VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ STUDIJŲ PROGRAMA

# Srautinio apdorojimo sistemų balansavimas taikant mašininį mokymąsi

## Balancing stream processing systems using machine learning

Mokslo tiriamasis darbas I

| | | |
|---|---|---|
| Atliko: | Vytautas Žilinas | (parašas) |
| Darbo vadovas: | Partn. doc. Andrius Adamonis | (parašas) |
| Recenzentas: | Prof. dr. Aistis Raudys | (parašas) |

Vilnius – 2020

# 1. Tyrimas

## 1.1. Tyrimo aktualumas ir naujumas

Realaus laiko duomenų apdorojimas (angl. real-time data processing) yra jau senai nagrinėjamas kaip vienas iš būdų apdoroti didelių kiekių duomenis (angl. Big data). Vienas iš realaus laiko apdorojimo sprendimų yra srautinis duomenų apdorojimas. Srautinis duomenų apdorojimas (angl. stream processing) – lygiagrečių programų kūrimo modelis, pasireiškiantis sintaksiškai sujungiant nuoseklius skaičiavimo komponentus srautais, kad kiekvienas komponentas galėtų skaičiuoti savarankiškai [Bea15].

Yra keli pagrindiniai srautinio apdorojimo varikliai: „Apache Storm“, „Apache Spark“, „Heron“ ir kiti. „Apache Storm“ ir „Heron“ apdoroja duomenis ne duomenų srautais, o „Apache Spark“ mikro-paketais [KKW+15]. „Heron“ srautinio apdorojimo variklis, buvo išleistas „Twitter“ įmonės 2016 metais kaip greitesnė alternatyva „Apache Storm“ srautinio apdorojimo varikliui [Ram16].Šiame darbe bus naudojamas „Heron“, kadangi tai yra naujesnis ir greitesnis srautinio apdorojimo variklis nei „Apache Storm“ [KBF+15].

Srautinio apdorojimo sistemų balansavimas (angl. auto-tuning) - tai sistemos konfigūracijos valdymas siekiant užtikrinti geriausią resursų išnaudojimą - duomenų apdorojimas neprarandant greičio, bet ir naudojant tik reikiamą kiekį resursų. Kadangi srautinio apdorojimo sistemų komponentai yra kuriami kaip lygiagretus skaičiavimo elementai, todėl jie gali būti plečiami horizontaliai ir vertikaliai [Bea15] keičiant sistemų konfigūraciją. Tačiau lygiagrečių elementų kiekio keitimas nėra vienintelis būdas optimizuoti resursų išnaudojimą. Kiekvienas variklis turi savo rinkinį konfigūruojamų elementų. Darbe naudojamas „Heron“ variklis leidžia optimizuoti sistemas naudojant 56 konfigūruojamus svertus (angl. levers) [Her19].

Yra skirtingi būdai kaip gali būti parenkama tinkama konfigūracija. Kadangi dar nėra naudojimui paruoštų sprendimų, kurie galėtų balansuoti srautinio apdorojimo sistemas savarankiškai, dažniausiai tai daro duomenų inžinieriai, kurie dirba su šiomis sistemomis. Kadangi srautinio apdorojimo sistemų apkrovos gali būti skirtingų pobūdžiu (duomenų kiekis, skaičiavimų sudėtingumas, nereguliari apkrova), o inžinieriai konfigūruodami išbando tik keletą derinių ir pasirenka labiausiai tinkanti [FA17], lieka labai daug skirtingų neišbandytų konfigūracijos variacijų. Optimalios konfigūracijos suradimas yra NP sudėtingumo problema [SSP04], nes žmonėms yra sunku suvokti didelį kiekį konfigūracijos variacijų. Vienas iš būdų automatiškai valdyti konfigūraciją buvo pasiūlytas 2017 metų straipsnyje „Dhalion: self-regulating stream processing in heron“, kuriame autoriai aprašo savo sukurtą sprendimą „Dhalion“, kuris konfigūruoja „Heron“ srautinio apdorojimo sistemas pagal esama apkrova ir resursus, t.y. jei apdorojimo elementų išnaudojimas išauga >100%, „Dhalion“ padidina lygiagrečiai dirbančių apdorojimo elementų kiekį [FAG+17]. Tačiau šis sprendimas leidžia reguliuoti tik elementų lygiagretumą ir tai daro tik reaktyviai. Vienas iš naujausių būdų balansuoti srautinio apdorojimo sistemas - mašininis mokymasis. Vienas iš tokių bandymų buvo aprašytas 2018 metų straipsnyje „Auto-tuning Distributed Stream Processing Systems using Reinforcement Learning“, kuriame buvo atliktas tyrimas - „Apache Spark“ sistemos balansavimui buvo naudojamas skatinamojo mokymo REINFORCE algoritmas, kuris pagal dabartinę konfigūraciją ir

gaunamas metrikas keitė srautinio apdorojimo sistemos konfigūracijas. Šiame tyrime buvo nustatyta, jog sprendimas, naudojantis mašininį mokymąsi, suranda konfigūraciją per trumpesnį laiką nei žmones ir taip pat sukurtą konfigūraciją naudojančios srautinio apdorojimo sistemos laukimo trukme (angl. latency) yra 60-70% mažesnė nei tyrimo metu žmonių sukurtos konfigūracijos [VC18]. Šiame darbe naudojamas „Heron“ variklis leidžia prie savęs prijungti sukurtą išorinę metrikų surinkimo programą, kuri gali rinkti tokias sistemų metrikas kaip: naudojama RAM atmintis, CPU apkrova, komponentų paralelizmas ir kitas, kurios bus naudojamos sistemos balansavimui.

Skatinamasis mokymasis yra vienas iš mašininio mokymosi tipų. Šis mokymasis skiriasi nuo kitų, nes nereikia turėti duomenų apmokymui, o programos mokosi darydamos bandymus ir klysdamos. Pagrindinis uždavinys naudojant skatinamąjį mokymąsi yra surasti balansą tarp naujų sprendimų tyrinėjimo (angl. exploration) ir turimos informacijos išnaudojimo (angl. exploitation) [JOA10]. Vienas iš pagrindinių privalumų naudojant skatinamąjį mokymąsi balansavimui - nereikia turėti išankstinių duomenų apmokymui kas leidžia jį paprasčiau pritaikyti skirtingoms srautinio apdorojimo sistemų apkrovoms. Tačiau tokio tipo mašininis mokymasis turi ir problemų: sudėtinga aprašyti tinkamos konfigūracijos apdovanojimo (angl. reward) funkciją ir turi būti teisingai aprašytas balansas tarp tyrinėjimo ir išnaudojimo tam, kad nebūtų patirti nuostoliai [FA17].

Yra sukurta daug skatinamojo mokymosi algoritmų (Monte Carl, Q-learning, Deep Q Network ir kiti), tad šiame darbe jie bus apžvelgti ir vienas iš jų bus pasirinktas ir pritaikytas išsikeltam uždaviniui. Algoritmas bus pasirinktas pagal tai, kuris yra tinkamiausias srautinių apdorojimų sistemų balansavimui.

# 2. Darbas

## 2.1. Darbo tikslas

Ištirti mašininio mokymosi tinkamumą srautinio apdorojimo sistemų balansavimui.

## 2.2. Uždaviniai

1. Apibrėžti duomenų pobūdį ir apkrovas, kurios bus naudojamos eksperimente bei pasirinkti srautinio apdorojimo sistemų metrikas, kurios bus naudojamos eksperimento rezultatų palyginimui.
2. Atlikti literatūros analizė apie esamus skatinamojo mokymosi algoritmus ir pasirinkti vieną iš jų eksperimentui.
3. Sukurti eksperimentinį sprendimą, kuris pritaiko pasirinktą mašininio mokymosi algoritmą srautinio apdorojimo sistemų balansavimui.
4. Atlikti eksperimentą ir palyginti gautus rezultatus su alternatyvomis - „Heron“ su standartine konfigūracija, „Heron“ su „Dhalion“ priedu bei „Heron“ balansavimas pritaikius REINFORCE algoritmą.

## 2.3. Laukiami rezultatai

1. Apibrėžtas duomenų pobūdis ir surinktos metrikos, reikalingos „Heron“ srautinio apdorojimo sistemos balansavimo efektyvumo įvertinimui ir eksperimento rezultatų palyginimui.
2. Išanalizavus esamų skatinamojo mašininio mokymosi algoritmų savybes pasirinktas algoritmas, kuris tinka balansavimo uždaviniui spręsti.
3. Sukurtas eksperimentinis sprendimas, kuris balansuotų „Heron“ srautinio apdorojimo sistemą naudojant pasirinkta skatinamojo mokymosi algoritmą.
4. Atliktas eksperimentas su „Heron“ ir alternatyvomis ir išanalizuoti bei palyginti gauti rezultatai.

# Literatūra


[Bea15] Jonathan Beard. A short intro to stream processing. http://www.jonathanbeard.io/blog/2015/09/19/streaming-and-dataflow.html, 2015-09.

[FA17] Avrilia Floratou ir Ashvin Agrawal. Self-regulating streaming systems: challenges and opportunities. *Proceedings of the International Workshop on Real-Time Business Intelligence and Analytics*, BIRTE '17, 1:1–1:5, Munich, Germany. ACM, 2017. ISBN: 978-1-4503-5425-7. DOI: 10.1145/3129292.3129295. URL: http://doi.acm.org/10.1145/3129292.3129295.

[FAG+17] Avrilia Floratou, Ashvin Agrawal, Bill Graham, Sriram Rao ir Karthik Ramasamy. Dhalion: self-regulating stream processing in heron. *Proceedings of the VLDB Endowment*, 10:1825–1836, 2017-08. DOI: 10.14778/3137765.3137786.

[Her19] Heron. Heron documentation on cluster configuration. http://heron.incubator.apache.org/docs/cluster-config-overview/, 2019.

[JOA10] Thomas Jaksch, Ronald Ortner ir Peter Auer. Near-optimal regret bounds for reinforcement learning. *J. Mach. Learn. Res.*, 11:1563–1600, 2010-08. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1756006.1859902.

[KBF+15] Sanjeev Kulkarni, Nikunj Bhagat, Maosong Fu, Vikas Kedigehalli, Christopher Kellogg, Sailesh Mittal, Jignesh M. Patel, Karthik Ramasamy ir Siddarth Taneja. Twitter heron: stream processing at scale. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, p. 239–250, Melbourne, Victoria, Australia. ACM, 2015. ISBN: 978-1-4503-2758-9. DOI: 10.1145/2723372.2742788. URL: http://doi.acm.org/10.1145/2723372.2742788.

[KKW+15] Holden Karau, Andy Konwinski, Patrick Wendell ir Matei Zaharia. *Learning spark: lightning-fast big data analysis*. " O'Reilly Media, Inc.", 2015.

[Ram16] Karthik Ramasamy. Open sourcing twitter heron, 2016.

[SSP04] David G. Sullivan, Margo I. Seltzer ir Avi Pfeffer. Using probabilistic reasoning to automate software tuning. *SIGMETRICS Perform. Eval. Rev.*, 32(1):404–405, 2004-06. ISSN: 0163-5999. DOI: 10.1145/1012888.1005739. URL: http://doi.acm.org/10.1145/1012888.1005739.

[VC18] Luis M. Vaquero ir Felix Cuadrado. Auto-tuning distributed stream processing systems using reinforcement learning, 2018. arXiv: 1809.05495 [cs.DC].