15	Actor critic algorithm pseudo code[32], [42]	24
Figure 2-16	RL behaviour[45]	25
Figure 2-17	Transition probability function	26
Figure 2-18	State value function	28
Figure 2-19	Action Value function	28
Figure 2-20	Optimal Policy[51]	29
Figure 2-21	Optimal state value function[51]	29
Figure 2-22	Optimal action value function[51]	29
Figure 2-23	Bellman Optimality Equation	30
Figure 2-24	Actor and Critic Agent definitions	30
Figure 2-25	Classification of Popular RL Algorithms[53]	31
Figure 2-26	Q-Learning flow[54]	33
Figure 2-27	Q-table[55]	33
Figure 2-28	state-action function[55]	34
Figure 2-29	Deep Q Network	34
Figure 2-30	Deep Q-network Architecture	35
Figure 2-31	Policy Gradient[56]	35
Figure 2-32	Policy Network architecture.	36
Figure 3-1	batch.size configuration	40
Figure 3-2	linger.ms configuration	40
Figure 3-3	iot-sensor-data table DDL	41
Figure 3-4	Improved real-time data stream architecture	42
Figure 3-5	Actor-network definition	46
Figure 3-6	Action select function	47
Figure 3-7	Reward function	48
Figure 3-8	A2C Agent	48
Figure 3-9	Critic loss function	49
Figure 3-10	Actor loss function	49
Figure 4-1	Experiments hardware specification	51

Figure 4-2	A2C approach Increased data rate with default batch size results	53
Figure 4-3	Regular approach Increased data rate with default batch size results	53
Figure 4-4	mHC approach Increased data rate with default batch size results	53
Figure 4-5	Exp19-A2C-latency variation	54
Figure 4-6	Exp19-A2C- batch size and linger.ms value variation	54
Figure 4-7	Exp19-A2C-average latency variation for each 10 percent segment	54
Figure 4-8	Exp19-regular latency variation	55
Figure 4-9	Exp19-regular-average latency variation for each 10 percent segment	55
Figure 4-10	Exp20-A2C-latency variation	56
Figure 4-11	Exp20-A2C- batch size and linger.ms value variation	56
Figure 4-12	Exp20-A2C-average latency variation for each 10 percent segment	56
Figure 4-13	Exp20-regular latency variation	57
Figure 4-14	Exp20-regular-average latency variation for each 10 percent segment	57
Figure 4-15	mHC-exp-12-Latency Variation	58
Figure 4-16	mHC-exp-12-batch size and linger.ms value variation	59
Figure 4-17	mHC-exp12-average latency variation for each 10 percent segment	59
Figure 4-18	mHC-exp-12-Latency Variation	60
Figure 4-19	mHC-exp-15-batch size and linger.ms value variation	61
Figure 4-20	mHC-exp15-average latency variation for each 10 percent segment	61
Figure 4-21	mHC-exp-18-Latency Variation	62
Figure 4-22	mHC-exp-18-batch size and linger.ms value variation	63
Figure 4-23	mHC-exp18-average latency variation for each 10 percent segment	63
Figure 4-24	A2C approach Increased data rate with default batch size and high linger.ms results	66
Figure 4-25	Regular approach Increased data rate with default batch size results	66
Figure 4-26	mHC approach Increased data rate with default batch size results	66
Figure 4-27	Exp23-A2C-latency variation	67
Figure 4-28	Exp23-A2C- batch size and linger.ms value variation	67
Figure 4-29	Exp23-A2C-average latency variation for each 10 percent segment	67
Figure 4-30	Exp23-regular latency variation	68
Figure 4-31	Exp23-regular-average latency variation for each 10 percent segment	68
Figure 4-32	Exp24-A2C-latency variation	69
Figure 4-33	Exp24-A2C- batch size and linger.ms value variation	69

Figure 4-34	Exp24-A2C-average latency variation for each 10 percent segment	69
Figure 4-35	Exp24-regular latency variation	70
Figure 4-36	Exp24-regular-average latency variation for each10 percent segment	70
Figure 4-37	mHC-exp-20-Latency Variation	71
Figure 4-38	mHC-exp-20-batch size and linger.ms value variation	72
Figure 4-39	mHC-exp20-average latency variation for each 10 percent segment	72
Figure 4-40	mHC-exp-21-Latency Variation	73
Figure 4-41	mHC-exp-21-Batch size and linger.ms variation	74
Figure 4-42	mHC-exp21-average latency variation for each 10 percent segment	74
Figure 4-43	A2C approach lower data rate with default batch size results	76
Figure 4-44	Regular approach lower data rate with default batch size results	76
Figure 4-45	mHC approach lower data rate with default batch size results	76
Figure 4-46	Exp17-A2C-latency variation	77
Figure 4-47	Exp17-A2C- batch size and linger.ms value variation	77
Figure 4-48	Exp17-A2C-average latency variation for each 10 percent segment	77
Figure 4-49	Exp17-regular latency variation	78
Figure 4-50	Exp17-regular-average latency variation for each 10 percent segment	78
Figure 4-51	Exp18-A2C-latency variation	79
Figure 4-52	Exp18-A2C- batch size and linger.ms value variation	79
Figure 4-53	Exp18-A2C-average latency variation for each 10 percent segment	79
Figure 4-54	Exp18-regular latency variation	80
Figure 4-55	Exp18-regular-average latency variation for each 10 percent segment	80
Figure 4-56	mHC-exp-36-Latency Variation	81
Figure 4-57	mHC-exp-36-batch size and linger.ms value variation	82
Figure 4-58	mHC-exp36-average latency variation for each 10 percent segment	82
Figure 4-59	mHC-exp-37-Latency Variation	83
Figure 4-60	mHC-exp-37-batch size and linger.ms value variation	84
Figure 4-61	mHC-exp37-average latency variation for each 10 percent segment	84
Figure 4-62	A2C approach lower data rate , default batch size with high linger value results	87
Figure 4-63	Regular approach lower data rate , default batch size with high linger value results	87
Figure 4-64	mHC approach lower data rate , default batch size with high linger value results	87

Figure 4-65	Exp21-A2C-latency variation	88
Figure 4-66	Exp21-A2C- batch size and linger.ms value variation	88
Figure 4-67	Exp21-A2C-average latency variation for each 10 percent segment	88
Figure 4-68	Exp21-regular latency variation	89
Figure 4-69	Exp21-regular-average latency variation for each 10 percent segment	89
Figure 4-70	Exp22-A2C-latency variation	90
Figure 4-71	Exp22-A2C- batch size and linger.ms value variation	90
Figure 4-72	Exp22-A2C-average latency variation for each 10 percent segment	90
Figure 4-73	Exp22-regular latency variation	91
Figure 4-74	Exp22-regular-average latency variation for each 10 percent segment	91
Figure 4-75	mHC-exp-38-Latency variation	92
Figure 4-76	mHC-exp-38-batch size and linger.ms value variation	93
Figure 4-77	mHC-exp38-average latency variation for each 10 percent segment	93
Figure 4-78	mHC-exp-39-Latency Variation	94
Figure 4-79	mHC-exp-39-batch size and linger.ms value variation	95
Figure 4-80	mHC-exp39-average latency variation for each 10 percent segment	95