HPI Future SOC Lab - Proceedings 2020

Christoph Meinel, Andreas Polze, Karsten Beins,
Rolf Strotmann, Ulrich Seibold, Kurt Rödszus,
Jürgen Müller, Jürgen Sommer(Eds.)

Technische Berichte Nr. 159

des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam | 159

Christoph Meinel | Andreas Polze | Karsten Beins | Rolf Strotmann | Ulrich Seibold |
Kurt Rödszus | Jürgen Müller | Jürgen Sommer(Eds.)

HPI Future SOC Lab

Proceedings 2020

Universitätsverlag Potsdam

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://dnb.dnb.de/ abrufbar.

Universitätsverlag Potsdam 2024
http://verlag.ub.uni-potsdam.de/

Am Neuen Palais 10, 14469 Potsdam
Tel.: +49 (0)331 977 2533 / Fax: 2292
E-Mail: verlag@uni-potsdam.de

Die Schriftenreihe **Technische Berichte des Hasso-Plattner-Instituts für Digital Engineering an der Universität Potsdam** wird herausgegeben von den Professoren des Hasso-Plattner-Instituts für Digital Engineering an der Universität Potsdam.

ISSN (print) 1613-5652
ISSN (online) 2191-1665

Druck: docupoint GmbH Magdeburg

ISBN 978-3-86956-565-1

Zugleich online veröffentlicht auf dem Publikationsserver der Universität Potsdam:
https://doi.org/10.25932/publishup-59801
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-598014

Preface

The *HPI Future SOC Lab* is a cooperation of the Hasso Plattner Institute (HPI) and industry partners. Its mission is to enable and promote exchange and interaction between the research community and the industry partners.

The HPI Future SOC Lab provides researchers with free of charge access to a complete infrastructure of state of the art hard and software. This infrastructure includes components, which might be too expensive for an ordinary research environment, such as servers with up to 64 cores and 2 TB main memory. The offerings address researchers particularly from but not limited to the areas of computer science and business information systems. Main areas of research include cloud computing, parallelization, and In-Memory technologies.

This technical report presents results of research projects executed in 2020. Selected projects have presented their results on April 21st and November 10th 2020 at the Future SOC Lab Day events.

Contents

Energy Efficiency, Virtualization and Performance
Fourth (WEEVILF)

Carlos Juiz, Belen Bermejo, and Alejandro Calle

Computer Science Department
University of the Balearic Islands, Spain
{cjuiz,belen.bermejo}@uib.es

1 Project idea

The prevailing international scientific opinion on climate change is, that human activities resulted in substantial global warming from mid-20^{th} century, and that continued growth in greenhouse gas concentrations, caused by human-induced emissions, is mainly (direct or indirectly) due to energy consumption. The sector where energy consumption has grown tremendously is IT (Information Technologies). On one hand, datacentres that host cloud services are becoming huge warehouses with lot of servers, additional electronic equipment and complex cooling systems. These computers and telecommunications networks are today primarily responsible for electronical energy consumption caused by datacentres, which maintain the quality of Internet service in operation. Giving more capacity to these datacentres usually means more cloud servers and consequently additional more equipment and cooling are needed, too. On the other hand, the increasing number of users, especially due to the popularization of cloud services, produced continuously capacity problems at datacentres due to performance requirements [5, 6]. Thus, a new challenge directly related to this research area emerges: the energy efficiency of cloud servers. WEEVILT project is aimed by this combination of these topics. Our main project objectives are:

1. The characterization of the performance and energy consumption from consolidated servers through benchmarking and monitoring techniques.
2. Modelling the energy consumption patterns and performance of consolidated servers. We shall infer models of such energy, performance and virtualization, together.
3. Performing several comparative studies of benchmarking between physical and virtual servers.
4. The validation of our performance-oriented previous findings and energy efficiency virtualization models through experimentations in real datacenters, as HPI Future SOC Lab.

The WEEVILF project is defined as the extension of the previous one (WEEVILT) developed using the HPI Future SOC Lab infrastructure (RX600S5-1 server). This

project was developed with the collaboration of the Mrs. Alejandro Calle, undergraduate student of computer science.

Then, we attempt to go in depth in CPU-overhead sources when consolidating virtual machines. Specifically, we study how the behaviour of OV_v and OV_c under different % of CPU utilization, as we state in the previous report as next steps (WEEVILF project).

2 Used Future SOC Lab resources

In order to answer the research question, we design a methodology based in the following stages:

1. To characterize the performance of physical servers through benchmarking and monitoring techniques [8].
2. Modelling the performance degradation of consolidated servers. We shall infer models of such energy, performance and virtualization, together.
3. Performing several comparative studies of benchmarking between physical and virtual servers.
4. The validation of our performance-oriented previous findings of virtualization models through experimentation in real datacentres, as HPI Future SOC Lab.

The exposed methodology was performed in the HPI Future SOC Lab infrastructure, specifically in the rx600s5-1 server which hardware has 48 CPUs and 1024 GB of RAM memory. We can see the actions developed in each stage and the correspondent outcome, as follows:

- Stage 1:
  - Characterizing the performance of PM through benchmarking and monitoring techniques
  - PM's performance when consolidating virtual machines
  - PM's performance when varying the % of CPU utilization
- Stage 2:
  - Description of virtual machine consolidation behaviour and CiS^2 index
- Stage 3:
  - Measuring and quantifying the virtual machine consolidation overhead in HPI infrastructure and UIB's servers
- Stage 4:
  - Discussion and results' analysis
  - Comparison between UIB and HPI servers in terms of performance
  - Comparison between UIB and HPI servers in terms of CiS^2 index [7].

3 Findings

In previous section we explain the stages we perform to answer the research question, as well as the developed actions and its outcomes. In this section we summarize some of the main funding resulting of this projects, which are in-depth explained in the related publication.

In tables 1 and 4 the mean response time for the HPI and UIB servers are depicted, respectively. We varied the number of parallel physical servers, which executes a N-th part of the workload, and the % of CPU utilization too. The mean response time for he HPI server is lesser than the UIB one due to the hardware resources of HPI server. Besides, for both servers, among the number of parallel machines growth, the mean response time decrease due to the workload division. In addition, the more % of physical CPU utilization, the less mean response time. This fact is due to the increment in CPU resources, through the % of utilization.

Regarding the mean response time in server consolidation, we can observe in tables 2 and 5 its values for the HPI and UIB servers. As the previous case, the mean response time decreases among the number of consolidated virtual machines increases and also the % of CPU utilization. However, the particularity here is the stagnation of the mean response time when a % of CPU is reached. This is due to the fact of the consolidation overhead, which was studied in the previous project and published in [3].

The last studied issue is the CiS^2 index [7]. The values of this index for different number of consolidated machines and different values of % of CPU utilization are depicted in tables 3 and 6. The CiS^2 index aim is to quantify the trade-off between the performance degradation and energy consumption in server consolidation. In previous works, we calculated this index value in CPU saturation, and we concluded that some combinations of virtual machines in some physical servers are inefficient. The same behaviour occurs when the % of CPU is varied.

Table 1: Response time for the HPI server varying the % of CPU for the physical execution

	N		
% CPU	**2**	**4**	**6**
25	451,3935	254,742375	198,5752
50	338,1674	253,713925	197,22865
75	331,56505	245,2905	184,851167
100	335,5619	230,5299	183,731583

Table 2: Response time for the HPI server varying the % of CPU when server consolidation

	N		
% CPU	**2**	**4**	**6**
25	451,3935	254,742375	198,5752
50	338,1674	253,713925	197,22865
75	331,56505	245,2905	184,851167
100	335,5619	230,5299	183,731583

Table 3: CiS^2 index for the HPI server varying the % of CPU when server consolidation

	N		
% CPU	**2**	**4**	**6**
25	0,59631549	0,6340977	0,75584553
50	1,31006444	2,42613798	17,1386902
75	1,62648378	3,02900971	3,32301838
100	2,08608068	3,35418179	4,09212269

Table 4: Response time for the UIB server varying the % of CPU for the physical execution

	N		
% CPU	**2**	**4**	**6**
25	629,30868	240,40758	136,72034
50	355,7786	136,3657	78,84584
75	276,9959	104,20788	60,44646
100	226,99736	84,86738	49,10422

Table 5: Response time for the UIB server varying the % of CPU when server consolidation

	N		
% CPU	**2**	**4**	**6**
25	721,01767	342,20313	304,398312
50	457,688335	343,387975	305,58103
75	458,08443	344,05751	306,37435
100	460,81093	343,37447	306,433495

Table 6: CiS^2 index for the UIB server varying the % of CPU when server consolidation

	N		
% CPU	**2**	**4**	**6**
25	0,70291071	0,54938744	0,92384779
50	0,85523943	1,64513861	2,62568501
75	1,37684598	2,77970367	4,37302116
100	2,06000227	4,10530487	6,52616365

Publications

The use of the HPI infrastructure collaborates to develop the following research works:

1. B. Bermejo, C. Juiz, and C. Guerrero. "On the Linearity of Performance and Energy at Virtual Machine Consolidation: the CiS2 Index for CPU Workload in Server Saturation". In: *Proceedings of the IEEE 20th International Conference on High Performance Computing and Communications*. Exeter, UK, 2018, pages 928–933.

2. B. Bermejo, C. Juiz, and C. Guerrero. "Virtualization and consolidation: a systematic review of the past 10 years of research on energy and performance". In: *The Journal of Supercomputing* 75.2 (2019), pages 808–836. ISSN: 1573-0484. DOI: 10.1007/s11227-018-2613-1.

3. B. Bermejo, C. Juiz, and N. Thomas. "On the virtualization oveead and energy consumption in consolidated servers". In: *UK- Performance Engineering Workshop (UKPEW)*. Newcastle upon Tyne, UK, 2018.

4. B. Bermejo and C. Juiz. "Virtual machine consolidation: a systematic review of its overhead influencing factors". In: *The Journal of Supercomputing* (2019). *Accepted and pending of publication.*

5. C. Juiz and B. Bermejo. "The CiS^2: a new metric for performance and energy trade-off in consolidated servers". In: *Cluster Computing* 23.4 (2020), pages 2769–2788. ISSN: 1573-7543. DOI: 10.1007/s10586-019-03043-8.

6. B. Bermejo and C. Juiz. "On the classification and quantification of server consolidation overheads". In: *The Journal of Supercomputing* (2020), pages 1–21. DOI: 10.1007/s11227-020-03258-2.

4 Next steps

In this project we attempt to answer the research question: "How to determine the performance degradation of physical servers due to Virtual Machine Consolidation when varying the % of physical CPU?". For that, we design a methodology based on monitoring and benchmarking techniques and we apply it to our servers and then, we extend the experiment to HPI infrastructure.

The HPI allows us to extend the work and also to obtain more accurate results and conclusions. Due to that, there are very interesting research questions that we would like to answer with the support to the HPI infrastructure. The question is: "Could we generalize the behaviour of the virtual machine consolidation?"

In order to answer the proposed question, we will apply for a new project in the Hasso Plattner Institute in order to use the HPI Future SOC Lab's IT infrastructure.

References

[1] B. Bermejo, C. Juiz, and C. Guerrero. "On the Linearity of Performance and Energy at Virtual Machine Consolidation: the CiS2 Index for CPU Workload in Server Saturation". In: *Proceedings of the IEEE 20th International Conference on High Performance Computing and Communications*. Exeter, UK, 2018, pages 928–933.

[2] B. Bermejo, C. Juiz, and N. Thomas. "On the virtualization oveead and energy consumption in consolidated servers". In: *UK- Performance Engineering Workshop (UKPEW)*. Newcastle upon Tyne, UK, 2018.

[3] B. Bermejo and C. Juiz. "On the classification and quantification of server consolidation overheads". In: *The Journal of Supercomputing* (2020), pages 1–21. DOI: 10.1007/s11227-020-03258-2.

[4] B. Bermejo, C. Juiz, and C. Guerrero. "Virtualization and consolidation: a systematic review of the past 10 years of research on energy and performance". In: *The Journal of Supercomputing* 75.2 (2019), pages 808–836. ISSN: 1573-0484. DOI: 10.1007/s11227-018-2613-1.

[5] B. Bermejo, S. Filiposka, C. Juiz, B. Gómez, and C. Guerrero. "Improving the Energy Efficiency in Cloud Computing Data Centres Through Resource Allocation Techniques". In: *Research Advances in Cloud Computing*. Edited by S. Chaudhary, G. Somani, and R. Buyya. Singapore: Springer Singapore, 2017, pages 211–236. ISBN: 978-981-10-5026-8. DOI: 10.1007/978-981-10-5026-8_9.

[6] R. Buyya, C. Vecchiola, and S. T. Selvi. *Mastering Cloud Computing: Foundations and Applications Programming*. 1st edition. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013. ISBN: 978-0-12-409539-7.

[7] C. Juiz and B. Bermejo. "The CiS^2: a new metric for performance and energy trade-off in consolidated servers". In: *Cluster Computing* 23.4 (2020), pages 2769–2788. ISSN: 1573-7543. DOI: 10.1007/s10586-019-03043-8.

[8] X. Molero, C. Juiz, and M. Rodeño. *Evaluación y Modelado del Rendimiento del os Sistemas Informáticos*. Prentice Hall, 2004. ISBN: 978-84-205-4093-1.

Implementation of methodological improvements to the detection diabetes mellitus from voice

System to automate reading tests and data collection

Julia Sidorova and Lars Lundberg

Blekinge Institute of Technology
julia.a.sidorova@gmail.com, lars.lundberg@bth.se

In this report we explain an alternative computational analysis to the detection diabetes Type 2 from voice, which is an end-to-end pipeline, the input to which is a speech file and the output is a prediction about its category (diseased or control), and it consists of 1) a feature extraction script to obtain richer representation of the speech signal (6000 parameters in place of less than 20), and 2) learning and testing of a classification function that assigns a category to a new sample. The feature extraction can be used together with the classical statistical analysis currently considered to be the gold standard in the literature on diabetes detection from voice.

1 Introduction

There is a recent research effort to diagnose diabetes with non-invasive methods such as hair analysis, facial expression analysis, and voice acoustic analysis of voice. The aim is building a cost-effective tool for the diagnosis of diabetes mellitus (DM) in the contexts when standard tests are unaffordable or decided against for a different reason . It should be said that non-invasive biomarkers for diabetes are high risk research with many products being retracted or never reaching the market, e.g. in non-invasive glucose monitoring, and therefore the methodological issues need to be addressed with rigour.

The studies on the relation of diabetes and voice rely on a statistical methodology: once the parameters have been measured, the tests for normality are carried out, in order to decide between the use of parametric and nonparametric methods, then the populations of diabetics and controls are compared with respect to the values of the few parameters from the gold standard. Given the research objectives, there are two problems with the approach: 1) none of the studies achieved the detection of DM from voice, and 2) in 3, the differences in voice quality between the diabetics and controls were confirmed via a perceptual analysis (i.e. head), and yet they were not reflected in the values of the parameters from the gold set. These challenges are addressable via the following methodology from the field of voice biomarkers, where the purpose is the detection of the disease from voice.

I. A large number of acoustic features (6000) is extracted from the voice samples by the patients and controls. From this point, a speech sample is represented as a vector with features extracted from the speech file and its class category (diabetic or

control). A feature selection procedure efficiently selects a smaller relevant subset from a huge set of candidate features.

II. From training data a classification function is learnt to classify between the controls (healthy patterns) and patients (pathological ones), and finally the discriminative ability of such a function is tested on the voice samples that were not used during the training stage. A script for the above-described methodology is available from the author on request. The statistical analysis reveals how stable the accuracy will be on new data of the same type, namely, confidence margins as a function of the accuracy of the predictive system and number of speech samples in the test set.

Initially this approach of pattern recognition based on a large number of acoustic features was designed for emotion recognition from voice and then was successfully applied for detection of diverse pathologies and conditions, as was summarized in [1]. The 6000 features were developed over the past decade starting with a couple hundreds features that implemented every possible statistics calculated from the gold set of acoustic parameters (e.g. with the praat system) and augmented with new feature designs, such as mathematically sophisticated features that are a function of a group of features from the original set. The development of such features was an active research field with a centralized effort through Interspeech conferences. Other information such as patient's ethnicity, language, sex, glycemic control, neuropathy and possibly new factors that will be found relevant can also be naturally integrated as features. In pattern recognition, many domains have natural taxonomies, for example species, chemicals, etc. form families and subfamilies. Within a taxonomic category (for example, females or persons with neuropathy), objects have comparable patterns, and for example a hierarchical classification guided by a subgroup indicator can be applied. This methodology also brings the advantage that it can bypass ethnic specificity (if the hypothesis that the voice changes are ethnic-specific turns out to be a correct one), namely, from a large set of features, ethnicity- and language-independent ones may be found. For example, the diagnostics of the Alzheimer's disease was build from a multilingual corpus from a start.

2 Experiments

The example below aims to demonstrate the feature extraction capabilities and classification for a complex physiological and cognitive state, namely, being aware of one's hypoglycemic state. The patient knew her glucose value at the time of speaking. Since the scripts implement a widely used methodology, even though not applied yet to Type 2 diabetes detection, the review of more sophisticated and specified versions of this procedure can be found in [1]. The voice data is provided as an example, in order to be able to install and set up the feature extraction and prediction scripts. The details, analysis and interpretation of the study from which the data comes are subject to a separate publication. The institutional review board (CEIm) approved the study protocol and all patients gave written informed consent before participating in the study. The wav files are in directory VOICE, containing speech

by the patient in the state of hypoglycemia and normal/high glucose values. The 6000 features were extracted using the libraries from the open source feature extraction system openSMILES and is done running the extractFeatures.py script. The output file numericFeatures.csv contains the 6000 numeric values extracted for each speech file. The feature selection and classification relies on the use of the weka open source library for machine learning and is achieved via running the analysis.py script. The balanced accuracy per class achieved is 73 %, the mode of testing was 10-fold cross-validation.

3 Conclusion

Based on the a literature analysis of T2D and voice, we have explained an alternative computational analysis and included a script that implements its vanilla version as supplementary material. The computational analysis is an end-to-end analysis, the input to which is a speech file and the output is a prediction about its category (diseased or control), and it consists of 1) a feature extraction script to obtain richer representation of the speech signal (6000 parameters in place of less than 20), and 2) learning and testing of a classification function that assigns a category to a new sample. The feature extraction can be used together with the classical statistical analysis.

References

[1] J. Sidorova, S. Karlsson, O. Rosander, M. L. Berthier, and I. Moreno-Torres. "Towards Disorder-Independent Automatic Assessment of Emotional Competence in Neurological Patients with a Classical Emotion Recognition System: Application in Foreign Accent Syndrome". In: *IEEE Transactions on Affective Computing* (2019). DOI: 10.1109/TAFFC.2019.2908365.

Behavior-based authentication

Feature engineering and performance evaluation based on large user profiles

Vera Weidmann, Leon Lowitzki, and Marvin Mirtschin

neXenio GmbH
vera.weidmann@nexenio.com
leon.lowitzki@nexenio.com
marvin.mirtschin@nexenio.com

1 Introduction

Our project contributes to the growing interest of mobile and behavior-based authentication systems. Next to fingerprints and facial features, another meaningful authentication method is a person's gait pattern. Our smartphones are equipped with a variety of sensors and these can measure the environment and behavior surrounding our phones. We carry these phones with us around the clock, even closely attached to our bodies. Accordingly, data corresponding to our body movement is generated. Research shows that these recorded data signals can be shaped to a unique signature of the phone's owner. The authentication accuracy, though, is highly dependent on the users' physiological and environmental circumstances. The more situations that are covered by the user's training state, the more the system's precision is challenged. In this context, statistical investigations and machine learning algorithms are applied.

Our research team at neXenio confronts behavior-, in particular gait-based authentication, with an enormous data set, covering six thousand walking sequences that have been recorded over the last years.

This semester, our third period at Future SOC Lab, we focused on signal processing and feature engineering.

2 Behavior-based authentication

Nowadays, smartphones can be unlocked via password, fingerprint, or, as more recently introduced, face recognition. The last two methods are biometric authentication methods, which use user related characteristics, e.g. the friction ridges of the finger or facial features, to recognize people.

Nowadays, another biometric generates interest and enthusiasm: gait authentication. Here, the user's motions, such as the leg's forward, backward or sideways

movements, are observed by sensors. The next sections introduce the general methodology for gait-authentication as well as neXenio's course of action in this field.

2.1 Methodology for gait-based authentication

Nowadays, gait-based authentication methods are well researched. Especially walking sequences, recorded by a smartphone or smart watches' inbuilt sensors, such as the accelerometer and gyroscope sensor, are reflected. Sprager and Juric [9] as well as Connor [2] summarized the state of the art methodologies used in this field. Guidelines for general data processing are stated by [1] and [3].

In brief, a walking sequence is cut in pieces and summarized by an average gait cycle. This cycle is used as a template and the dissimilarity of a new step is concluded by different distance metrics, such as Euclidean, Hamming, Manhattan or Tanimoto distance. Other approaches use statistical measurements in the time and frequency domain to gather descriptive information about walking sequences. Calculations such as mean, median, standard deviation, third and fourth order cumulants, skewness and kurtosis, and distribution parameters, such as a ten-bin histogram are used ([5, 7, 8]). Likewise, we found features such as the root mean squared, median absolute deviation, average absolute variation, quantiles, interquartile range, correlations and zero-crossing in several papers. In the frequency domain, Fourier and cepstral coefficients, discrete cousin transformations and wavelets are used [9]. Next to principal component analysis, Gait Dynamic Images [12] or geometric template matching were applied to gait sequences [4].

Besides cycle-based assignments, different machine learning techniques are used for authentication: linear and logistic regressions, k-nearest neighbors, support vector machines (SVM), hidden markov models and convolutional neuronal networks.

2.2 seamless.me @ neXenio

neXenio GmbH is a small company located in the middle of Berlin. They take up the challenge of developing high secure IT-solutions that address data sharing and virtual collaboration needs of digital workspaces. One of their products called SEAMLESSme targets the idea of behaviour-based authentication. The most important difference to other authentication systems and research approaches is that neXenio's implementation focuses on the premise of data privacy and security. Since data is processed locally on the device itself, data is never sent to external computation resources. Thus, the underlying classification approach is required to be a one-classification problem. An algorithm is deployed that only considers data from a single person. This training set is neither polluted by any outlier nor is enriched by data from other users. The algorithm must detect whether a new unknown walking observation fits to the learned pattern. In general, this type of classification shows less precise results than binary or multi-class classifications as the difference between users are not known. Only very few research studies considered this classification type, e.g. by using one-class SVMs.

Another challenge regarding local processing is the device's battery drain. Therefore, neXenio implemented a more efficient outlier technique instead of widely utilized battery-expensive algorithms. This outlier recognition algorithm is based on multi-level hierarchical nested histograms. Each histogram creates a discrete frequency distribution of the sensor data's processed features to a specific grain. Features, mentioned in the previous section 2.1, were evaluated and assembled in a way that the best authentication performances were attained.

3 The Problem of learning from a variety of gait patterns

Real-world gait-based authentication applications have to deal with the variety of natural gait forms. If a specific, but genuine form is unknown by the algorithm, it can hardly be assigned to the user. Gait-affecting factors are of physiological or environmental nature. While physiological factors induce more unconscious circumstances, such as permanent gait abnormalities or temporal changes caused by mood, environmental factors are represented by clothing, shoes, surfaces, slopes or obstacles [9]. Real world behavior-based authentication systems need to be robust for long-term utilization as these factors can change by varying degrees from time to time.

Public datasets for gait recognition do not imply a comprehensive overview about a user's possible walking situations. While OU-ISIR Biometric Database of Osaca University [6] is the largest database in the field of gait recognition, holding nearly 750 subjects, just one environmental factor, inclined terrain, was recorded in one session. Frank et al. [4] published another dataset in cooperation with the McGill University. This database contains just 20 participants, but data is recorded in the wild in two distinct sessions, meaning that people could have worn different clothes and shoes per session. Research that relies on this database, detected the effect of clothes the most. In all analyses the authentication performance suffers in cross-day comparison, particularly when people had a major change in trouser type ([4, 10]). This effect is also shown by the larger dataset of Subramanian et al. [11].

Purpose of Future SOC Lab utilization

In the last years, we collected a great amount of data files, covering this variety of physiological and environmental forms. At the moment, our statistical investigations as well as our novelty detector, which we use for user classification, can be just tested by dividing our dataset into smaller subsets. An evaluation that comprises all walking circumstances is not feasible as the limits of our machines according to RAM are reached.

By applying for utilization of the HPI Future SOC Lab we aimed to improve our authentication approach, refine signal preprocessing, feature engineering and modeling in regard to meet stability for wide user profiles.

4 Usage of Future SOC Lab resources

Again, Future SOC Lab provided us with an isolated server, allowing us to process all of our data in parallel. This was one of its main advantages as it greatly sped up our analyses. Even ad-hoc analyses, which evaluate the impact of e.g. an additional filter method, were possible.

In the beginning of this period, we refactored our whole process of data transformation. We connected the server with our database, piped this data to our feature transformers and scored the performance by the loss function equal error rate. Cross validation and grid search functions were used, both possible through parallel computing, taking advantage of SOC Lab's core clusters. Therefore, feature ranking as well as model parameters were evaluated and ranked automatically in almost 60 minutes.

Similar to last semester we ran into a high usage of RAM (>500 GB) when combining all our recordings into one pandas data frame. This is not even suitable for the provided server. Hence, we implemented two ways of processing. The first enables us to run a small but fast analysis, the second a large but slow one, as data is pulled for single cross validation splits separately (~20 GB).

5 Evaluation

5.1 Performance metrics

For classification objectives, generally, the true positives, false positives (also called false matches; imposters get authenticated), false negatives (called false non-matches; genuine sequences do not match) and true negatives are calculated for evaluating the performance of an algorithm. Biometric algorithms are ranked by the equal error rate (EER). This metric reflects the closest point of false matches and false non-matches. The lower the equal error rate value, the higher the accuracy of the biometric system.

5.2 Findings

Our new transformation pipeline enables us to evaluate the authentication system much more precisely than in the last semesters. Even if our result looks slightly worse, it covers much more of the user's walking reality. Compared to other research studies that cover gait-affecting factors, our rate is still satisfying.

We achieved an EER of 16 % while keeping the phones' pocket position fixed. Using data of different pockets increases the error to 25 %. This problem will be addressed in our future work.

6 Conclusion and future work

The resources of HPI's Future Soc Lab enabled us to intensify our data analysis and improve the authentication model. Only by using the advantage of the server's parallel computing, could we process our data — and even speed up the total run time.

At the moment, we are developing an approach similar to "eigenfaces" and "eigensteps", combining principal components and support vector machines, but in the context of a one class classification problem. Furthermore, we are implementing a coordinate transformation that converts a phone's orientation in a trouser pocket into a resilient and unbiased device and environment independent coordinate system. We expect to make a statement about whether there is an improvement in the result compared to the calculation of the acceleration's magnitude. In this regard, we look forward to using the resources of Future SOC Lab in the next semester again. We want to thank the Future SOC Lab team for the great support.

References

[1] A. Buriro, Z. Akhtar, B. Crispo, and S. Gupta. "Mobile Biometrics: Towards A Comprehensive Evaluation Methodology Mobile Biometrics : Towards A Comprehensive Evaluation Methodology". In: *2017 International Carnahan Conference on Security Technology (ICCST)*. 2017. ISBN: 978-1-5386-1585-0. DOI: 10.1109/CCST.2017.8167859.

[2] P. Connor and A. Ross. "Biometric recognition by gait: A survey of modalities and features". In: *Computer Vision and Image Understanding* 167 (2018), pages 1–27. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2018.01.007.

[3] R. Ferrero, F. Gandino, B. Montrucchio, M. Rebaudengo, A. Velasco, and I. Benkhelifa. "On gait recognition with smartphone accelerometer". In: *Proceedings - 2015 4th Mediterranean Conference on Embedded Computing, MECO 2015 - Including ECyPS 2015, BioEMIS 2015, BioICT 2015, MECO-Student Challenge 2015* (2015), pages 368–373. DOI: 10.1109/MECO.2015.7181946.

[4] J. Frank, S. Mannor, J. Pineau, and D. Precup. "Time Series Analysis Using Geometric Template Matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.3 (2013), page 1. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.121.

[5] T. Hoang, D. Choi, and T. Nguyen. "Gait authentication on mobile phone using biometric cryptosystem and fuzzy commitment scheme". In: *International Journal of Information Security* 14.6 (2015), pages 549–560. ISSN: 1615-5270. DOI: 10.1007/s10207-015-0273-1.

[6] T. T. Ngo, Y. Makihara, H. Nagahara, Y. Mukaigawa, and Y. Yagi. "The largest inertial sensor-based gait database and performance evaluation of gait-based

personal authentication". In: *Pattern Recognition* 47.1 (2014), pages 228–237. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2013.06.028.

[7] C. Nickel. "Accelerometer-based Biometric Gait Recognition for Authentication on Smartphones". PhD thesis. Darmstadt: Technischen Universität Darmstadt, June 2012. URL: http://tuprints.ulb.tu-darmstadt.de/3014/.

[8] C. Nickel and C. Busch. "Classifying accelerometer data via hidden Markov models to authenticate people by the way they walk". In: *IEEE Aerospace and Electronic Systems Magazine* 28.10 (2013), pages 29–35. ISSN: 0885-8985. DOI: 10.1109/MAES.2013.6642829.

[9] S. Sprager and M. Juric. "Inertial Sensor-Based Gait Recognition: A Review". In: *Sensors* 15.12 (Sept. 2015), pages 22089–22127. ISSN: 1424-8220. DOI: 10.3390/s150922089.

[10] S. Sprager and M. B. Juric. "An Efficient HOS-Based Gait Authentication of Accelerometer Data". In: *IEEE Transactions on Information Forensics and Security* 10.7 (2015), pages 1486–1498. ISSN: 1556-6013. DOI: 10.1109/TIFS.2015.2415753.

[11] R. Subramanian, S. Sarkar, M. Labrador, K. Contino, C. Eggert, O. Javed, J. Zhu, and H. Cheng. "Orientation invariant gait matching algorithm based on the Kabsch alignment". In: *2015 IEEE International Conference on Identity, Security and Behavior Analysis, ISBA 2015* (2015), pages 1–8. DOI: 10.1109/ISBA.2015.7126347.

[12] Y. Zhong, Y. Deng, and G. Meltzner. "Pace independent mobile gait biometrics". In: *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems, BTAS 2015* (2015), pages 1–8. DOI: 10.1109/BTAS.2015.7358784.

Measurement-Based Software Performance Engineering for Microservices and Multi-Core Systems

Project Report for HPI Future SOC Period Fall 2019

André van Hoorn, Markus Frank, and Henning Schulz

Institute of Software Technology
University of Stuttgart

This report provides a summary of our project "Measurement-Based Software Performance Engineering for Microservices and Multi-Core Systems" conducted during the HPI Future SOC Lab period spring 2019.

1 Project Idea

Our project was divided into two subprojects, namely (1) "DevOps-oriented Load Testing for Microservices" and (2) "Software Performance Engineering for Multi-Core Systems". Both subprojects are a direct continuation of the works that we started in the previous periods.

In the remainder of this report, we will provide some more details about the project context (sections 1.1 and 1.2), list the granted Future SOC Lab resources (section 2), provide a brief description of our findings (section 3), and outline next steps (section 4).

1.1 DevOps-oriented Load Testing for Microservices

Modern software engineering paradigms and technologies — such as DevOps [3] (including automation as part of continuous delivery) and microservices [15] — are gaining more and more attraction in the software and services engineering communities. Of particular interest are quality-of-service concerns, for instance, w. r. t. performance and reliability. While established approaches for classic contexts (i.e., which do not use DevOps and microservices) exist, their adoption to DevOps and microservices requires considerable research efforts [4, 5, 6, 14].

Load testing is a measurement-based approach to assess performance-related properties of software systems. In this project, we focus on the experimental evaluation of DevOps-oriented approaches for load testing microservices.

In the past periods, we have already conducted experiments for our newly developed approaches in the HPI infrastructure (e.g. [2, 16]). The activities on load testing conducted during this period were a direct continuation of the activities started during the previous periods.

1.2 Software Performance Engineering for Multi-Core Systems

Multicore systems are a permanent part of our daily life. Regardless of whether we consider nowaday's desktop PCs, notebooks, or smart phones—all devices are running on multicore CPUs. To use these hardware features in an efficient way, developers need to build parallel-enabled software. However, the development of such software is more complex than developing sequential software.

To handle the rising complexity, it is necessary to develop software in an engineering-like way. In such a process, software architects plan and analyze software designs on a model level. Software architects can use tools like Palladio to simulate and analyze early-phase software designs. Unfortunately, current approaches and tools lack the ability to consider multicore systems. Therefore, in this subproject, we aim to find performance prediction methods for multicore systems in the context of our ongoing research [9, 10, 11, 12].

In the previous periods, we have started to use the HPI infrastructure to study performance properties for performance predictions of multi-core systems (e.g. [7, 13]).

2 Used Future SOC Lab resources

We requested and received dedicated (root) access to the following computing resources (servers):

1. 896 GB RAM, 80 cores
2. 32 GB RAM, 24 cores

Dedicated access has been given to us due to our expected high resource demands.

3 Findings

We have worked on the previously stated goals in both subprojects. In this section, we will provide a summary of the experiments and results for both subprojects.

3.1 DevOps-oriented Load Testing for Microservices

In previous periods, we have evaluated our approach on domain-based scalability assessment of microservices in the HPI cluster. During the previous period we have extended the publication by further experiments into a journal submission, which has been accepted [1].

3.2 Software Performance Engineering for Multi-Core Systems

Based on the process proposed by Wert et al. [17], we continued the controlled experiments started in the previous period [8] to measure the behavior of performance-influencing factors of highly parallel software on multi-core architectures. Therefore, we executed various resource demands raging from processor-intensive to I/O-intensive on different hardware configurations. We used four different parallelization paradigms and measured memory bandwidth, cache behavior, and speedup.

Figure 1 shows the Level 2 Cache behavior of the large HPI server when using OpenMP as parallelization paradigm.

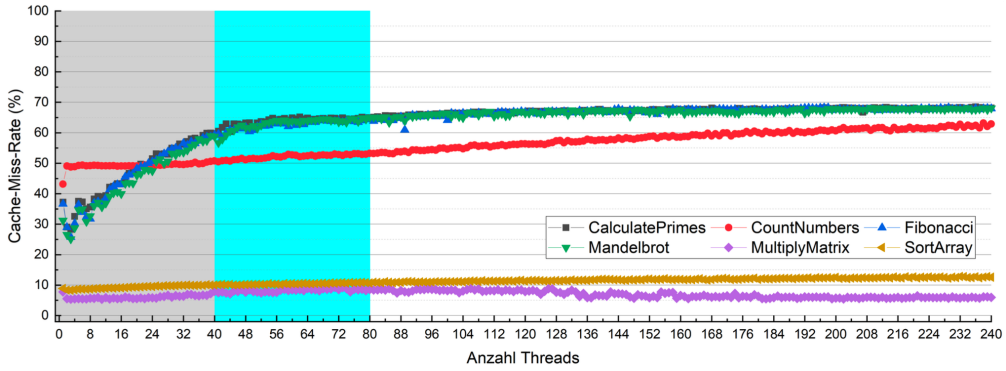


Figure 1: Comparison of cache behavior

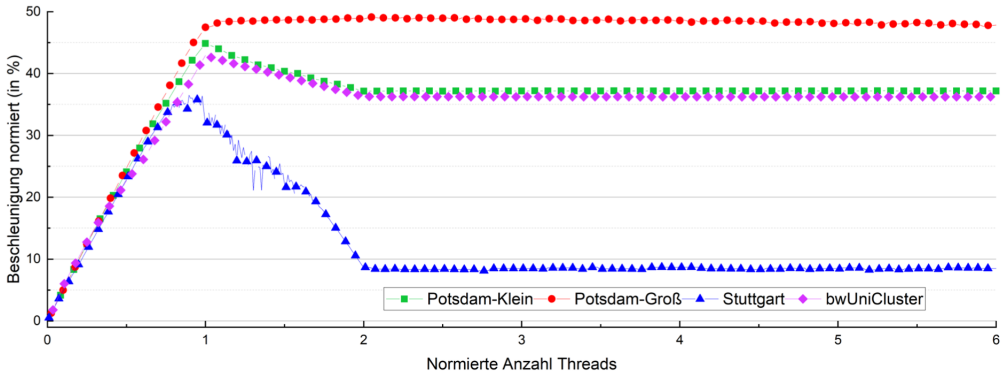


Figure 2: Performance comparison of hardware configurations

Besides various insights of the performance-influencing factors we were able to compare the performance of different hardware configurations in the sense of their speedup capabilities (see figure 2). A conference paper is in preparation.

In the future, we plan to extract performance curves from the measurements to build a performance prediction model for parallel software and thereby help software architects to increase the (currently inadequate) accuracy of parallel performance prediction approach.

4 Next steps

We will continue our ongoing research in the two subprojects. To have the ability to execute extensive experiments, we will apply for the next HPI Future SOC Lab period. We plan to extend our load testing approaches to Functions-as-a-Service (FaaS). Extensions for the multi-core experiments include additional configurations and more complex parallel programs.

Acknowledgment

We are thankful to the HPI Future SOC Lab for having granted us access to the computing infrastructure. The environment eases the joint work of different organizations on the platform, which has so far been hindered by university-internal access constraints—apart from the fact that an equipment comparable to that of the HPI Future SOC Lab has not been available to us, and that enables extensive performance configuration tests needed to reach our goals.

References

[1] A. Avritzer, V. Ferme, A. Janes, B. Russo, A. van Hoorn, H. Schulz, D. Menasché, and V. Rufino. "Scalability Assessment of Microservice Architecture Deployment Configurations: A Domain-based Approach Leveraging Operational Profiles and Load Tests". In: *Journal of Systems and Software* 165 (2020).

[2] A. Avritzer, V. Ferme, A. Janes, B. Russo, H. Schulz, and A. van Hoorn. "A quantitative approach for the assessment of microservice architecture deployment alternatives using automated performance testing". In: *Proceedings of the 12th European Conference on Software Architecture (ECSA 2018)*. LNCS. Springer, 2018.

[3] L. J. Bass, I. M. Weber, and L. Zhu. *DevOps : A Software Architect's Perspective*. SEI series in software engineering. Addison-Wesley, 2015. ISBN: 978-0-13-404984-7.

[4] C. Bezemer, S. Eismann, V. Ferme, J. Grohmann, R. Heinrich, P. Jamshidi, W. Shang, A. van Hoorn, M. Villaviencio, J. Walter, and F. Willnecker. "How is

Performance Addressed in DevOps? A Survey on Industrial Practices". In: *Proceedings of the ACM/SPEC International Conference on Performance Engineering (ICPE 2019)*. ACM, 2019, pages 45–50.

[5] A. Brunnert, A. van Hoorn, F. Willnecker, A. Danciu, W. Hasselbring, C. Heger, N. Herbst, P. Jamshidi, R. Jung, J. von Kistowski, A. Koziolek, J. Kroß, S. Spinner, C. Vögele, J. Walter, and A. Wert. *Performance-oriented DevOps: A Research Agenda*. Technical report SPEC-RG-2015-01. SPEC Research Group — DevOps Performance Working Group, Standard Performance Evaluation Corporation (SPEC), 2015.

[6] S. Eismann, C.-P. Bezemer, W. Shang, A. van Hoorn, and D. Okanović. "Microservices: A Performance Tester's Dream or Nightmare?" In: *Proceedings of the 11th ACM/SPEC International Conference on Performance Engineering (ICPE 2020)*. ACM, 2020.

[7] M. Frank, S. Becker, A. Kaplan, and A. Koziolek. "Performance-influencing Factors for Parallel and Algorithmic Problems in Multicore Environments". In: *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering (ICPE '19)*.

[8] M. Frank, S. Becker, A. Kaplan, and A. Koziolek. "Performance-Influencing Factors for Parallel and Algorithmic Problems in Multicore Environments: Work-In-Progress Paper". In: *Companion of the 2019 ACM/SPEC International Conference on Performance Engineering (ICPE 2019)*. ACM, 2019, pages 21–24. DOI: 10.1145/3302541.3313099.

[9] M. Frank and M. Hilbrich. "Performance Prediction for Multicore Environments:An Experiment Report". In: *Proceedings of the Symposium on Software Performance (SSP 2016)*. 2016.

[10] M. Frank, M. Hilbrich, S. Lehrig, and S. Becker. "Parallelization, Modeling, and Performance Prediction in the Multi-/Many Core Area: A Systematic Literature Review". In: *Proceedings of the 7th IEEE International Symposium on Cloud and Service Computing (SC2 2017)*. 2017.

[11] M. Frank, F. Klinaku, and S. Becker. "Challenges in Multicore Performance Predictions". In: *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering (ICPE 2018)*. ACM, 2018. DOI: 10.1145/3185768.3185773.

[12] M. Frank, S. Staude, and M. Hilbrich. "Is the PCM Ready for ACTORs and Multicore CPUs? — A Use Case-based Evaluation". In: *Proceedings of the 8th Symposium on Software Performance (SSP 2017)*. 2017.

[13] P. Gruber and M. Frank. "Modelling and Predicting Memory Behavior in Parallel Systems with Network Links: Palladio-based Experiment Report". In: *Proceedings of the 10th Symposium on Software Performance (SSP 2019)*. 2019.

[14] R. Heinrich, A. van Hoorn, H. Knoche, F. Li, L. E. Lwakatare, C. Pahl, S. Schulte, and J. Wettinger. "Performance Engineering for Microservices: Research Challenges and Directions". In: *Companion of the 8th ACM/SPEC International Conference on Performance Engineering* (*ICPE 2017*). ACM, 2017. ISBN: 978-1-4503-4899-7. DOI: 10.1145/3053600.3053653.

[15] S. Newman. *Building Microservices*. O'Reilly Media, Inc., 2015.

[16] H. Schulz, T. Angerstein, D. Okanović, and A. van Hoorn. "Microservice-tailored Generation of Session-based Workload Models for Representative Load Testing". In: *Proceedings of the 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems* (*MASCOTS 2019*). To appear. 2019.

[17] A. Wert, J. Happe, and D. Westermann. "Integrating Software Performance Curves with the Palladio Component Model". In: *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering* (*ICPE 2012*). ACM, 2012, pages 283–286.

CitySensing: Big Mobility Data Analytics for Traffic Monitoring and Control

Dragan Stojanovic, Natalija Stojanovic, and Aleksandra Stojnev Ilic

Computer Science and Engineering Department
University fo Nis, Faculty of Electronic Engineering
{firstname.lastname}@elfak.ni.ac.rs

The research and development in CitySensing project have been focused on Big mobility data processing and analytics for traffic monitoring and control, as well as air pollution analytics with regard to traffic. We analyse Big streaming car data to detect traffic conditions and events, such as travel times along the road segments and traffic congestions. Upon detecting the traffic congestion at particular streets/area, the system can reconfigure and adapt the traffic (e.g. semaphore lights duration, speed limit) to increase the traffic flow in critical directions. Also, the air pollution measurements along the road infrastructure provides correlation of heavy traffic and traffic jams with increasing air pollution, which provides prediction of air pollution depending on traffic characteristics. We have tested and evaluated developed Spark Streaming applications on the FSOC Lab cluster infrastructure varying streaming workload and cluster parameters.

1 Introduction

The increase in the amount and availability of Big mobility and IoT data has fueled the vision of Smart Cities that will improve every aspect of our urban lives, transport, health, energy use, environment, etc [2]. The activities within CitySensing project are focused on research and development of methods, techniques, software systems and applications for monitoring, analysis and control of Smart City mobility and traffic, based on processing, analysis and visualization of Big streaming data collected by moving vehicles and air pollution sensors.

We have improved and extended the TrafficSense application, and developed TrafficSense&Control system for traffic monitoring and control that employs Big vehicle mobility data, as well as street network and POI data to provides traffic analytics and detection of important traffic events and conditions, such as:

- Average travel times over street segments,
- Slow traffic and traffic jams,
- Traffic stop for a longer time,
- Heavy traffic along particular street segments.

We have implemented two TrafficSense&Control applications. The first one, upon detection of traffic conditions and events, can change and adapt the traffic and the traffic flow leveraging feedback control loop principle. The control and adaptation are performed by changing the duration of particular semaphore lights at congested intersections and changing the street parameters (one-way, closing/open lanes, speed limit, etc.). In that way, a traffic self-adaptation is achieved that reduces traffic congestions and heavy traffic conditions.

Within the second TrafficSense&Control application, by collecting, processing and analysis of air pollution data, the application is able to produce the correlation between the heavy traffic conditions and air pollution information (ozone, particulate matters, carbon monoxide, sulfur dioxide, nitrogen dioxide) obtained through Smart City sensor infrastructure. Such correlation enables generation of a machine learning model that could predict the air pollution level depending on the number of vehicles in particular areas and street, their average speed and congestion. The prediction of possible air pollution could provide feedback control of the traffic flow, by closing some street lanes, reducing the traffic flow, and rearranging the traffic to reduce air pollution in certain areas.

We have implemented, tested and evaluated TrafficSense&Control applications using open-source Big Data technologies, such as Apache Spark,[1] Hadoop/HDFS,[2] Kafka[3] and Cassandra[4] NoSQL data system.

2 FSOC Lab Infrastructure for Big Traffic Data Analytics

For the purpose of deploying, testing and evaluation of TrafficSense&Control applications for Big streaming data processing and analytics, we use the following Future SOC Lab infrastructure:

- Multi-core computer based on Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20 GHz, with two sockets, 40 CPU cores, 256 GB RAM.
- The cluster of 9 virtual machines: 1 master (XL VM - 8 CPU Cores and 8 GB RAM), 8 slaves (L VM - 4 CPU Cores and 4 GB RAM).
- Multi GPU NVIDIA Tesla K80 GPUs.

Apache Hadoop, Spark, Kafka and Cassandra were installed and configured on both platforms and the containerization of Spark and Cassandra using Docker have been performed on the multi-core computer. Both applications were developed using Python programming language.

[1] https://spark.apache.org (last accessed 2020-01-01).
[2] https://hadoop.apache.org/ (last accessed 2020-01-01).
[3] https://kafka.apache.org/ (last accessed 2020-01-01).
[4] https://cassandra.apache.org/ (last accessed 2020-01-01).

3 Traffic Analysis and Control

The first TrafficSense&Control application was implemented using Apache Structured Streaming framework. It receives the Big mobility streaming data simulating city traffic, via Kafka message broker. For the realistic traffic data generation we use SUMO, an open source, microscopic and continuous traffic simulator, developed by the Institute of Transportation Systems at the German Aerospace Center. To provide the control of the simulation and traffic parameters TRACI (Traffic Control Interface) is used, giving access to a running road traffic simulation, allowing retrieval of simulated objects values and manipulation their behaviour "online". The self-adaptation of the traffic within TrafficSense&Control application is based on CrowdNav simulation tool that extends SUMO and implements reconfiguration of simulation and traffic parameters using Kafka messages while the simulation is running [1]. The map matching of the vehicles' locations on the street segments has been performed in advance on multi GPU NVIDIA Tesla K80 GPUs.

Docker images for CrowdNav/SUMO and Kafka have been created and deployed at the master node (XL virtual machine), so CrowdNav/SUMO and Kafka run as Docker containers. Traffic simulation data generated by CrowdNav/SUMO are sent to dedicated Kafka topic and consumed by Spark Structured Streaming Jobs of the TrafficSense&Control application. The application continuously processes and analyses incoming floating car streaming data, and based on OpenStreetMap street network data (City of Nis), continuously calculates the average speed along street segments, the number of vehicles on street segments, and traffic congestions on street intersections. Upon detection of congested street segments and intersections, the application send appropriate messages to CrowdNav/SUMO simulator via Kafka, to change and adapt simulation parameters via TraCI while simulation is running: the duration of street lights in certain directions at intersections, speed limits in crowded streets, traffic mode (one/two ways) for certain street segments, etc. This reconfiguration performs the feedback control loop mechanism and cause self-adaptation of traffic to overcome traffic congestions and heavy traffic [4].

TrafficSense&Control application provides dashboard functionality through real-time Web application for monitoring of traffic and mobility of cars in a Smart City. To provide dynamic visualisation of traffic congestions occurring in the city, as well as slow traffic in particular street segments, Eclipse Vert.x (vertx.io), an event-driven application framework, has been used. We have implemented the Vert.x service and the Web application connected trhough Web Socket interface that visualizes the traffic congestions (heat maps), and dynamic travel times along the street segments, as results of processing and analysis of Spark Structured Streaming jobs.

The TrafficSense&Control application are executed using one Spark Driver running at the Master node and eight Spark Executors running at Worker nodes. We have also evaluated the maximum workload generated by 10 000 simulated vehicles, reporting their positions each second, generating more than 250 millions of messages sent to Kafka during the duration of the simulation (~40 min). The peak system throughput is about 10 550 messages/second. The Spark UI (figure 1) shows the Spark Executors in action.

Executors

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Logs	Thread Dump
driver	192.168.42.160:36307	Active	0	8.4 MB / 956.6 MB	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B		Thread Dump
6	192.168.42.64:39415	Active	0	8.3 MB / 1.2 GB	0.0 B	4	5	0	323	328	3.0 min (12 s)	0.0 B	514.5 KB	766.1 KB	stdout stderr	Thread Dump
0	192.168.42.32:35094	Active	0	8.3 MB / 1.2 GB	0.0 B	4	5	0	359	364	2.8 min (9 s)	0.0 B	579.2 KB	782.9 KB	stdout stderr	Thread Dump
2	192.168.42.163:35503	Active	0	8.3 MB / 1.2 GB	0.0 B	4	5	0	377	382	3.1 min (13 s)	0.0 B	561.3 KB	770.1 KB	stdout stderr	Thread Dump
7	192.168.42.118:33322	Active	0	8.3 MB / 1.2 GB	0.0 B	4	5	0	467	472	3.2 min (9 s)	0.0 B	767.9 KB	776.9 KB	stdout stderr	Thread Dump
1	192.168.42.48:33496	Active	0	8.3 MB / 1.2 GB	0.0 B	4	4	0	471	475	3.5 min (11 s)	0.0 B	735.1 KB	774.1 KB	stdout stderr	Thread Dump
3	192.168.42.165:43683	Active	0	8.3 MB / 1.2 GB	0.0 B	4	4	0	543	547	3.3 min (13 s)	0.0 B	883.8 KB	783.3 KB	stdout stderr	Thread Dump
4	192.168.42.174:46829	Active	0	8.3 MB / 1.2 GB	0.0 B	4	5	0	579	584	3.5 min (10 s)	0.0 B	881.8 KB	768.5 KB	stdout stderr	Thread Dump
5	192.168.42.172:43591	Active	0	8.4 MB / 1.2 GB	0.0 B	4	4	0	634	638	3.7 min (10 s)	0.0 B	988.2 KB	779.1 KB	stdout stderr	Thread Dump

Figure 1: Spark Executors in action

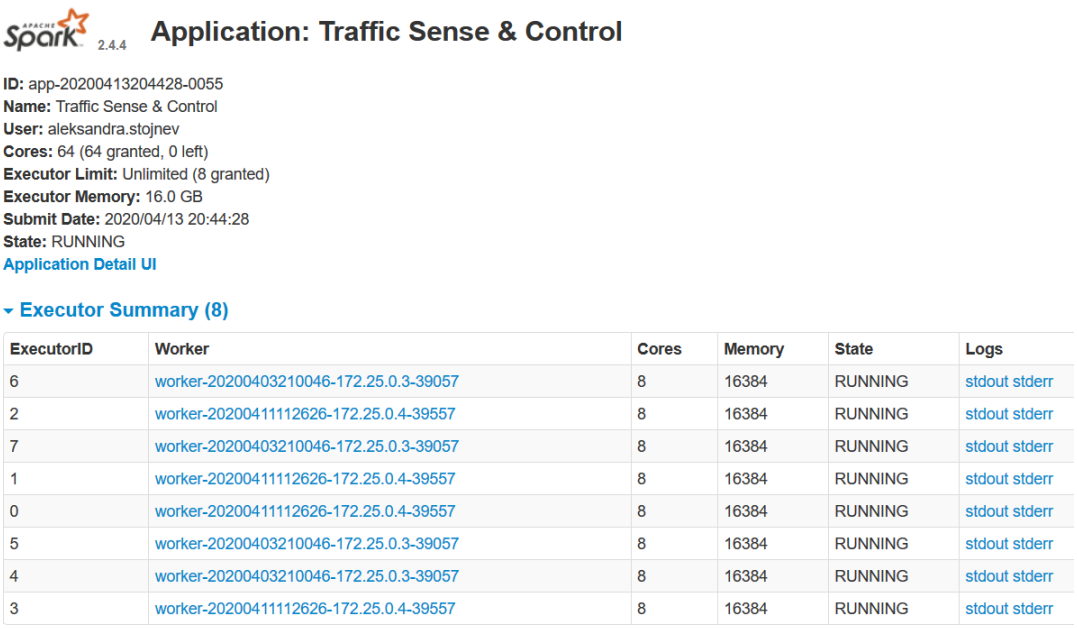
4 Traffic Analysis and Air Pollution

The second TrafficSense&Control application finds the correlation between streaming traffic mobility data and air pollution data from a city sensor network. To simulate real-world scenario we have used CityPulse open dataset[5] representing the vehicle traffic, observed between two points in a street network for a duration of time in the city of Aarhus in Denmark. Also, a collection of pollution measurements designed to complement the vehicle traffic dataset, is used, in the form of pollution mockup stream that simulates one sensor for each of the traffic sensor at the exact location of this traffic sensor [3]. Both traffic and pollution datasets are available in raw, CSV format. To simulate real-time behavior, a helper applications have been developed that reads from sensor data files (traffic and pollution datasets) and sends (publishes) the traffic and pollution data at regular intervals to the Kafka broker, to be consumed by the Spark Streaming application as a subscriber.

The application has two modes of work. The offline part of the application uses machine learning technique (linear regression, random forest) to predict the air pollution level splitting the parts of the traffic and pollution data in training and test datasets. The goal of the offline phase is to generate the model that could predict the level of air pollution depending on the amount of traffic in certain streets/area (the number of cars and average speed). The online part of the application simulates the streaming of the vehicle mobility data and continuously predicts the air pollution level with the ultimate goal to re-arrange and adapt the traffic in order to decrease the level of air pollution in certain streets/areas. The Web application for visualization of streaming traffic and air pollution data and their correlation is currently underway.

[5]https://iot.ee.surrey.ac.uk:8080/datasets.html (last accessed 2020-01-01).

The application is tested and evaluated on the multi-core computer with 40 CPU cores using three Spark Docker containers, one Master and two Worker nodes, that can execute 8 Spark Executors with 8 CPU cores and 16 GB RAM (figure 2).



Application: Traffic Sense & Control

ID: app-20200413204428-0055
Name: Traffic Sense & Control
User: aleksandra.stojnev
Cores: 64 (64 granted, 0 left)
Executor Limit: Unlimited (8 granted)
Executor Memory: 16.0 GB
Submit Date: 2020/04/13 20:44:28
State: RUNNING
Application Detail UI

Executor Summary (8)

ExecutorID	Worker	Cores	Memory	State	Logs
6	worker-20200403210046-172.25.0.3-39057	8	16384	RUNNING	stdout stderr
2	worker-20200411112626-172.25.0.4-39557	8	16384	RUNNING	stdout stderr
7	worker-20200403210046-172.25.0.3-39057	8	16384	RUNNING	stdout stderr
1	worker-20200411112626-172.25.0.4-39557	8	16384	RUNNING	stdout stderr
0	worker-20200411112626-172.25.0.4-39557	8	16384	RUNNING	stdout stderr
5	worker-20200403210046-172.25.0.3-39057	8	16384	RUNNING	stdout stderr
4	worker-20200403210046-172.25.0.3-39057	8	16384	RUNNING	stdout stderr
3	worker-20200411112626-172.25.0.4-39557	8	16384	RUNNING	stdout stderr

Figure 2: Traffic Sense & Control application in action

5 Conclusion

The work presented in this report shows the viability of Big mobility data processing and analytics for traffic monitoring and control, as well as prediction of air pollution. The streaming vehicle data can be analyzed in real-time to detect traffic conditions and events, such as travel time along the road segments and traffic congestion. Upon detection of traffic congestions, the TrafficSense&Control system can reconfigure and adapt the traffic to increase the traffic flow in critical directions and avoid further congestion. Developing the machine learning model, the system can define the correlation between heavy traffic and traffic congestions with increasing air pollution, providing possibility to reduce the air pollution by reducing and adapting the traffic in the critical area. The testing and evaluate of developed Spark Streaming applications on the FSOC Lab cluster infrastructure with varied streaming workload and cluster parameters prove the feasibility of our system and approach and provide directions for further research and useful insights in deploying and execution of Big data applications on the real cloud infrastructure.

References

[1] S. Amini, I. Gerostathopoulos, and C. Prehofer. "Big data analytics architecture for real-time traffic control". In: *5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2017, Naples, Italy, June 26-28, 2017*. IEEE, 2017, pages 710–715. DOI: 10.1109/MTITS.2017.8005605.

[2] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. Siddiqa, and I. Yaqoob. "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges". In: *IEEE Access* 5 (2017), pages 5247–5261. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2017.2689040.

[3] D. Puiu, P. M. Barnaghi, R. Toenjes, D. Kuemper, M. I. Ali, A. Mileo, J. X. Parreira, M. Fischer, S. Kolozali, N. FarajiDavar, F. Gao, T. Iggena, T. Pham, C. Nechifor, D. Puschmann, and J. Fernandes. "CityPulse: Large Scale Data Analytics Framework for Smart Cities". In: *IEEE Access* 4 (2016), pages 1086–1108. DOI: 10.1109/ACCESS.2016.2541999.

[4] S. Schmid, I. Gerostathopoulos, C. Prehofer, and T. Bures. "Self-Adaptation Based on Big Data Analytics: A Model Problem and Tool". In: *12th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems, SEAMS@ICSE 2017, Buenos Aires, Argentina, May 22-23, 2017*. IEEE Computer Society, 2017, pages 102–108. DOI: 10.1109/SEAMS.2017.20.

Exploring Game-Theoretic Formation of Realistic Networks

Tobias Friedrich, Pascal Lenzner, and Christopher Weyand

Algorithm Engineering Group
Hasso Plattner Institute for Digital Engineering
{firstname.lastname}@hpi.de

Many real world networks from different domains share the same structural properties. So far, there only exist models which reproduce these properties via a random process and thus have only limited explanatory power. In contrast to this, we have developed an agent-based game-theoretic model which promises a better explanation of the structure of real world networks. Our new model was investigated in computationally demanding large-scale experiments performed on the hardware of the HPI Future SOC Lab.

1 Introduction

Complex networks from the Internet to various (online) social networks have a huge impact on our lives and it is thus an important research challenge to understand these networks and the forces that shape them. The emergence of the Internet has kindled the interdisciplinary field of Network Science [3], which is devoted to analyzing and understanding real-world networks.

Extensive research, e.g. [1, 3, 4, 6, 13, 16, 17], on real world networks from many different domains like communication networks, social networks, protein-protein interaction networks, neural networks, etc. has revealed the astonishing fact that most of these networks share the following basic properties:

- *Small-world property:* The diameter and average distances in these networks are logarithmic in the number of nodes or even smaller.
- *Clustering:* Two nodes which are both adjacent to a third node have a high probability of being neighbors themselves, i.e. real networks contain an abundance of triangles and small cliques.
- *Power-law degree distribution:* In real networks the probability that a node has degree k is proportional to $k^{-\beta}$, for some constant $2 \leq \beta \leq 5$. That is, the degree distribution follows a power-law. Such networks are called *scale-free networks*.

The phenomenon that real world networks from different domains are very similar calls for a scientific explanation, i.e. formal models which generate networks with the above properties from very simple rules.

Many such models have been proposed, most prominently the preferential attachment model [4], the Chung-Lu random graph model [8], hyperbolic random graphs [12, 14] and geometric inhomogenous random graphs [5]. However, all these

models describe a purely random process which eventually outputs a network having realistic properties. On the one hand, this is desirable for sampling such networks, e.g. for testing algorithms on them, but on the other hand having a purely random process yields only a limited explanation of the structure of real world networks. Most real world networks evolved over time by the interaction of various rational agents. In case of the Internet the selfish agents correspond to the Internet Service Providers which control the Autonomous Systems, in case of social networks, the agents are people or companies who choose carefully with whom to connect. Thus, a model with higher explanatory power should consider rational selfish agents which use and modify the network to their advantage. Such models are at the core of the young field of Algorithmic Game Theory [18, 19].

2 Networks via Game Theory

In game-theoretic models for network formation selfish agents are associated to nodes of a network. Each agent chooses as strategy any subset of other agents to form a link to. The union of all links which are chosen by some player then determines the links of the created network.

The individual goal of each agent is modeled via a cost function, which typically consists of costs for creating links and of a service cost term, which measures the perceived quality of the created network for the individual agent, e.g. the service cost could be the sum of distances to all other agents [11] or just the number of reachable agents [2].

Any assignment of strategies to agents is considered an outcome of the game. Among all those outcomes the so-called equilibria are particularly interesting. In an equilibrium no agent wants to change her current strategy, given that all other players' strategies are fixed, i.e. no agent can reduce her costs in the current situation by forming another set of links. Analyzing the structure of such equilibrium networks then ideally yields insights into why real world networks exhibit the mentioned properties.

So far, such game-theoretic approaches can explain the small-world property, that is, it was proven that the diameter of all equilibrium networks is small [10]. However, to the best of our knowledge, no known game-theoretic model can explain the emergence of clustering and a power-law degree distribution. Thus, it is still an open problem to find and validate such a model.

3 Aims of the Project

Building on our previous work [7, 9, 15], we have developed a new game-theoretic model, called *strategic network augmentation*, which promises to solve the open problem. Initial experiments performed on the Future SOC Lab compute cluster [20, 21] revealed that the obtained equilibrium networks from our new model have the

small-world property, show significant clustering and the node degree distribution seems to be governed by a power-law.

The aim of this project was to explore another model variant which is based on a modified cost model for establishing edges. Our new model is inspired by social networks where it is easier/cheaper to establish a connection towards a friend of a friend than to connect to a node which is far away. The goal here was to establish if this variant also yields realistic node degree distributions and a non-negligible average local clustering coefficient.

4 Used Future SOC Lab Resources

The experiments were run on the high-performance cluster of the HPI Future SOC Lab. The cluster consisted of 22 nodes with 80 cores each and 1 TB of memory each.The experiments were run via the slurm job scheduler on all nodes in parallel.

5 Findings

A sample network generated with our new model variant is shown in figure 1.

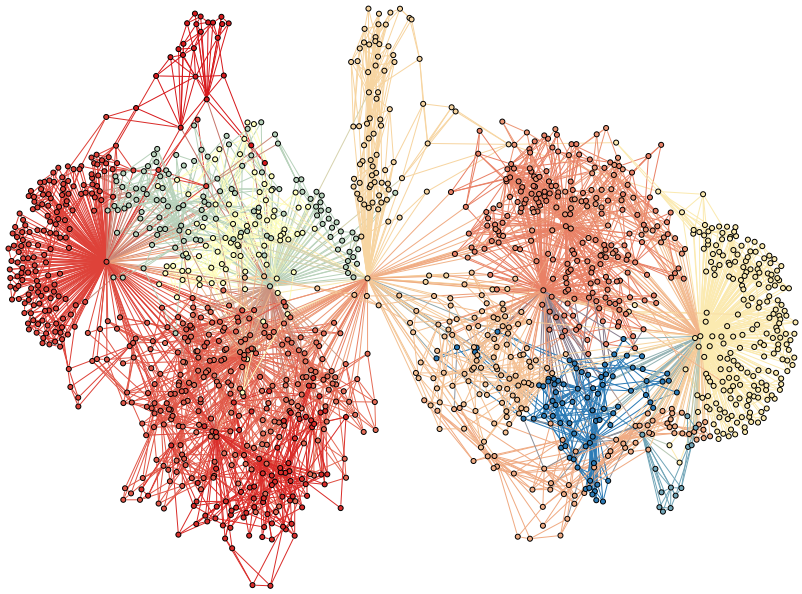


Figure 1: A sample network generated with our new model variant that is inspired by social networks

It illustrates that also our new variant of the model yields promising results. Compared to the previously studied model, we get a slightly larger diameter and also

the average local clustering coefficient is higher. Regarding the node degree distribution our new model yields good results but not as good as the old model. We get slightly too many high degree nodes and in several experimental runs the obtained maximum node degree was clearly too large—in some extreme cases we even got an induced spanning star.

We studied different parameter settings of our new model and we have uncovered settings which yield good results. However, compared to the old model, the new variant is much more sensitive to specific parameter settings and hence adujsting the parameters is much more tedious and error-prone.

6 Next Steps

A natural next step is to consider a combination of both so far studied models. Maybe it is possible to combine the models in such a way that we get the promising properties of both models.

The main problem of both current models is that the diameter in the obtained networks is rather low. While a diameter of $\Theta(\log n)$ or $\Theta(\log n / \log\log n)$ would be ideal, the experiments suggest that we get a constant diameter instead. We are currently working on an analytical proof that also our new model variant will only yield networks with a constant diameter (although larger than the diameter of the old model). This calls for further modifications of the models.

Last but not least, we want to validate our models with data from real networks. That is, we want to measure if real networks are close to being in equilibrium in our setting and we want to use time series of real networks to investigate if our models predict the right structural changes over time.

References

[1] R. Albert, H. Jeong, and A.-L. Barabási. "Internet: Diameter of the world-wide web". In: *nature* 401.6749 (1999), page 130.

[2] V. Bala and S. Goyal. "A noncooperative model of network formation". In: *Econometrica* 68.5 (2000), pages 1181–1229. DOI: 10.1007/978-3-540-24790-6_7.

[3] A.-L. Barabási. *Network science*. Cambridge University Press, 2016. DOI: 10.1098/rsta.2012.0375.

[4] A.-L. Barabási and R. Albert. "Emergence of Scaling in Random Networks". In: *Science* 286.5439 (1999), pages 509–512. DOI: 10.1515/9781400841356.349.

[5] K. Bringmann, R. Keusch, and J. Lengler. "Geometric inhomogeneous random graphs". In: *Theoretical Computer Science* 760 (2015), pages 35–54. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2018.08.014. arXiv: 1511.00576 [cs.SI].

[6] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. "Graph structure in the Web". In: *Computer Networks* 33.1 (2000), pages 309–320. DOI: 10.1016/S1389-1286(00)00083-9.

[7] A. Chauhan, P. Lenzner, A. Melnichenko, and L. Molitor. "Selfish Network Creation with Non-Uniform Edge Cost". In: *SAGT'17*. Springer. 2017, pages 160–172. DOI: 10.1007/978-3-319-66700-3_13.

[8] F. Chung and L. Lu. "The average distances in random graphs with given expected degrees". In: *PNAS* 99.25 (2002), pages 15879–15882. DOI: 10.1073/pnas.252631999.

[9] A. Cord-Landwehr and P. Lenzner. "Network Creation Games: Think Global - Act Local". In: *MFCS'15*. Edited by G. F. Italiano, G. Pighizzini, and D. Sannella. Volume 9235. Lecture Notes in Computer Science. Springer, 2015, pages 248–260. ISBN: 978-3-662-48053-3. DOI: 10.1007/978-3-662-48054-0.

[10] E. D. Demaine, M. T. Hajiaghayi, H. Mahini, and M. Zadimoghaddam. "The Price of Anarchy in Network Creation Games". In: *ACM Transactions on Algorithms* 8.2 (2012), page 13. DOI: 10.1145/1281100.1281142.

[11] A. Fabrikant, A. Luthra, E. Maneva, C. H. Papadimitriou, and S. Shenker. "On a Network Creation Game". In: PODC'03. Boston, Massachusetts: ACM, 2003, pages 347–351. DOI: 10.1145/872035.872088.

[12] T. Friedrich and A. Krohmer. "On the diameter of hyperbolic random graphs". In: *ICALP'15*. Springer. 2015, pages 614–625. DOI: 10.1007/978-3-662-47666-6_49.

[13] J. Kleinberg. "The Small-world Phenomenon: An Algorithmic Perspective". In: *STOC'00*. STOC '00. Portland, Oregon, USA: ACM, 2000, pages 163–170. ISBN: 1-58113-184-4. DOI: 10.1145/335305.335325.

[14] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguñá. "Hyperbolic geometry of complex networks". In: *Physical Review E* 82 (3 Sept. 2010), page 036106. DOI: 10.1103/PhysRevE.82.036106.

[15] P. Lenzner. "Greedy Selfish Network Creation". In: *WINE'12*. Edited by P. W. Goldberg. Volume 7695. Lecture Notes in Computer Science. Springer, 2012, pages 142–155. ISBN: 978-3-642-35310-9. DOI: 10.1007/978-3-642-35311-6_11.

[16] J. Leskovec, J. Kleinberg, and C. Faloutsos. "Graphs over time: densification laws, shrinking diameters and possible explanations". In: *SIGKDD'05*. ACM. 2005, pages 177–187. DOI: 10.1145/1081870.1081893.

[17] M. Newman, A.-L. Barabasi, and D. J. Watts. *The structure and dynamics of networks*. Princeton University Press, 2011. DOI: 10.1515/9781400841356.

[18] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. New York, NY, USA: Cambridge University Press, 2007. DOI: 10.1017/CBO9780511800481.

[19] C. H. Papadimitriou. "Algorithms, games, and the internet". In: *STOC'01*. Edited by J. S. Vitter, P. G. Spirakis, and M. Yannakakis. ACM, 2001, pages 749–753. ISBN: 1-58113-349-9. DOI: 10.1145/380752.380883.

[20] D. N. Schumann. "Exploring Game Theoretic Models for Generating Real World Networks". Master's thesis. Hasso Plattner Institute, University of Potsdam, 2018.

[21] C. Weyand. "Locality in Game Theoretic Models for Real-World Networks". Master's thesis. Hasso Plattner Institute, University of Potsdam, 2018.

Designing practical algorithms through overfitting

Markus Wagner

Optimisation and Logistics Group
University of Adelaide, Australia
markus.wagner@adelaide.edu.au

Since 2017, the teams around Prof. Tobias Friedrich (HPI, Chair for Algorithm Engineering) and Dr Markus Wagner (University of Adelaide, Australia) have explored the concept of automated algorithm configuration to design new search operators. This project builds upon the existing work and will push it toward real-world, interconnected problems.

While the HPI Future SOC Lab infrastructure has already allowed us to perform a great number of experiments, we have barely been able to scratch the surface. Additional experiments will be needed to further explore the field of Data-Driven Search-Based Software Engineering (DSE) that we have founded. DSE requires to be driven by potent hardware. DSE is in its infancy, and it will become increasingly important—see the following claim from our inaugural DSE paper:

> Claim4: Data mining without optimisation should be deprecated. Conclusions reached from an unoptimised data miner can be refuted, just by running the same tuned learner on the same data [...]. Since they can be so easily refuted, this community should stop publishing analytics papers that lack an optimisation component. [6]

1 Introduction

Data-Driven Search-Based Software Engineering (DSE) [6] and the recent generalisation Data Mining Algorithms Using/Used-by optimisers (DUO) [1] combine insights from Mining Software Repositories (MSR) and Search-based Software Engineering (SBSE). While MSR formulates software engineering problems as data mining problems, SBSE reformulates SE problems as optimisation problems and use meta-heuristic algorithms to solve them. Both MSR and SBSE share the common goal of providing insights to improve software engineering. The algorithms used in these two areas also have intrinsic relationships. Combining these two fields is useful for situations (a) which require learning from a large data source or (b) when optimisers need to know the lay of the land to find better solutions, faster.

Contributions to date

Prof. Tobias Friedrich and I have employed the DSE concept in 2017 to investigate heuristic search approaches to extensively explore the design space of heuristics

for certain optimisation problems. We distilled from the results new designs which our team was able to analyse theoretically [4, 5], proving that the new approaches performed asymptotically better than state-of-the-art approaches.

So far, we have only investigated relatively simple single-objective problems, which can be analysed theoretically, such as linear functions and some submodular functions. What is missing is the connection to single- and multi-objective real-world problems. This is what this project aims to do. We will use automated algorithm configuration to optimise heuristics for classes of problems and to families of problem instances. We are hoping to achieve breakthroughs similar to the ones achieved to date. Recent results include our work on topic modelling and on relating good parameter configurations with properties of the respective text corpora [8].

2 Problems with Interconnected Components

In optimisation research, problems with different characteristics are investigated. To find an appropriate algorithm for a practical problem often assumptions about characteristics are made, and then a suitable algorithm is chosen or designed. For instance, an optimisation problem can have several components interacting with each other. Because of their interaction they form an interwoven system where interdependencies in the design and the objective space exist. An optimal solution for each component independently will in general not be a good solution for the interwoven optimisation problem.

In order to provide an academic interwoven optimisation test problem, the Traveling Thief Problem (TTP) was proposed in 2013 [7] where two well-known subproblems, the Traveling Salesman Problem (TSP) and the Knapsack Problem (KP), interact with each other. As in the TSP problem a so-called thief has to visit each city exactly once. In addition to just travelling, the thief can make profit during his tour by stealing items and putting them in the rented knapsack. However, the thief's traveling speed decreases depending on the current knapsack weight, which then increases the rent that the thief has to pay for the knapsack. The TTP's canonical formulation is a single-objective one, however, because the travelling time and profit represent solutions with different trade-offs, the problem is bi-objective in nature. To date, pretty much no research has considered this.

3 Solution

Our method is based on a genetic algorithm with customisation addressing problem characteristics, because we address the BI-TTP, a bi-objective version of the TTP, where the goal is to minimise the overall travelling time and to maximise the profit of the collected items. We incorporate domain knowledge through a combination of near-optimal solutions of each subproblem in the initial population and a custom repair operation to avoid the evaluation of infeasible solutions. Moreover, the

independent variables of the TSP and KP components are unified to a real variable representation by using a biased random-key approach. The bi-objective aspect of the problem is addressed through an elite population extracted based on the non-dominated rank and crowding distance of each solution. We provide a comprehensive study which shows the influence of parameters on the performance of our method and investigate the performance of each parameter combination over time.

Non-Dominated Sorting Biased Random-Key Genetic Algorithm (NDS-BRKGA)

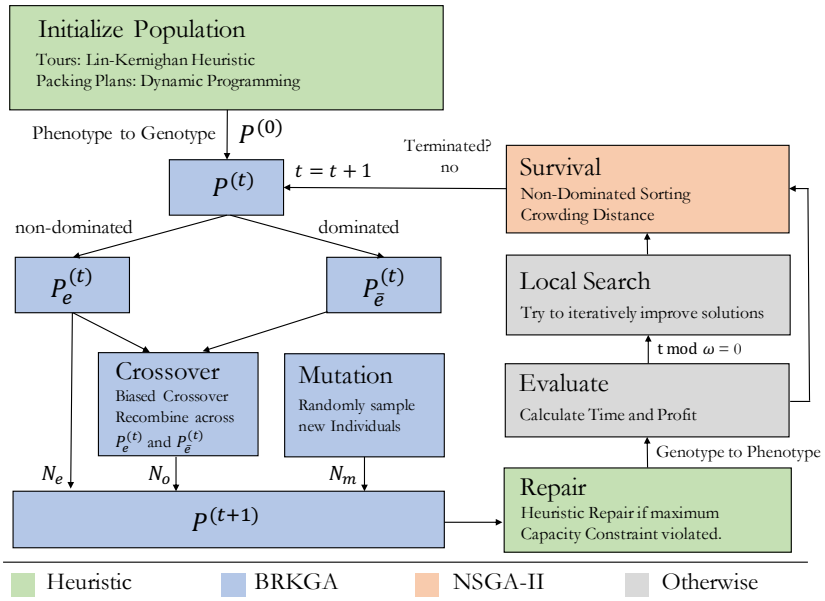


Figure 1: NDS-BRKGA: A customised genetic algorithm

Figure 1 illustrates the overall procedure of our highly-configurable NDS-BRKGA. At first, we generate the initial population using efficient solvers for the subproblems independently. Afterward, we combine the optimal or near-optimal solutions for both subproblems and convert them to their genotype representation which results in the initial population. For the purpose of mating, the population is split into an elite population $P_e^{(t)}$ and non-elite population $P_{\bar{e}}^{(t)}$. The individuals for the next generations $P^{(t+1)}$ are a union of the elite population $P_e^{(t)}$ directly, the offspring of a biased crossover and mutant individuals. In case an individual violates the maximum capacity constraint, we execute a repair operation. Then, we convert each individual to its corresponding phenotype and evaluate it on the problem instance. In order to insert an explicit exploitation phase in our algorithm, we apply at some evolutionary cycles a local search procedure in some elite individuals. Finally, the survival selection is applied and if the termination criterion is not met, we increase the generation

counter t by one and continue with the next generation. In the following, we describe the purpose of each of the design decisions we have made and explain what role it plays during a run of the algorithm.

4 Implementation

We have used the so-called 1000-core cluster of the HPI Future SOC Lab for the exhaustive hyper-parameter Study. Running all 3072 combinations (see table 1) on 9 instances, and performing 10 independent runs of 5 hours each consumed over 160 CPU years in the final experiments.

The parameters were population size N, elite population size N_e, mutant population size N_m, elite allele inheritance probability p_e, fraction $\alpha$of the initial population created from TSP and KP solvers, and the frequency ω for local search.

5 Evaluation

In figure 2, we visualise the best parameter configurations at six different execution times. In each plot the best obtained parameter configuration regarding hypervolume is highlighted in red and parameter configurations up to 0.1 % worse than the best are highlighted in blue. Note that the importance of values of each parameter among the best parameter configurations is indicated by the intensity of the blue color once some parameter configurations share some parameter values. The following can be observed:

1. **More execution time, better results:** The number of parameter configurations that are capable of generating large hypervolume values increases as the execution time of our algorithm increases. This means that in some runs even though the parameters have not been set appropriately, the algorithm is still able to converge.

Table 1: Parameter values considered during the experiment. The values highlighted in red have been added since the last report, and their addition resulted in the use of an additional 100 CPU years of runtime.

Parameter	Values
N	100, 200, 500, 1000
N_e	$0.3N$, $0.4N$, $0.5N$, $0.6N$
N_m	$0.0N$, $0.1N$, $0.2N$
ρ_e	0.5, 0.6, 0.7, 0.8
α	0.0, 0.1, 0.2 , 0.3
ω	1, 10, 50, 100

2. **Importance of TSP and KP solvers:** It has influence on the overall performance of the algorithm if TSP and KP solvers are used for initialization which is determined by α. The best results are obtained if at least 10 % of the initial solutions are biased towards those solutions found a TSP and KP solvers.

3. **Trends when execution time increases**: We can see a trend as the execution time increases. Our method performs better with a large population, a large survival rate, a small or no explicit diversification through mutant individuals, a small influence of single-parent inheritance, an minor influence of a good initial population, and significant influence of local search procedure.

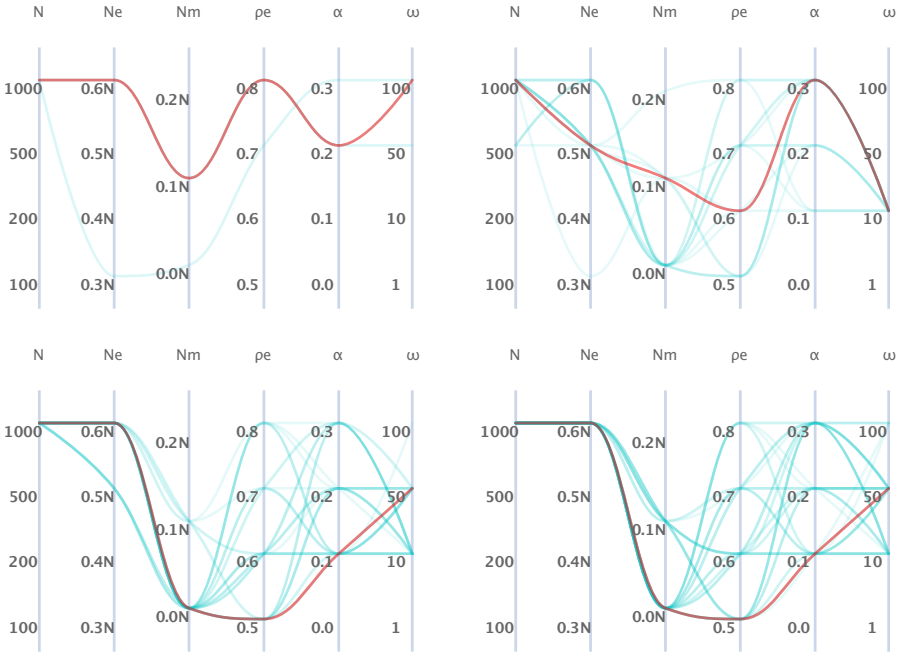


Figure 2: Best parameter configurations over all instances with varying execution times. From top left to bottom right: 600/1800/7200/18000 seconds.

Comparison with single-objective TTP solutions

As the BI-TTP is related to the original TTP (which has been subject to many studies), we now build the bridge to it. Therefore, we compare our results with the best known single-objective TTP objective scores, which come from a comprehensive comparisons of a variety of algorithms reported in [9]. The computational budgets of the approaches which have obtained the best known solutions might vary.

Table 2 compares for each instance the best known score of the TTP with the best score found by our algorithm when it optimised the BI-TTP. Note that despite the

strong connection of the BI-TTP to the single-objective TTP, maximising the single-objective TTP objective score is not an explicit goal of the BI-TTP. Nevertheless, NDS-BRKGA has found better scores for the three smallest instances with 280 cities with up to 2790 items.

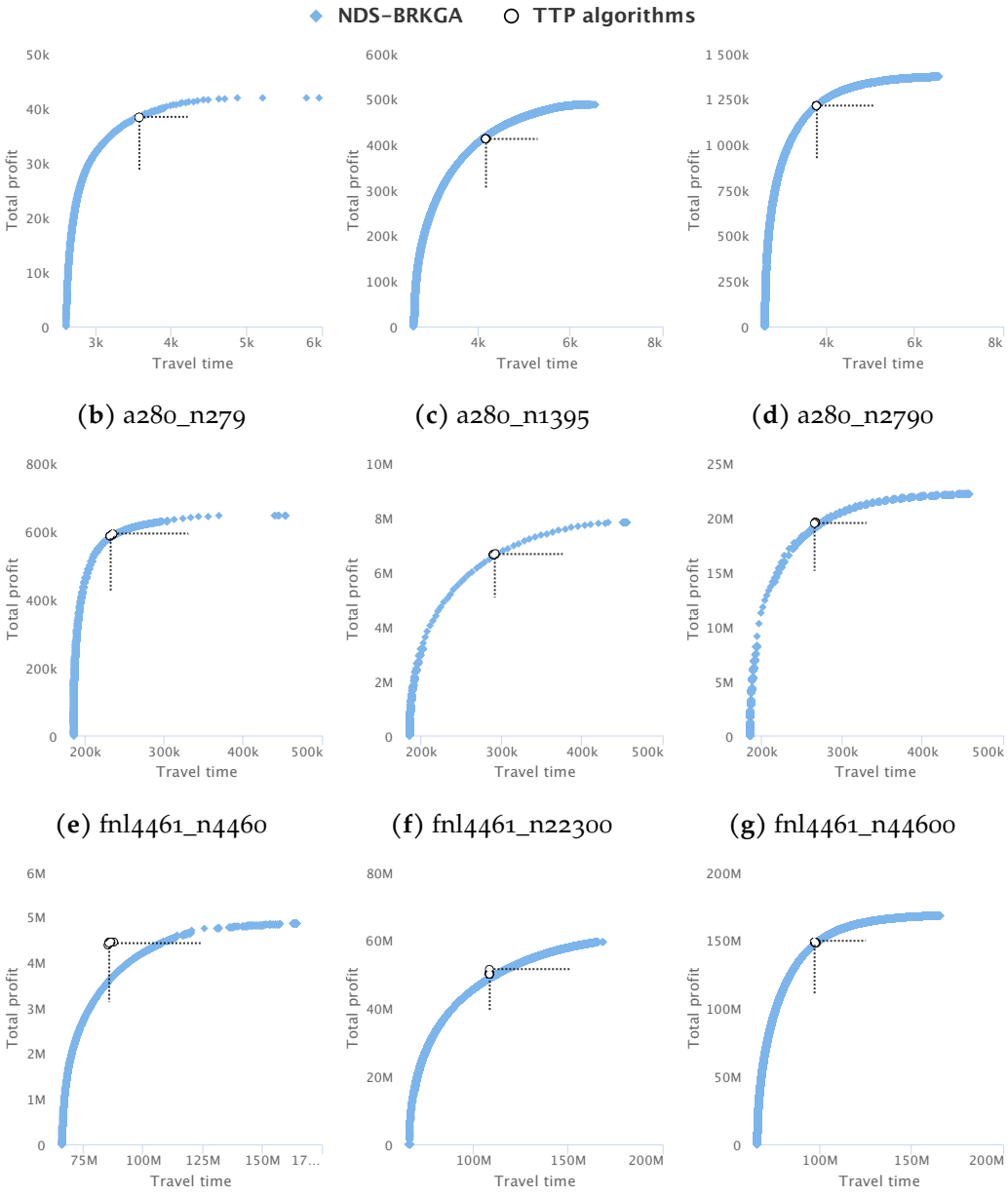
Table 2: Single-objective comparison of the TTP objectives scores

Instance	**TTP score***	**NDS-BRKGA**
a280_n279	18 470.000[a]	18 603.120
a280_n1395	110 147.219[b]	115 445.521
a280_n2790	429 081.783[c]	429 085.353
fnl4461_n4460	263 040.254[d]	257 394.821
fnl4461_n22300	1 705 326.000[a]	1 567 933.421
fnl4461_n44600	6 744 903.000[a]	6 272 240.702
pla33810_n33809	1 863 667.592[d]	1 230 174.003
pla33810_n169045	15 634 853.130[e]	12 935 090.876
pla33810_n338090	58 236 645.120[f]	55 688 288.508

* Available at https://cs.adelaide.edu.au/~optlog/research/combinatorial.php.
[a] Obtained from CS2SA approach proposed by [2] [b] obtained from C3 approach [c] obtained from C4 approach [d] obtained from S5 approach [e] obtained from S1 approach [f] obtained from C6 approach. All these last approaches have been proposed by [3].

In figure 3, we plot the 100 % attainment surface for each instance showing the values of the two objectives of all non-dominated solutions found by our algorithm,[1] as well as for the TTP solutions found by running 10 times the best algorithm as identified in [9] for each instance. In order to better analyse the results, we delimit the region dominated by the TTP solutions by dotted lines. For each instance, the TTP solutions found have presented similar quality concerning their scores, as we can see by the superimposition of the TTP solutions in the plots. Note that in almost all instances, few or none of our solutions have been dominated. The largest number of dominated solutions has been obtained for the instance *pla33810_n33809*, in which our algorithm has had the worst performance concerning the other solutions approaches.

[1] Available at https://github.com/jonatasbcchagas/nds-brkga_bi-ttp.



(**b**) a280_n279 (**c**) a280_n1395 (**d**) a280_n2790

(**e**) fnl4461_n4460 (**f**) fnl4461_n22300 (**g**) fnl4461_n44600

(**h**) pla33810_n33809 (**i**) pla33810_n169045 (**j**) pla33810_n338090

Figure 3: NDS-BRKGA solutions and the best known TTP solutions

6 Future Work

While the infrastructure allowed us to perform a great number of experiments, we have barely been able to scratch the surface. Additional experiments will be needed to further explore the data-driven design of our custom approach to this problem.

Among the next steps is the write-up of the preliminary findings in a scientific article—which will be submitted following HPI's review of the draft. We are currently finalising the article for submission to the Journal of Heuristics.

We plan to apply for an extension of the current project, as DSE requires to be driven by potent hardware. DSE is in its infancy, and it will become increasingly important—see the following claim from our inaugural DSE paper:

> Claim4: Data mining without optimisation should be deprecated. Conclusions reached from an unoptimised data miner can be refuted, just by running the same tuned learner on the same data [...]. Since they can be so easily refuted, this community should stop publishing analytics papers that lack an optimisation component. [6]

References

[1] A. Agrawal, T. Menzies, L. L. Minku, M. Wagner, and Z. Yu. *Better Software Analytics via "DUO": Data Mining Algorithms Using/Used-by Optimizers*. 2018. DOI: 10.1007/s10664-020-09808-9. arXiv: 1812.01550 [cs.SE].

[2] M. El Yafrani and B. Ahiod. "Population-based vs. single-solution heuristics for the travelling thief problem". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM. 2016, pages 317–324. DOI: 10.1145/2908812.2908847.

[3] H. Faulkner, S. Polyakovskiy, T. Schultz, and M. Wagner. "Approximate approaches to the traveling thief problem". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM. 2015, pages 385–392. DOI: 10.1145/2739480.2754716.

[4] T. Friedrich, A. Göbel, F. Quinzan, and M. Wagner. "Heavy-Tailed Mutation Operators in Single-Objective Combinatorial Optimization". In: *Parallel Problem Solving from Nature – PPSN XV*. Edited by A. Auger, C. M. Fonseca, N. Lourenço, P. Machado, L. Paquete, and D. Whitley. Springer, 2018, pages 134–145. DOI: 10.1007/978-3-319-99253-2_11.

[5] T. Friedrich, F. Quinzan, and M. Wagner. "Escaping Large Deceptive Basins of Attraction with Heavy-tailed Mutation Operators". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO '18. Kyoto, Japan: ACM, 2018, pages 293–300. DOI: 10.1145/3205455.3205515.

[6] V. Nair, A. Agrawal, J. Chen, W. Fu, G. Mathew, T. Menzies, L. Minku, M. Wagner, and Z. Yu. "Data-driven Search-based Software Engineering". In: *Proceedings of the 15th International Conference on Mining Software Repositories*. MSR '18. Gothenburg, Sweden: ACM, 2018, pages 341–352. DOI: 10.1145/3196398.3196442.

[7] S. Polyakovskiy, M. R. Bonyadi, M. Wagner, Z. Michalewicz, and F. Neumann. "A Comprehensive Benchmark Set and Heuristics for the Traveling Thief Problem". In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. GECCO '14. Vancouver, BC, Canada: ACM, 2014, pages 477–484. DOI: 10.1145/2576768.2598249.

[8] C. Treude and M. Wagner. "Predicting Good Configurations for GitHub and Stack Overflow Topic Models". In: *Proceedings of the 16th International Conference on Mining Software Repositories*. MSR '19. Montreal, Quebec, Canada: IEEE Press, 2019, pages 84–95. DOI: 10.1109/MSR.2019.00022.

[9] M. Wagner, M. Lindauer, M. Mısır, S. Nallaperuma, and F. Hutter. "A case study of algorithm selection for the traveling thief problem". In: *Journal of Heuristics* 24.3 (June 2018), pages 295–320. ISSN: 1572-9397. DOI: 10.1007/s10732-017-9328-y.

Integrating Hardware Accelerators in Virtualized Environments

Data Transfer Throughput in Scale-Out GPU Scenarios Using On-the-Fly I/O Link Compression

Max Plauth and Andreas Polze

Operating Systems and Middleware Group
Hasso Plattner Institute for Digital Engineering
{firstname.lastname}@hpi.de

The overhead of moving data is the major limiting factor in todays hardware, especially in heterogeneous systems where data needs to be transferred back and forth frequently between host and GPU memory. In the Fall 2019 period, this project performed a first end-to-end evaluation, investigating the effective transfer throughput that can be achieved using 842-based *On-the-Fly I/O Link Compression* for Scale-Out GPU Scenarios.

1 Introduction

In the age of ever-increasing data volumes, the overhead of data transfers is a major inhibitor of further performance improvements on many levels. In heterogeneous compute architectures, the overhead of transferring data (e.g. between host and GPU memory) is already a major limiting factor on the intra-node level. The increased latencies and limited bandwidths available in most scale-out scenarios aggravate the situation.

Contributions

Preceding efforts of the research community have identified compression as a viable method for improving data transfer efficiency for certain application domains [4, 16]. To work around the issue of insufficient compression throughput, many of these investigations have proposed the use of *Offline I/O Link Compression*, where the payload for data transfers is available in a pre-compressed form. More recently, however, hardware-accelerated compression techniques are becoming available in an increasing number of computer architectures [2, 6]. In this report, we demonstrate that *On-the-Fly I/O Link Compression* is a feasible method for improving data transfer efficiency for scale-out GPU workloads.

2 Background: 842 Compression

The 842 compression algorithm has been introduced by IBM and has been implemented in hardware in the *nx842* on-chip compression accelerator available in their POWER processors. The main design goal of the 842 algorithm [3] is to allow high-throughput/low-latency hardware implementations that can be placed directly on transmission channels [17]. Hereinafter, the basic procedure of the 842 algorithm is outlined.

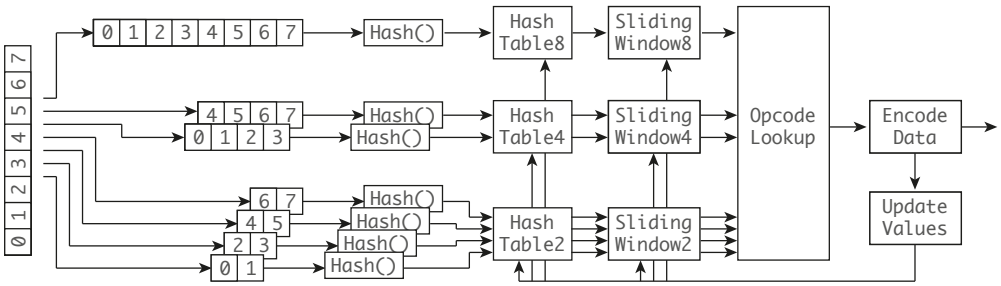


Figure 1: The 842 compression algorithm operates on units of 8 bytes, treating the input data as sub-phrases of 8, 4 and 2 bytes length. The algorithm relies on efficient hashing and sliding window buffers containing past compressed data.

As illustrated in figure 1, the 842 algorithm [3] implemented by the *nx842* on-chip accelerator operates on units of 8 bytes, treating the input data as sub-phrases of 8, 4 and 2 bytes length, respectively. For each phrase length, a hash function and a hash table with offsets to a sliding window buffer of past encoded data are used to detect possible matches of the sub-phrase therein. Based on the lookup, a template is chosen that encodes 8 bytes of raw data. Each 5-bit template encodes a permutation of offsets or literals of 8, 4 and 2 bytes length, followed the actual offsets and literals. With a clock frequency of $2.3GHz$, and the ability to ingest 8 bytes per cycle, one *nx842* on-chip accelerator can achieve a maximum throughput of 18 GB/sec [6]. Being equipped with two *nx842* on-chip accelerators [1], the total compression throughput of a POWER8 processor can be as high as 36 GB/sec.

The 842 algorithm can be attributed to the family of Lempel-Ziv derivatives [3]. The compression process deviates from the original Lempel-Ziv algorithm [18] in several aspects. However, decompression works almost identical compared to LZ'77 [3]. With a sufficient number of approaches available that have demonstrated efficient decompression of Lempel-Ziv derivatives on Graphics Processing Units (GPUs) [5, 15, 16, 19], we assume that decompression of 842-encoded data can be implemented efficiently on GPUs. Regarding Field-Programmable Gate Arrays (FPGAs), an efficient implementation of 842 for both compression and decompression has been demonstrated as well [17], providing further indication that efficient implementations of the algorithm are feasible for most popular accelerator types.

3 Experimental Setup

In order to hide the compression facilities behind the regular OpenCL API, we integrated them into the *dOpenCL* library [9]. We employ the compression (CPU) and decompression (GPU) facilities brought forward in preceding Future SOC Lab Periods of this project [10]. The effective transfer throughput with and without *On-the-Fly I/O Link Compression* has been measured between the master and compute nodes specified in table 1. This test has been performed using a modified version[1] of the `oclBandwidthTest` sample application from the NVIDIA OpenCL SDK [14]. It uses several synthetic patterns as compression payload, including a periodic pattern of 256 bytes (`00 01 02 03 …FD FE FF 00 01`, compression ratio $r = 0.21$), a series of zeros ($r = 0.0036$), as well as pseudo-random data ($r = 1$). Those artificial payloads are intended to test effective data transfer bandwidth for worst case and best case edge cases. To include more representative payloads, we have included tests that are transfering the first gigabyte of the *Large Text Compression Benchmark* [11] (enwik9, $r = 0.71$), the *Open Library Works Dump* [8] (OLW, $r = 0.54$), as well as the *Curiosity's 1.8-Billion-Pixel Panorama* [12] (Space, $r = 0.51$), respectively.

To evaluate effective transfer throughput in a scale-out scenario, we have implemented a modified version[2] of the previous benchmark that performs data transfers to eight nodes, simultaneously. We created eight Docker containers with one GPU attached each to emulate a scale-out scenario with eight compute nodes. This test uses the same data sets and only increases the data volume in proportion to the larger number of nodes. The effective transfer throughput is aggregated across all nodes.

Table 1: Specifications of the employed test systems

	Master Node	Compute Node
Model	HPE ProLiant DL560 Gen10 [7]	NVIDIA DGX-1 [13]
CPU	2×Intel Xeon Gold 6148, 3.4 GHz, 20C/40T each	2×Intel Xeon E5-2698v4 2.20 GHz, 10C/20T each
Memory	1.5TB DDR4 ECC, 2666 MHz	512GB DDR4 ECC, 2133 MHz
GPU	n/a	8×NVIDIA Tesla V100
NIC	10 Gbps Ethernet	10 Gbps Ethernet
OS	Ubuntu 18.04.4	Ubuntu 18.04.4
Kernel	4.15.0	4.15.0

[1] https://github.com/joanbm/nvidia-opencl-sdk-samples-mod (last accessed 2020-01-01).
[2] https://github.com/joanbm/dopencl/blob/1dcb95aee69b/standalone_test/src/read_write_compressed.cpp (last accessed 2020-01-01).

4 Results

Our measurements for the effective single-node transfer throughput and the effective scale-out transfer throughput are presented in panels (a) and (b) of figure 2, respectively. The transfer throughput tests demonstrate that using real-world payloads, *On-the-Fly I/O Link Compression* can indeed improve effective transfer throughput between 1.20× and 1.83× in the single node scenario, and between 1.45× and 2.11× in the scale-out scenario. Random data as the worst-case payload has no significant negative impact on the throughput, whereas benevolent payloads such as the periodic pattern or zeroed data can yield drastic performance improvements. A closer look at the scale-out results for real-world payloads reveals that the limited bandwidth, especially on the master node network interface, remains as a major bottleneck, as the aggregated effective bandwidth of the scale-out test is only slightly above the level of the single node test.

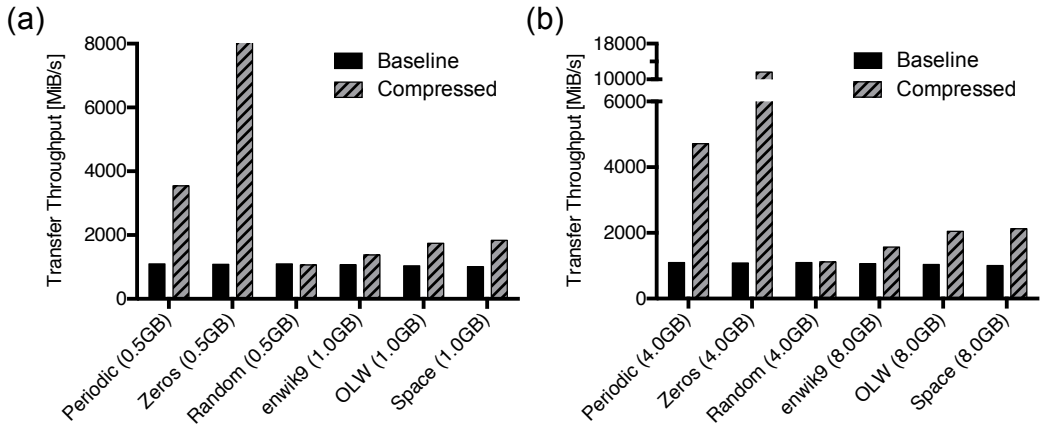


Figure 2: The effective transfer throughput with and without *On-the-Fly I/O Link Compression* are documented in panels (a) and (b) for a single node and a scale-out configuration, respectively.

5 Outlook

To study the feasibility of *On-the-Fly I/O Link Compression*, the next logical step is to perform additional experiments using benchmarks from the heterogeneous computing domain. As soon as the hardware-accelerated *nx842* engines become accessible from user-space software, the compression throughput should be high enough to extend the same beneficial performance effects on scale-out workloads executed accross higher performance networks (e.g. 40 Gbps and possibly even more).

References

[1] S. Chabrolles, J. Limodin, and F. Moyen. *Enabling POWER 8 Advanced Features on Linux*. 2016.

[2] *Datapath Acceleration Architecture 2*. NXP Semiconductors.

[3] P. A. Franaszek, L. A. Lastras-Montaño, S. Peng, and J. T. Robinson. "Data Compression with Restricted Parsings". In: *Data Compression Conference (DCC'06)*. Mar. 2006, pages 203–212. DOI: 10.1109/DCC.2006.22.

[4] S. Funasaka, K. Nakano, and Y. Ito. "Adaptive Loss-Less Data Compression Method Optimized for GPU Decompression". In: *Concurrency and Computation: Practice and Experience* 29.24 (Dec. 2017), e4283. DOI: 10.1002/cpe.4283.

[5] S. Funasaka, K. Nakano, and Y. Ito. "Fully Parallelized LZW Decompression for CUDA-Enabled GPUs". In: *IEICE Transactions on Information and Systems* 99.12 (2016), pages 2986–2994.

[6] H. Hellmuth and J. Klauke. *POWER NX842 Compression for Db2*. White Paper. IBM Corporation, Sept. 2017.

[7] Hewlett Packard Enterprise. *HPE ProLiant DL560 Gen10 Server QuickSpecs*. (Website). 2020. URL: https://www.hpe.com/h20195/v2/GetDocument.aspx?docname=a00021850enw (last accessed 2020-04-08).

[8] Internet Archive. *Open Library Works Dump*. (Website). 2020. URL: https://openlibrary.org/developers/dumps (last accessed 2020-04-08).

[9] P. Kegel, M. Steuwer, and S. Gorlatch. "dOpenCL: Towards a Uniform Programming Approach for Distributed Heterogeneous Multi-/Many-Core Systems". In: *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*. IPDPSW '12. Washington, DC, USA: IEEE Computer Society, 2012, pages 174–186. ISBN: 978-0-7695-4676-6. DOI: 10.1109/IPDPSW.2012.16.

[10] *Lib842 Repository*. (Website). URL: https://github.com/plauth/lib842 (last accessed 2020-04-08).

[11] M. Mahoney. *Large Text Compression Benchmark*. URL: http://mattmahoney.net/dc/textdata.html (last accessed 2020-04-08).

[12] NASA. *Curiosity's 1.8-Billion-Pixel Panorama*. (Website). 2020. URL: https://mars.nasa.gov/resources/24801/curiositys-18-billion-pixel-panorama/ (last accessed 2020-04-08).

[13] NVIDIA. *NVIDIA DGX-1 Datasheet*. (Website). 2019. URL: https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-1/dgx-1-rhel-datasheet-nvidia-us-808336-r3-web.pdf (last accessed 2020-04-08).

[14] *NVIDIA OpenCL Examples*. (Website). 2012. URL: https://github.com/sschaetz/nvidia-opencl-examples (last accessed 2020-04-08).

[15] A. Ozsoy and M. Swany. "CULZSS: LZSS Lossless Data Compression on CUDA". In: *2011 IEEE International Conference on Cluster Computing*. Sept. 2011, pages 403–411. DOI: 10.1109/CLUSTER.2011.52.

[16] E. Sitaridi, R. Mueller, T. Kaldewey, G. Lohman, and K. A. Ross. "Massively-Parallel Lossless Data Decompression". In: *2016 45th International Conference on Parallel Processing (ICPP)*. Aug. 2016, pages 242–247. DOI: 10.1109/ICPP.2016.35.

[17] S. Suresh and V. Udayashekar. "High-Throughput, Lossless Data Compression and Decompression On FPGAs". Master's Thesis. California State University, Northridge, May 2012, page 84.

[18] J. Ziv and A. Lempel. "A universal algorithm for sequential data compression". In: *IEEE Transactions on Information Theory* 23.3 (May 1977), pages 337–343. ISSN: 0018-9448. DOI: 10.1109/TIT.1977.1055714.

[19] Y. Zu and B. Hua. "GLZSS: LZSS Lossless Data Compression Can Be Faster". In: *Proceedings of Workshop on General Purpose Processing Using GPUs*. Salt Lake City, UT, USA: ACM, 2014, 46:46–46:53. ISBN: 978-1-4503-2766-4. DOI: 10.1145/2588768.2576785.

Benchmarking the Sort Algorithm on Ethernet Cluster

Benchmarking PCJ — a library for parallel computing in Java

Marek Nowicki

Department of Computer Science
Faculty of Mathematics and Computer Science
Nicolaus Copernicus University in Toruń, Poland
faramir@mat.umk.pl

The paper shows the evaluation of various implementations of the sort algorithm on the 1000 Core Cluster at HPI Future SOC Lab. The implementations was done using PGAS paradigm using the PCJ library — a Java library for parallel computing. This is the second report of usage the Java on the Ethernet Cluster.

1 Introduction

The project called *Benchmarking Java on Ethernet Cluster* checks the scalability of parallel, network-intensive microbenchmarks and applications written in Java, using the PCJ library, on the cluster with high-performance Ethernet — on the 1000 Core Cluster — Ethernet-based cluster — at the Hasso Plattner Institute. Current paper describe the findings done during the first extension of the project in the period from fall 2019 to spring 2020 associated with sort algorithm. A previous technical report paper [1] presents the performance of some selected microbenchmarks and one of the sort implementation.

The PCJ library [6] is HPC Challenge 2014 award-winning Java library for high-performance parallel computing. The PCJ library implements PGAS (Partitioned Global Address Space) paradigm for running concurrent applications on a multicore computer or computing clusters, that is, the systems consist of many multicore nodes. The PCJ library is an open source library (BSD license) [5]. More detailed information about the library can be found in the previous technical report [1] and other papers [2, 3].

The sort benchmark presented in this paper has assumption about input data. The data consists of many 100-byte records that is composed of 10-byte key and 90-byte value. The data is read from file by application, records are sorted comparing key values, and then saved to file(s). The execution time count all the steps needed, not only the sorting part.

The rest of the paper is organized as follows. Section 2 contains implementation description, then section 3 presents the benchmark results and the paper is concluded by section 4.

2 PCJ implementations

This paper evaluates five different PCJ implementations of a sort algorithm. Each PCJ implementation is a variant of *Sample Sort*. The execution is divided into 5 steps, similar to Edahiro's *Mapsort* described in [4]:

1. reading pivots,
2. reading input and placing records into proper buckets,
3. exchanging buckets data between threads,
4. sorting data stored in buckets,
5. writing output data.

Detailed description of each step is described in the following subsection.

2.1 Basic version implementation

The first and the basic PCJ implementation uses single file as an input file, and writes result to a single output file. Total number of records are derived from the input file size. Every thread read own portion of input file. The number of thread's records is roughly equal for all threads. If the total number of records divide with a remainder, the threads with id less than remainder has one record more to process. Listing 1 shows the presented calculation of portions of input file to read by each thread.

2.1.1 Reading pivots

Pivots are selected records from the input file. Pivots will be used for dividing input data into buckets. It is a variation of the selecting sampled records for key ranges of *sample sort*, as the whole set of pivots is used for splitting input. There is an option called *a number of pivots* that stands for the number of pivots per thread. The default

Listing 1: Calculating portion of file to read by each thread

```
1  TeraFileInput input = new TeraFileInput(inputFile));
2  long totalElements = input.length();
3
4  long localElementsCount = totalElements / PCJ.threadCount();
5  long reminderElements = totalElements % PCJ.threadCount();
6  if (PCJ.myId() < reminderElements) {
7     ++localElementsCount;
8  }
9
10 long startElement
11       = PCJ.myId() * (totalElements / PCJ.threadCount())
12           + Math.min(PCJ.myId(), reminderElements);
13 long endElement = startElement + localElementsCount;
```

value for it is 3, selected in arbitrary way, and the value will be used in almost all presented benchmarks.

Listing 2 presents the code for reading pivots. Pivots are read evenly from specific portion of input file by each thread (line 1-5) — each thread starts reading input file in different place. Then Thread-0 performs reduce operation for gathering pivots data from other thread (line 7). The possible duplicate records are removed from the list and the list is being sorted (line 12). At the end the list is broadcasted to all threads (line 15). A thread starts reading input file when it receive the list (line 17).

The upper bound of total number of pivots is limited by:

$$\#\text{totalPivots} \leq \#\text{pivots} \cdot \#\text{threads}.$$

If the input file does not contain duplicate records, the number of pivots will be exactly the value.

Listing 2: Reading pivots

```
1  for (int i = 0; i < numberOfPivotsByThread; ++i) {
2     Element pivot = input.readElement();
3     pivots.add(pivot);
4  }
5  PCJ.barrier();
6  if (PCJ.myId() == 0) {
7     pivots = PCJ.reduce((left, right) -> {
8        left.addAll(right);
9        return left;
10    }, Vars.pivots);
11
12    pivots = pivots.stream()
13                .distinct().sorted()
14                .collect(Collectors.toList());
15    PCJ.broadcast(pivots, Vars.pivots);
16 }
17 PCJ.waitFor(Vars.pivots);
```

2.1.2 Reading input

Pivots are the records that divide input data into buckets. The total number of buckets is calculated as:

$$\#\text{totalBuckets} = \#\text{totalPivots} + 1.$$

Each thread has to have its own array of buckets that will be used for exchanging data between threads. Each bucket is a list of records. While reading input, the record's bucket is deducted from its possible insert place in pivots list by using built-in binary search and then added to it. Listing 3 presents the way of creating buckets array (line 1) and reading input data (line 6).

Listing 3: Reading input

```
List<Element>[] localBuckets = new List[pivots.size() + 1];
for (int i = 0; i < localBuckets.length; ++i) {
   localBuckets[i] = new LinkedList<>();
}

input.seek(startElement);
for (long i = startElement; i < endElement; ++i) {
   Element e = input.readElement();
   int bucketNo = Collections.binarySearch(pivots, e);
   if (bucketNo < 0) {
      bucketNo = -(bucketNo + 1);
   }
   localBuckets[bucketNo].add(e);
}
```

2.1.3 Exchanging buckets

After reading input file, it is necessary to send data from the buckets to threads that are *responsible for* them. The responsibility here means sorting and writing to an output file. Similarly like division of input file to threads, the number of buckets that thread is responsible for, is roughly equal for all threads. Some threads can have one bucket more—it is when the number of buckets does not divide without remainder by thread count. The number of buckets that thread is responsible for is presented in the equation:

$$\#\text{buckets} = \frac{\#\text{totalBuckets} + \#\text{threads} - (\text{threadId} + 1)}{\#\text{threads}}.$$

Buckets array does not have information about the destination *threadId* that is responsible for each element and the index in destination array to place the element. It is necessary to have the mapping from array index to the destination *threadId* and to destination array index. The easiest way to do so requires looping over threads that could be not efficient, as the searching would be done in linear time complexity. The better way is to make calculations in constant time complexity. Let *pack* be the set of buckets of a thread. As stated before, threads with lower id could have one bucket more in their *packs*—let's call the bigger packs *bigPacks*, and *smallPacks* the rest. The size of each *smallPack* and *bigPack* is:

$$\begin{aligned}\text{smallPackSize} &= \frac{\#\text{totalBuckets}}{\#\text{threads}}\\ \text{bigPackSize} &= \text{smallPackSize} + 1.\end{aligned}$$

The number of *bigPacks* is:

$$\#\text{bigPacks} = \#\text{totalBuckets} \bmod \#\text{threads}.$$

Having the number of *bigPacks* and their size, it is easy to get the index, in which the *smallPacks* starts (*bigPackLimit*):

$$\text{bigPackLimit} = \text{bigPackSize} \cdot \#\text{bigPacks}.$$

Now, if current processing index i of source buckets array is lower than *bigPackLimit*, then the destination *threadId* and destination array index could be calculated as:

$$\text{destThreadId} = \frac{i}{\text{bigPackSize}}$$
$$\text{destIndex} = i \bmod \text{bigPackSize}.$$

For the rest i values, the equations are:

$$\text{destThreadId} = \#\text{bigPacks} + \frac{i - \text{bigPackLimit}}{\text{smallPackSize}}$$
$$\text{destIndex} = (i - \text{bigPackLimit}) \bmod \text{smallPackSize}.$$

The exchanging buckets between threads is presented on listing 4. For the further sorting step, the bucket data is converted from `List` to an array before sending. It is not necessary as the sorting of `List` dumps the content of the list into an array, but can save some bytes that have to be send through network. Additionally, instead of sending the array when destination thread is current thread, the data is just put to *communicable* variable without cloning the array.

Listing 4: Exchanging buckets

```
1 Element[] bucket = localBuckets[i].toArray(new Element[0]);
2
3 if (PCJ.myId() != destThreadId) {
4    PCJ.asyncPut(bucket, destThreadId,
5          Vars.buckets, destIndex, PCJ.myId());
6 } else {
7    PCJ.putLocal(bucket,
8          Vars.buckets, destIndex, PCJ.myId());
9 }
```

After sending buckets data to all other nodes, it is necessary to wait for receiving data from them. The easiest way, but not the most efficient one, to do so using the PCJ library is to wait for exact number of data receives using `PCJ.waitFor(-)` method like presented on listing 5. Because of the simplicity of the solution, current application use the easiest implementation. The exact number of receives is equal to:

$$\#\text{receives} = \#\text{threads} \cdot \#\text{buckets}.$$

Listing 5: Waiting for exact number of data receives

```
1 PCJ.waitFor(Vars.buckets, PCJ.threadCount() * buckets.length);
```

2.1.4 Sorting

After receiving all buckets data it is time to sort. However, each bucket is shredded into small arrays—one array per thread. The *buckets* variable is three dimensional array: `Element[][][]`. First dimension is for bucket number, second dimension is for thread number, and third dimension stores records. It is necessary to flatten buckets data—join all records from specified bucket skipping the second dimension. Listing 6 contains the code for flattering (line 3) and sorting flattened data (line 6).

Each bucket is processed one by one and stored into *sortedBuckets* array.

Listing 6: Sorting buckets

```
1 Element[][] sortedBuckets = new Element[buckets.length][];
2 for (int i = 0; i < buckets.length; i++) {
3    sortedBuckets[i] = Arrays.stream(buckets[i])
4                       .flatMap(Arrays::stream)
5                       .toArray(Element[]::new);
6    Arrays.sort(sortedBuckets[i]);
7 }
```

2.1.5 Writing output

Writing buckets data to single output file in a correct order is next step. This is the most sequential part of an application. Each thread has to wait for its turn to write data to output file.

Listing 7 demonstrates the writing step. The turn is determined by a *sequencer* variable state. At the beginning, the sequencer variable is touched on Thread-0. Each thread, except Thread-0, waits on `PCJ.waitFor(-)` method. When Thread-0 finishes writing data, it touches the *sequencer* variable on next thread. Then next thread writes data and touches the *sequencer* variable on consecutive thread so subsequent thread can write data, and so on.

Writing was the last step of the implementation.

2.2 Other implementations

In total five implementations were benchmarked:

MultiplePivots is the first implementation described in the previous subsection;

OnePivot is the second implementation, based on the first one, where there are exactly one pivot per thread and the pivots are selected evenly from larger set of records (100 000 records);

OnePivotMultipleFiles is the third implementation, based on the second one, where each thread write output to separate file without synchronizing writing;

Listing 7: Writing buckets

```
if (PCJ.myId() == 0) {
   PCJ.putLocal(true, Vars.sequencer);
}
                                    …
PCJ.waitFor(Vars.sequencer);
try (TeraFileOutput output = new TeraFileOutput(outputFile)) {
   Arrays.stream(sortedBuckets)
        .forEach(output::writeElements);
}
PCJ.put(true, (PCJ.myId() + 1) % PCJ.threadCount(),
      Vars.sequencer);
```

OnePivotConcurrentWrite it the fourth implementation, based on the third one, where each thread writes concurrently to different part of the same output file using memory-mapped file;

ConcurrentSend is the fifth implementation, based on third one, where the parts of buckets are sent during reading phase.

Codes for all the implementations are available at GitHub [7].

3 Performance evaluation

This section presents results obtained on the HPI Future SOC Lab 1000 Core Cluster during Fall 2019 period.

3.1 Hardware and software

Hardware
The 1000 Core Cluster consists of 25 computational nodes. The computing nodes are equipped with the 4 Intel Xeon E7-4870 processors, with max. 2.40 GHz clock speed. Each processor contains 10 cores, each core can run 2 threads in hyper-threading mode. In total it is possible to run 80 threads per node. There is installed 1 TB of RAM on each node, that is, more than 800 GB should be available for the user. Nodes are connected using 10-Gigabit Ethernet using Intel 82599ES 10-Gigabit Ethernet Controller. The storage for the users (`/home` directory) is a NAS-mounted drive with 4 TB capacity.

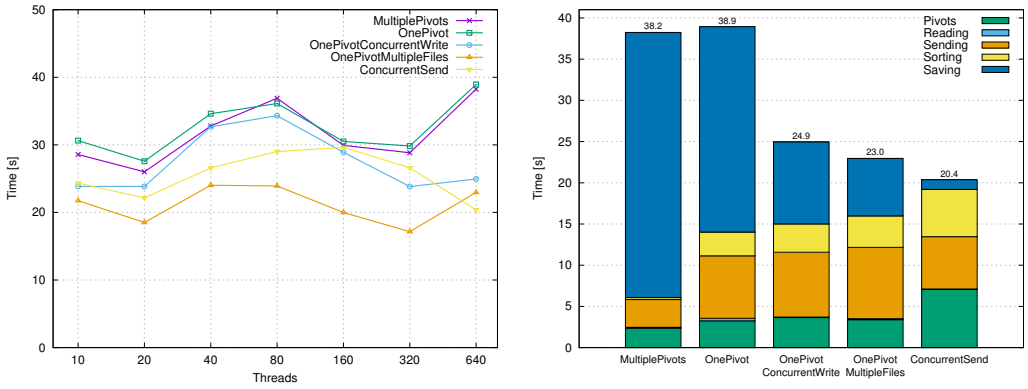
The benchmarks have used the maximum of 8 nodes for the computation.

Software
The operating system installed on the nodes was *Ubuntu* Linux in version 16.04.4 LTS (*Xenial Xerus*).

The *SLURM* (version 18.08.7) was used for submitting batch jobs to the cluster.

Java Virtual Machine from *Oracle JDK 13* package was used for the benchmarks of Java codes. Version 5.1.0 of the PCJ library was used.

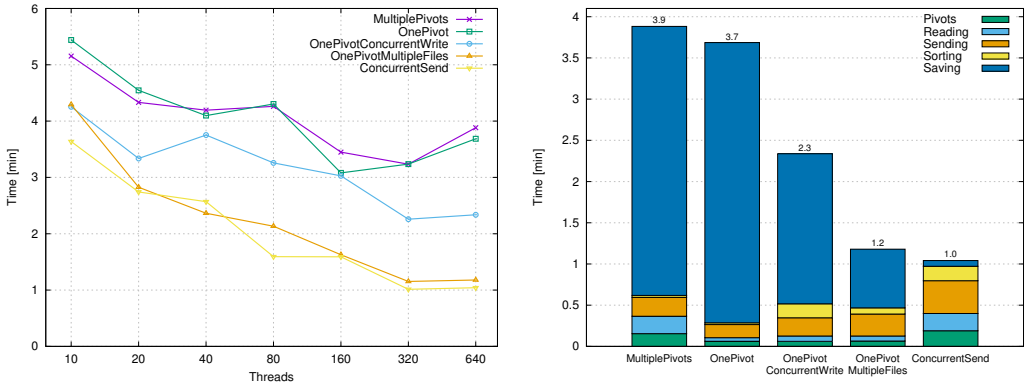
3.2 Results



(**a**) Sort time depending on the total number of threads

(**b**) Time consumed by each execution step using 640 threads

Figure 1: Sorting 10^7 elements

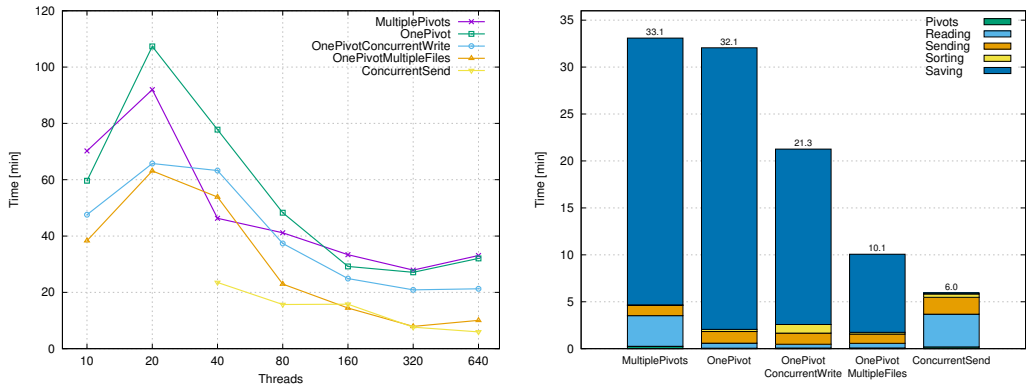


(**a**) Sort time depending on the total number of threads

(**b**) Time consumed by each execution step using 640 threads

Figure 2: Sorting 10^8 elements

Figure 1, 2 and 3 present total time needed to sort and save 10^7, 10^8 and 10^9 elements respectively. The plots on the left hand side presents the execution done



(**a**) Sort time depending on the total number of threads

(**b**) Time consumed by each execution step using 640 threads

Figure 3: Sorting 10^9 elements

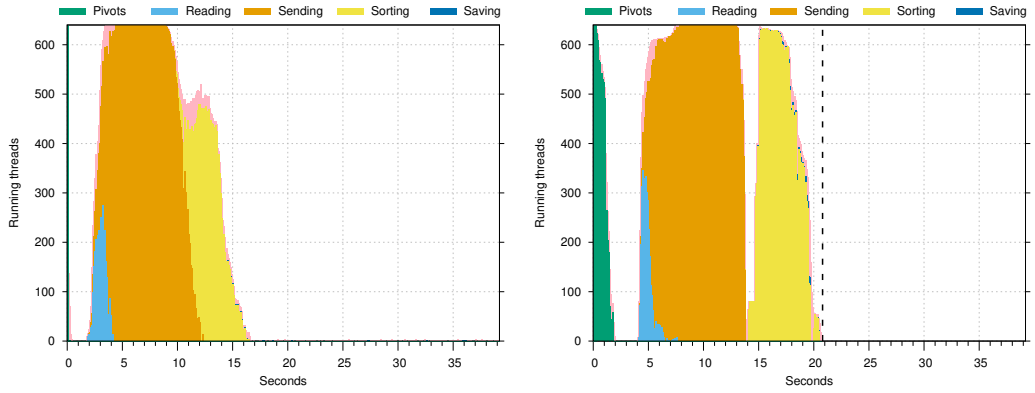
on the various configurations: using 10, 20, 40 and 80 threads on 1 node and 80 threads on 2, 4 and 8 nodes. On the right hand side plots presents time consumed by each step of execution using total 640 threads (8 nodes and 80 threads per node). However, the values are approximated, because the steps can overlap, as various threads can execute different steps in the same time.

The results show that the way of choosing pivots has little or even no impact on the performance—both *MultiplePivots* and *OnePivot* implementations show similar performance.

The writing part is the most important part for the performance. Even the writing is done through NFS by many threads in approximately the same time, the total writing time is much lower than sequential writing, where threads have to wait for their turn to write. However, *OnePivotMultipleFiles* and *ConcurrentSend* implementations creates multiple files that have to be combined to produce the single file result as *MultiplePivots* and *OnePivot* implementations. Time needed for concatenation can reduce gap between these two approaches. The concurrent writing done by *OnePivotConcurrentWrite* implementation, that is using memory-mapped file, is the intermediate solution—it allows concurrently filling single output file by multiple threads. Its performance is also in the middle between two other approaches.

Sending data while reading, when the bucket size is filled by 100 000 elements does not show the performance gains for most data size and thread count cases. It is probably due to too big limit for the size of the bucket, as there is visible some gains for 10^9-elements input and rather small number of threads (less than 80).

Figure 4 shows timeline of the number of threads that is executing corresponding step. There were 640 threads (8 nodes, 80 threads per node) processing 10^7 elements. The light pink color means that in the current time period some threads moved to the next step. The figure shows that the steps overlaps. The vertical dashed line on figure 4b means the execution finishes, but the X-axis is expanded to 39.2 seconds as this is the total execution time presented in figure 4a. The seemingly empty



(**a**) OnePivot (**b**) ConcurrentSend

Figure 4: Timeline of sorting 10^7 elements using 8 nodes and each node 80 threads

plot on figure 4a, starting at 17 second, presents sequentially writing output file by threads executing *OnePivot* implementation. Figure 4b does not have that behaviour, as threads of *ConcurrentSend* implementation use multiple output files.

4 Conclusion and future steps

The results of different implementation of sort algorithm show very good performance and generally, the observed behaviour is the expected one. The version with concurrently writing the single output file is not so fast as writing into separate files, but the latter requires joining files to provide exactly the same result, that takes additional time.

The performance of implementation that overlaps the reading and sending data is not as good as expected. That behaviour requires further investigation.

In the future steps the detailed performance comparison with TeraSort application of Apache Hadoop is planned.

Acknowledgements

The author would like to thank Future SOC Lab, Hasso Plattner Institute for Digital Engineering for awarding access to the 1000 Core Cluster.

References

[1] M. Nowicki. “Benchmarking Java on Ethernet Cluster”. In: *HPI Technical Report*. 2020. In press.

[2] M. Nowicki, Ł. Górski, and P. Bała. "PCJ–Java Library for Highly Scalable HPC and Big Data Processing". In: *2018 International Conference on High Performance Computing & Simulation (HPCS)*. IEEE. 2018, pages 12–20.

[3] M. Nowicki, M. Ryczkowska, Ł. Górski, M. Szynkiewicz, and P. Bała. "PCJ-a Java library for heterogenous parallel computing". In: *Recent Advances in Information Science (Recent Advances in Computer Engineering Series vol 36), WSEAS Press* (2016), pages 66–72.

[4] D. Pasetto and A. Akhriev. "A comparative study of parallel sort algorithms". In: *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*. ACM. 2011, pages 203–204.

[5] *PCJ at GitHub*. URL: https://github.com/hpdcj (last accessed 2020-03-31).

[6] *PCJ homepage*. URL: http://pcj.icm.edu.pl (last accessed 2020-03-31).

[7] *PCJ-TeraSort at GitHub*. URL: https://github.com/hpdcj/PCJ-TeraSort/tree/bff336474867c0b70b23fcea95444f51a34180d0 (last accessed 2020-03-31).

An Intelligent Intrusion Detection System

Sugandh Seth, Kuljit Kaur Chahal, and Gurwinder Singh

Department of Computer Science and Engineering
Guru Nanak Dev University, Amritsar

In recent years, the internet has become an integral part of all spheres of life. But, in this ever-progressing digital world, fortification of digital assets has become very essential. Intrusion Detection System (IDS) are widely deployed for detecting suspicious activities. Many intrusion detection systems look for signatures in network traffic to detect known attacks. Signature-based IDS are not efficient for detecting novel attacks. To overcome the above problem machine learning is widely used in Intrusion Detection Systems. But the majority of the proposed work is using the old IDS datasets that do not represent modern- day traffic. In this paper, we propose a Smart Intrusion Detection system using Extra Trees algorithm on the latest CIC-IDS 2018 dataset. An in-depth analysis of network parameters is also performed, which gives a deep insight into the variation of network parameters during the benign and malicious sessions.

1 Introduction

Data is the new oil. It is the biggest asset in this modernized digital world. We have witnessed a massive change in technology over time. From the era where everything started getting automated replacing the human manpower to an era where all systems are integrated with artificial intelligence overpowering the human brain. But with this ever-progressing technology, the cyber-attacks have also become way more sophisticated. The list of top 10 most critical risks related to web applications security, provided by OWASP (Open Web Application Security Project) indicates 'Injection' (including Structured Query Language (SQL), Operating System (OS), and Lightweight Directory Access Protocol (LDAP) injections) as a major vulnerability [3]. Other common attacks are DDOS (Distributed denial of Service), Session Hijacking, Cross-Site Scripting, CSRF(Cross-site request Forgery), XML-External Entities, Security Misconfiguration, Insecure Deserialization, and Insufficient Logging and Monitoring. Regardless of from which device the user is connected to the internet, the risk of data hacking and loss of privacy prevails. So, there is a need to develop a security mechanism that can protect our systems concerning all the vulnerable endpoints. Intrusion Detection System (IDS) are widely implemented [1] to safeguard digital assets against such cyber-attacks.

Contributions

To overcome the above gaps in research, this paper proposes a novel machine learning approach to implement a responsive IDS using the latest CIC IDS 2018 dataset. The major contributions of the paper are as follows

- Experimentation with the latest CIC IDS 2018 dataset to create a Realistic IDS that can detect the majority of the modern-day attacks.
- Proposed model has a high attack detection rate. The proposed model has a high attack detection rate of 100 % for various attack categories.
- A deep insight into the comparison between the network parameters observed during the benign and the malicious sessions.

2 Problem

Based on their identification criteria, IDSs are divided into two categories: anomaly and misuse [8]. Misuse based IDS are signature-based. The network activity is compared to signatures of documented vulnerabilities or attacks by misuse detection systems. Whereas, anomaly-based IDS compares the behavior of traffic to profiles [12]. The majority of the IDS are misused based though accurate but they fail to detect novel attacks. To overcome this gap machine and deep learning techniques are widely used in anomaly-based IDS to equip them to detect unknown attacks.

3 Solution

The right dataset is very crucial for building efficient IDS. Many datasets namely DARPA [4], KDD 99 [13], KOYOTO [14], NSLKDD [7] are commonly used for building IDS. The above-listed datasets are old and don't reflect modern-day traffic patterns. So the research done in this paper is using the latest CIC-IDS 2018 dataset that truly reflects the modern traffic trends. The CIC-IDS comprises 16 million rows with 80 features reflecting 13 modern-day attacks. The attacks are listed in table 1.

For building an intelligent IDS various machine learning techniques were applied to the CIC- IDS 2018 dataset. Random Forest [2], K- Nearest Neighbour [11], Extra trees [6], XGBoost [5], Naïve Bayes [10], and Decision Tree [9] machine learning algorithms were used during the experiments. To further enhance the results a voting-based ensemble model of RandomForest and K-Nearest Neighbor and KNN, DT and NB were also trained on the dataset.

In addition to the above machine learning models, a detailed analysis based on various network parameter were performed for benign and malicious sessions.

Table 1: Attack Categories in CIC-IDS 2018

S.no	Type
1.	Benign
2.	Bot
3.	Brute Force -Web
4.	Brute Force -XSS
5.	DDOS attacks-HOIC
6.	DDOS attacks-LOIC-UDP
7.	DDOS attacks-LOIC-HTTP
8.	DoS attacks- Golden Eye
9.	DoS attacks-Hulk
10.	Dos attacks-SlowHTTPTest
11.	Dos attacks-Slowloris
12.	FTP-BruteForce
13.	Infiltration
14.	Sql Injection

4 Used Future SOC Labs Resources

Used Future SOC Labs Resources For all the experimentation. We obtained access to 256 GB of RAM memory and GPU. On the provided resources by the lab, we ran python scripts for applying various machine learning models on a massive 16 million rows dataset for training the classification models and performing data analysis.

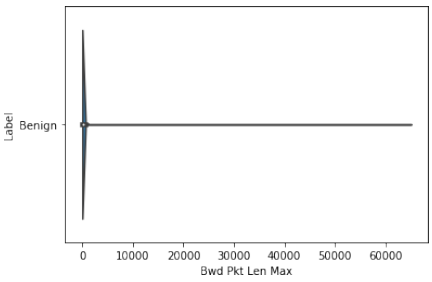
5 Findings

5.1 Feature Analysis

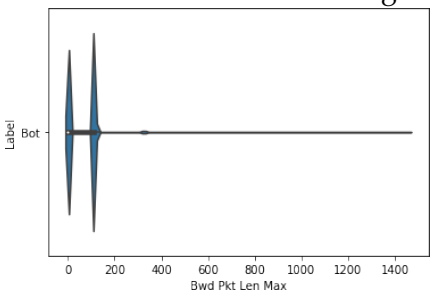
An in-depth analysis of various features of the dataset was performed. This analysis gives a deep insight into how various network parameters vary during the malicious and benign sessions. The analysis of four network parameters is done in the section below.

5.2 Bwd Pkt Len Max

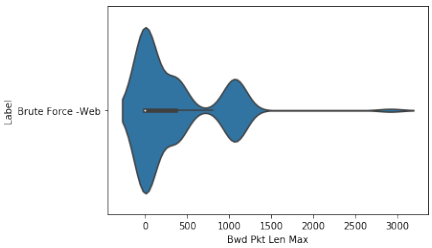
The table lists the variation of Bwd len Max for various attack categories.



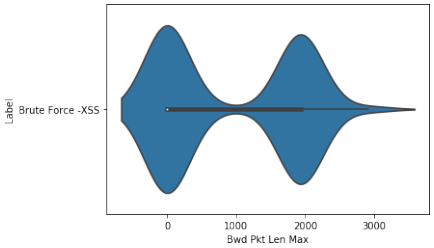
BENIGN



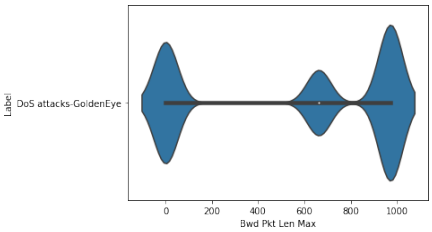
BOT



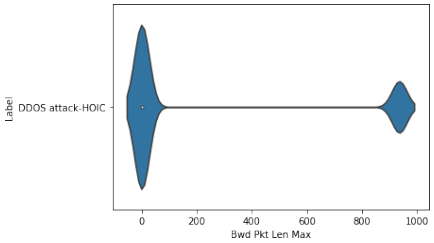
BRUTE FOREC WEB



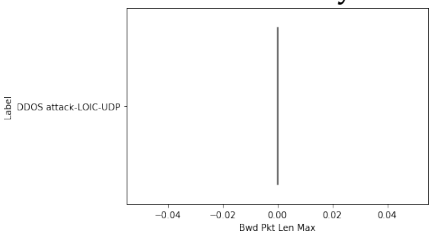
BRUTE FORCE -XSS



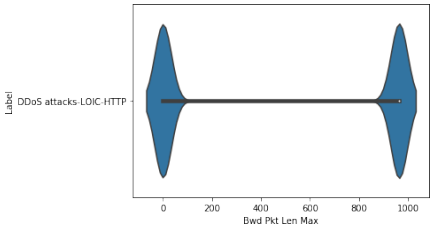
Dos attacksGoldenEye



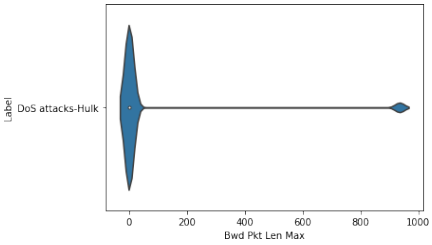
DOS Attack-HOIC



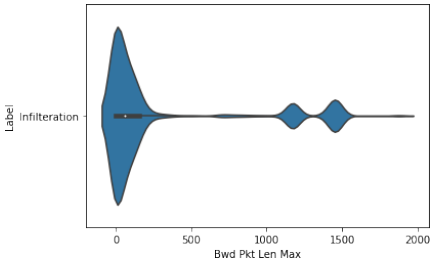
DDoS attack LOIC UDP



DDoS attack LOIC HTTP



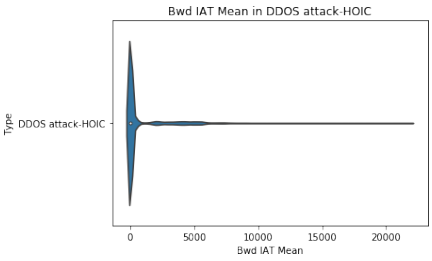
DDoS attacks Hulk



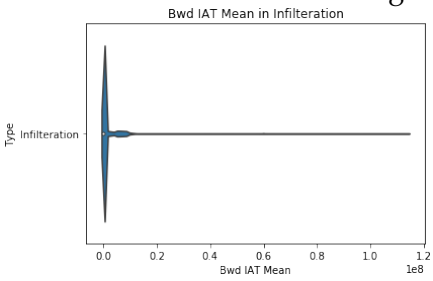
INFILTRATION

5.3 Bwd Iat Mean

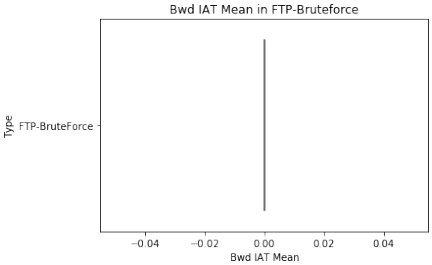
The table lists the variation of Bwd lat mean for various attack categories.



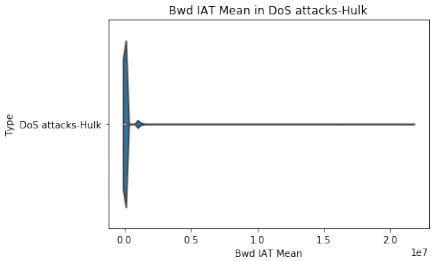
DDOS attack HOIC



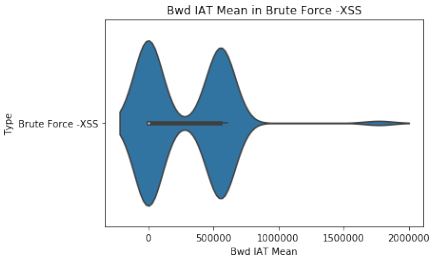
INFILTRATION



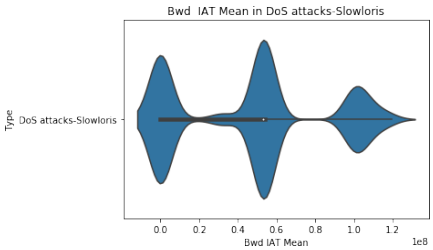
FTP-BruteForce



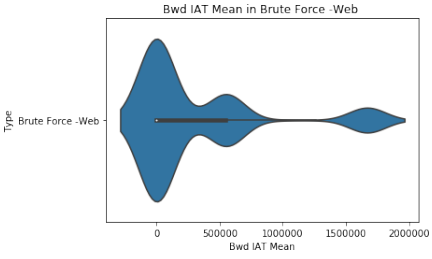
Dos attack Hulk



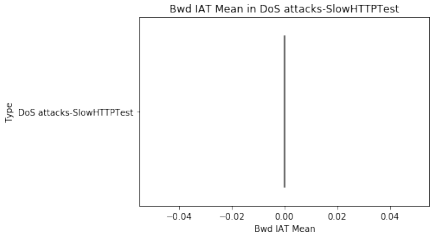
Brute Force XSS



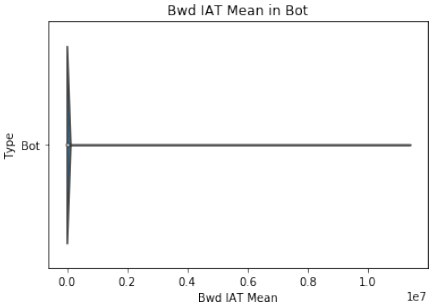
DOS attacks Slowris



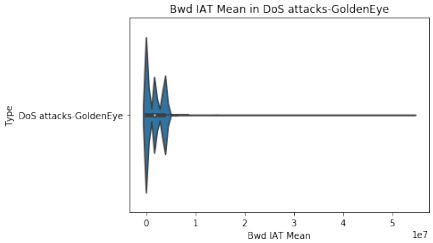
Brute Force-Web



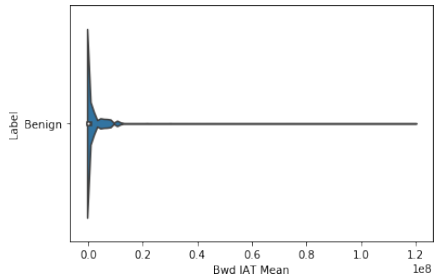
Dos attacks-SlowHTTpTest



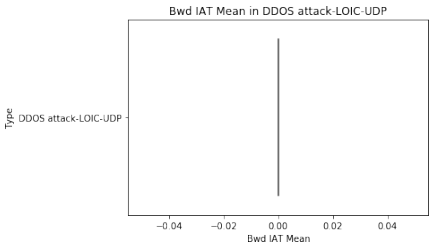
Bot



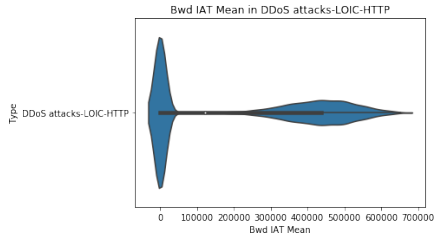
Dos attacks GoldenEye



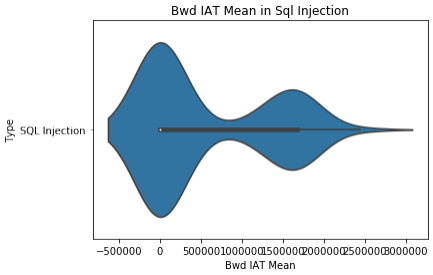
Benign



DDoS attacks LOIC-UDP



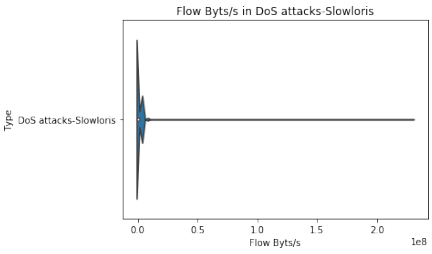
DDos attacks LOIC HTTp



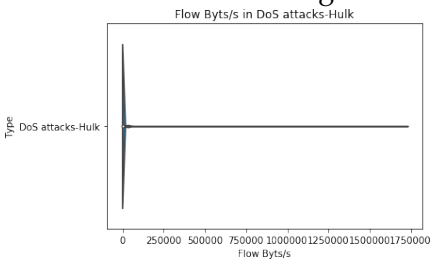
SQL Injection

5.4 Flow Bytes

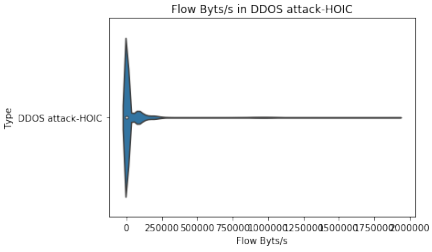
The table lists the variation Flow Bytes for various attack categories.



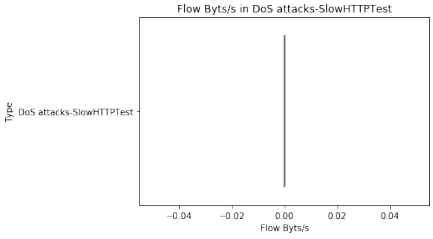
DOS attacks Sloworis



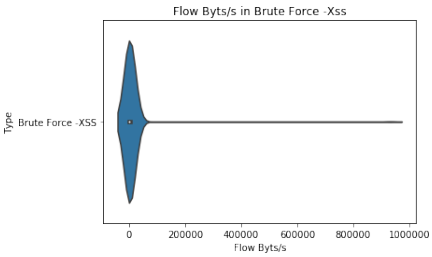
Dos attacks -Hulk



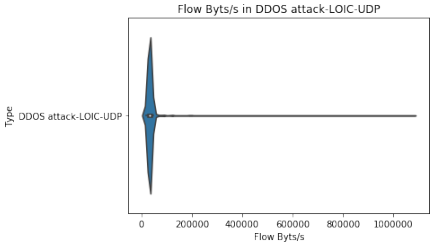
Dos attack HOIC



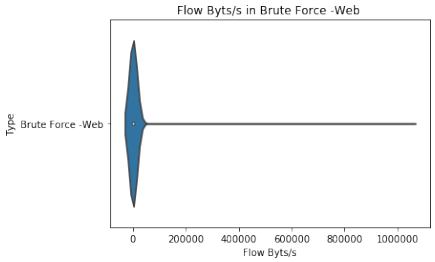
DoS attack SlowHTTPTest



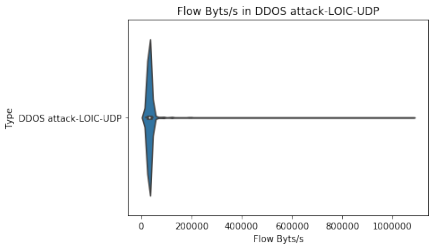
Brute Force -Xss



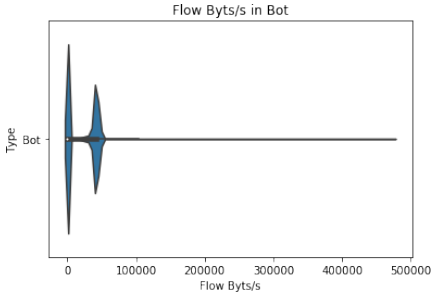
DDoS attack LOIC-UDP



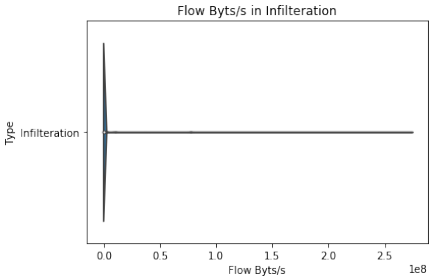
Brute Force Web



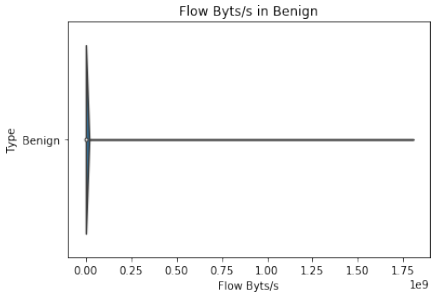
DDOS attack LOIC UDP



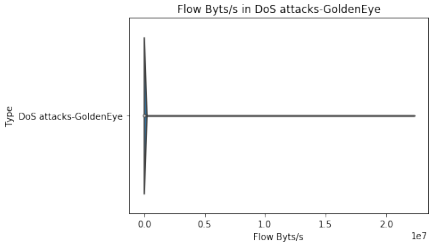
Bot



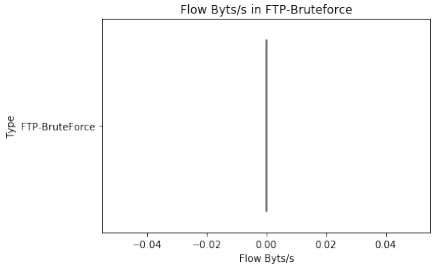
INFILTRATION



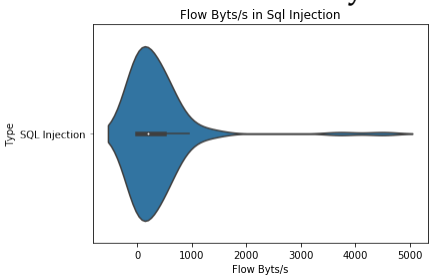
BENIGN



DoS attacks Golden Eye



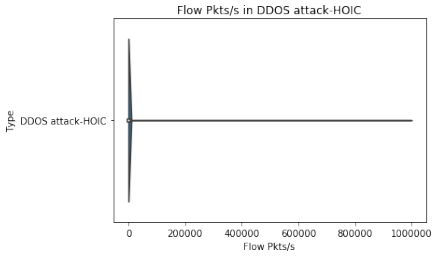
FTP-Brute Force



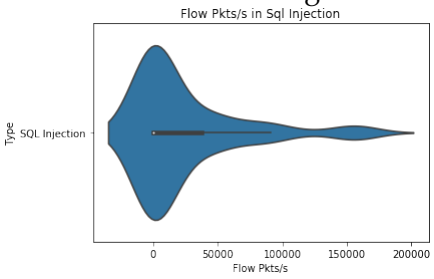
SQL Injection

5.5 Flow pkts

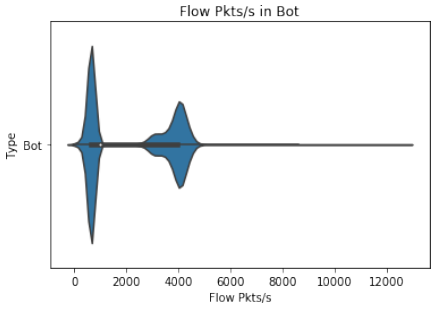
The table lists the variation Flow pkts for various attack categories.



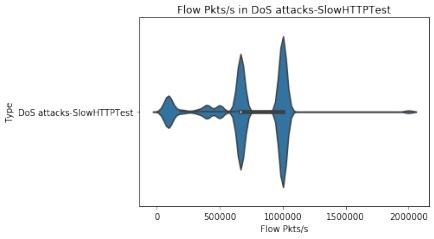
DDOS Attacks HOIc



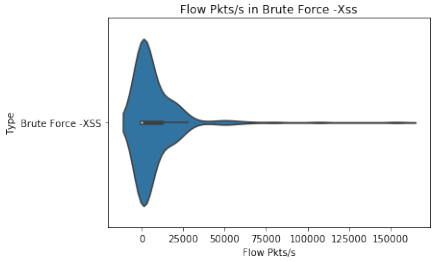
SQL INJECTION



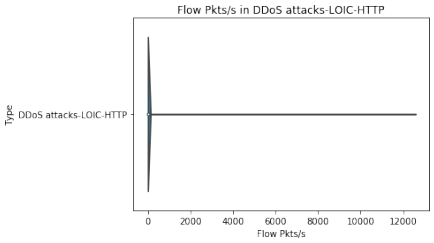
BOT



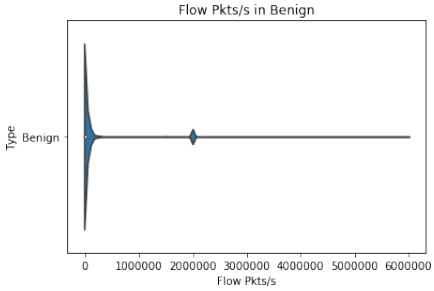
DoS attacks Slow HTTPTest



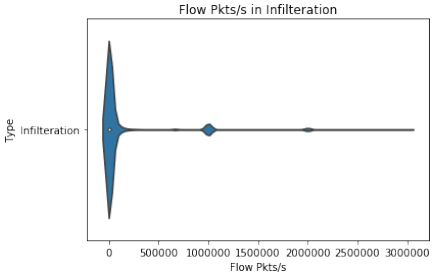
Brute Force -XSS



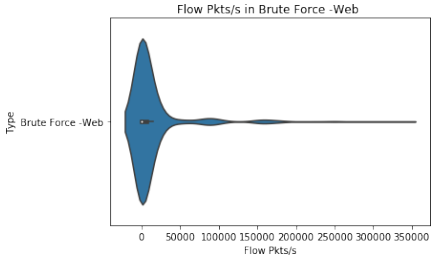
DDoS attacks LOIC HTTP



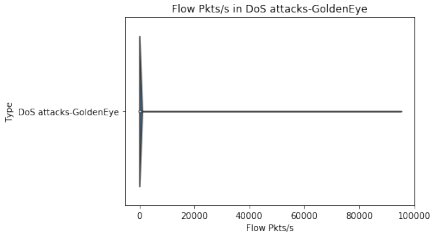
BENIGN



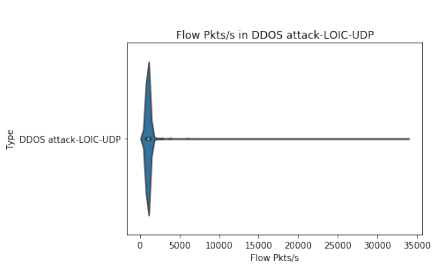
INFILTRATION



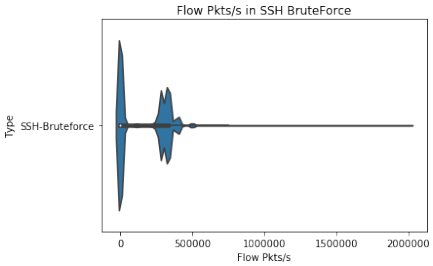
BRUTE FORCE WEB



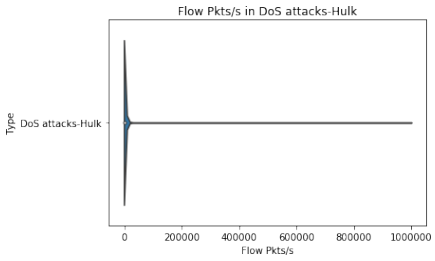
DOS attack GoldenEye



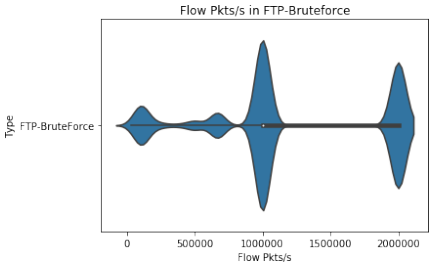
DDOS attack LOIC-UDP



SSH Bruteforce



DoS attacks Hulk



FTP BruteForce

5.6 Results

Comparison of performance of various machine learning model for detecting different attack types is listed in the table 2.

Table 2: Performance evaluation of machine learning models for different attack categories

Type	KNN	RF	NB	XGBOOST	DECISION TREE	EXTRATREES
Benign	99.83 %	43.67 %	78.86 %	99.92 %	97.02 %	93 %
Bot	99.99 %	99.34 %	98.11 %	99.53 %	99.06 %	100 %
Brute Force -Web	62.74 %	60.40 %	41.97 %	17.63 %	58.51 %	81 %
Brute Force -XSS	64.97 %	90.67 %	47.48 %	48.91 %	62.42 %	87 %
DDOS attack-HOIC	100.00 %	100.00 %	99.70 %	99.93 %	90.33 %	100 %
DDOS attack-LOIC-UDP	87.08 %	98.98 %	98.82 %	98.16 %	73.34 %	84 %
DDoS attacks-LOIC-HTTP	99.89 %	93.77 %	90.50 %	99.41 %	94.32 %	100 %
DoS attacks-GoldenEye	99.93 %	99.42 %	65.47 %	61.37 %	94.17 %	100 %
DoS attacks-Hulk	100.00 %	94.63 %	94.46 %	99.31 %	41.65 %	100 %
DoS attacks-SlowHTTPTest	69.11 %	99.26 %	99.10 %	68.06 %	30.09 %	2 %
DoS attacks-Slowloris	99.85 %	98.84 %	69.59 %	86.70 %	81.57 %	100 %
FTP-BruteForce	79.42 %	33.27 %	33.34 %	70.70 %	79.88 %	100 %
Infilteration	8.99 %	53.87 %	6.38 %	0.34 %	11.65 %	25 %
SQL Injection	44.04 %	65.50 %	41.70 %	0.00 %	36.06 %	55 %
SSH-Bruteforce	99.98 %	99.97 %	99.77 %	99.63 %	98.54 %	100 %

Different machine learning algorithms have varying detection capabilities for different attacks. The extra trees algorithm has a high attack detection rate of 100 % for eight attack categories. But, extra trees perform poorly especially DoS attacks SlowHTTPTest. Infiltration attack has low detection rate for all the machine learning

models with a max detection rate of 53.87 % by Random Forest classifier. SQL Injection attacks also have a max detection rate of 65 % by the Random Forest classifier.

To further enhance the detection rate two ensembles were trained. The first ensemble comprises KNN, DT, and the NB classifier. The second ensemble comprises KNN and RF. The performance of voting ensembles is listed in table 3.

Table 3: Comparing performance results of ensembles and Extra Trees classifier for various attack categories

Type	KNN+DT+NB	KNN+RF	Extra Trees
Benign	100 %	100 %	**93 %**
Bot	100 %	100 %	100 %
Brute Force -Web	32 %	44 %	81 %
Brute Force -XSS	43 %	43 %	87 %
DDOS attack-HOIC	100 %	100 %	100 %
DDOS attack-LOIC-UDP	97 %	84 %	**84 %**
DDoS attacks-LOIC-HTTP	99 %	100 %	100 %
DoS attacks-GoldenEye	96 %	100 %	100 %
DoS attacks-Hulk	99 %	100 %	100 %
DoS attacks-SlowHTTPTest	99 %	98 %	**2 %**
DoS attacks-Slowloris	72 %	99 %	100 %
FTP-BruteForce	37 %	52 %	100 %
Infilteration	2 %	1 %	25 %
SQL Injection	36 %	27 %	55 %
SSH-Bruteforce	100 %	100 %	100 %

It is evident from table 3 that the extra trees algorithm outperforms the voting ensembles for all attack categories except DDOS attack-LOIC-UDP and DoS attacks-SlowHTTPTest. Extra trees algorithm has a very high attack detection rate for Bot, DDoS attack-HOIC, DDoS attacks- LOIC-HTTP, DoS attacks-GoldenEye, DoS attacks-Hulk, DoS attacks-Slowloris , FTP- BruteForce, and SSH-Bruteforce attack.

6 Conclusion

With ever-progressing technology, cyber-attacks have also become way more sophisticated. Attackers can easily bypass weak IDS, so it is very essential to build modern IDS equipped with artificial intelligence. The majority of the research in this field was done on old datasets that do not reflect the current modern attacks. The main objective achieved with our research is a realistic Intelligent IDS with a high attack detection rate. In this paper, the latest CIC IDS 2018 dataset is used that truly reflects the modern-day traffic.

The extra trees algorithm achieved a high attack detection rate of 100 % for eight different attack categories. But, extra trees perform poorly especially DoS attacks SlowHTTPTest. Infiltration attack has a low detection rate for all the machine learning models with a max detection rate of 53.87 % by Random Forest classifier. SQL Injection attacks also have a max detection rate of 65 % by the Random Forest classifier.

References

[1] R. A. Ahmadian, A. Rasoolzadegan, and J. A. Javan. "A systematic review on intrusion detection based on the Hidden Markov Model". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal. 2018* 11.3 (2018), pages 111–134.

[2] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 1, 2001), pages 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[3] M. Choraś and R. Kozik. "Machine learning techniques applied to detect cyber attacks on web applications". In: *Logic Journal of the IGPL* 23.1 (Feb. 2015), pages 45–56. DOI: 10.1093/jigpal/jzu038.

[4] J. Chow. *An Assessment of the DARPA IDS Evaluation Dataset Using Snort*. 2005.

[5] S. S. Dhaliwal, A.-A. Nahid, and R. Abbas. "Effective Intrusion Detection System Using XGBoost". In: *Information* 9.7 (2018). ISSN: 2078-2489. DOI: 10.3390/info9070149.

[6] P. Geurts, D. Ernst, and L. Wehenkel. "Extremely randomized trees". In: *Machine Learning* 22 (Jan. 2021). DOI: 10.1007/s10994-006-6226-1.

[7] B. Ingre and A. Yadav. "Performance analysis of NSL-KDD dataset using ANN". In: *International Conference on Signal Processing and Communication Engineering Systems, 92–96*. 2015. DOI: 10.1109/spaces.2015.7058223.

[8] O. Joldzic, Z. Djuric, and P. Vuletic. "A transparent and scalable anomaly-based DoS detection method". In: *Computer Networks* 104 (2016). DOI: 10.1016/j.comnet.2016.05.004.

[9] M. Kumar, M. Hanumanthappa, and T. V. S. Kumar. "Intrusion Detection System using decision tree algorithm". In: *IEEE 14th International Conference on Communication Technology, Chengdu, China*. 2012, pages 629–634. DOI: 10.1109/ICCT.2012.6511281.

[10] S. Mukherjee and N. Sharma. "Intrusion Detection using Naive Bayes Classifier with Feature Reduction". In: *Procedia Technology* 4 (2012). DOI: 10.1016/j.protcy.2012.05.017.

[11] A. I. Saleh, F. M. Talaat, and L. M. Labib. "A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers". In: *Artificial Intelligence Review* 51 (2017), pages 403–443. DOI: 10.1007/s10462-017-9567-1.

[12] K. Scarfone and P. Mell. *Guide to intrusion detection and prevention systems (IDPS)*. NIST Special Publication 800-94, 2007. DOI: 10.6028/NIST.SP.800-94.

[13] K. Siddique, Z. Akhtar, A. Khan, F., and Y. Kim. "KDD Cup 99 Data Sets: A Perspective on the Role of Data Sets in Network Intrusion Detection Research". In: *Computer* 52.2 (2019). DOI: 10.1109/mc.2018.2888764.

[14] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao. "Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation". In: *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security - BADGERS '11*, 29–36. 2011. DOI: 10.1145/1978672.1978676.

Energy Efficiency, Virtualization and Performance
Fifth (WEEVIL5)

Carlos Juiz and Belen Bermejo

University of the Balearic Islands
{cjuiz,belen.bermejo}@uib.es

1 Project idea

The prevailing international scientific opinion on climate change is that human activities resulted in substantial global warming from mid-20^{th} century, and that continued growth in greenhouse gas concentrations, caused by human-induced emissions, is mainly (direct or indirectly) due to energy consumption. The sector where energy consumption has grown tremendously is IT (Information Technologies). On one hand, datacentres that host cloud services are becoming huge warehouses with lot of servers, additional electronic equipment and complex cooling systems. These computers and telecommunications networks are today primarily responsible for electronical energy consumption caused by datacentres, which maintain the quality of Internet service in operation. Giving more capacity to these datacentres usually means more cloud servers and consequently additional more equipment and cooling are needed, too. On the other hand, the increasing number of users, especially due to the popularization of cloud services, produced continuously capacity problems at datacentres due to performance requirements [6, 7]. Thus, a new challenge directly related to this research area emerges: the energy efficiency of cloud servers. WEEVILT project is aimed by this combination of these topics. Our main project objectives are:

1. The characterization of the performance and energy consumption from consolidated servers through benchmarking and monitoring techniques.
2. Modelling the energy consumption patterns and performance of consolidated servers. We shall infer models of such energy, performance and virtualization, together.
3. Performing several comparative studies of benchmarking between physical and virtual servers.
4. The validation of our performance-oriented previous findings and energy efficiency virtualization models through experimentations in real datacenters, as HPI Future SOC Lab.

The WEEVIL5 project is defined as the extension of the previous one (WEEVILF) developed using the HPI Future SOC Lab infrastructure (RX600S5-1 server). This project allowed the finalization of Belen Bermejo PhD thesis.

2 Used Future SOC Lab resources

In order to answer the research question, we design a methodology based in the following stages:

1. The characterization of performance from consolidated servers through benchmarking and monitoring techniques [9].
2. Modelling the performance degradation of consolidated servers. We shall infer models of such energy, performance and virtualization, together.
3. Performing several comparative studies of benchmarking between physical and virtual servers.
4. The validation of our performance-oriented previous findings of virtualization models through experimentation in real datacentres, as HPI Future SOC Lab.

The exposed methodology was performed in the HPI Future SOC Lab infrastructure, specifically in the rx600s5-1 server which hardware has 48 CPUs and 1024 GB of RAM memory. We can see the actions developed in each stage and the correspondent outcome, as follows:

- Stage 1:
  - Characterizing the performance of PM through benchmarking and monitoring techniques
  - PM's performance when consolidating virtual machines
  - PM's performance when varying the % of CPU utilization
- Stage 2:
  - Description of virtual machine consolidation behaviour and CiS^2 index
- Stage 3:
  - Measuring and quantifying the virtual machine consolidation overhead in HPI infrastructure and UIB's servers
- Stage 4:
  - Discussion and results' analysis
  - Comparison between UIB and HPI servers in terms of performance
  - Comparison between UIB and HPI servers in terms of CiS^2 index [8].

3 Findings and results

In previous section we explain the stages we perform to answer the research question, as well as the developed actions and its outcomes. In this section we summarize some

of the main funding resulting of this projects, which are in-depth explained in the related publication [8, 3, 4].

Moreover, the most important milestone of this project was the finalization of the Belen Bermejo thesis, which was submitted on 1st October 2020.

Publications

The use of the HPI infrastructure collaborates to develop the following research works:

1. B. Bermejo, C. Juiz, and C. Guerrero. "On the Linearity of Performance and Energy at Virtual Machine Consolidation: the CiS2 Index for CPU Workload in Server Saturation". In: *Proceedings of the IEEE 20th International Conference on High Performance Computing and Communications*. Exeter, UK, 2018, pages 928–933
2. B. Bermejo, C. Juiz, and C. Guerrero. "Virtualization and consolidation: a systematic review of the past 10 years of research on energy and performance". In: *The Journal of Supercomputing* 75.2 (2019), pages 808–836. ISSN: 1573-0484. DOI: 10.1007/s11227-018-2613-1
3. B. Bermejo, C. Juizo, and N. Thomas. "On the virtualization overhead and energy consumption in consolidated servers". In: *UK- Performance Engineering Workshop (UKPEW)*. Newcastle upon Tyne, UK, 2018
4. B. Bermejo and C. Juiz. "Virtual machine consolidation: a systematic review of its overhead influencing factors". In: The Journal of Supercomputing (2019). *Accepted and pending of publication.*
5. C. Juiz and B. Bermejo. "The CiS^2: a new metric for performance and energy trade-off in consolidated servers". In: *Cluster Computing* 23.4 (2020), pages 2769–2788. ISSN: 1573-7543. DOI: 10.1007/s10586-019-03043-8
6. B. Bermejo and C. Juiz. "On the classification and quantification of server consolidation overheads". In: *The Journal of Supercomputing* (2020), pages 1–21
7. B. Bermejo and C. Juiz. "Virtual machine consolidation: a systematic review of its overhead influencing factors". In: *The Journal of Supercomputing* 76.1 (2020), pages 324–361. ISSN: 1573-0484. DOI: 10.1007/s11227-019-03025-y

4 Next steps

In this project we attempt to answer the research question: "How to determine the performance degradation of physical servers due to Virtual Machine Consolidation when varying the % of physical CPU?". For that, we design a methodology based on monitoring and benchmarking techniques and we apply it to our servers and then, we extend the experiment to HPI infrastructure.

The HPI allows us to extend the work and also to obtain more accurate results and conclusions. Due to that, there are very interesting research questions that we would like to answer with the support to the HPI infrastructure.

In order to answer the proposed question, we will apply for a new project in the Hasso Plattner Institute in order to use the HPI Future SOC Lab's IT infrastructure.

References

[1] B. Bermejo, C. Juiz, and C. Guerrero. "On the Linearity of Performance and Energy at Virtual Machine Consolidation: the CiS2 Index for CPU Workload in Server Saturation". In: *Proceedings of the IEEE 20th International Conference on High Performance Computing and Communications*. Exeter, UK, 2018, pages 928–933.

[2] B. Bermejo, C. Juizo, and N. Thomas. "On the virtualization overhead and energy consumption in consolidated servers". In: *UK- Performance Engineering Workshop (UKPEW)*. Newcastle upon Tyne, UK, 2018.

[3] B. Bermejo and C. Juiz. "On the classification and quantification of server consolidation overheads". In: *The Journal of Supercomputing* (2020), pages 1–21.

[4] B. Bermejo and C. Juiz. "Virtual machine consolidation: a systematic review of its overhead influencing factors". In: *The Journal of Supercomputing* 76.1 (2020), pages 324–361. ISSN: 1573-0484. DOI: 10.1007/s11227-019-03025-y.

[5] B. Bermejo, C. Juiz, and C. Guerrero. "Virtualization and consolidation: a systematic review of the past 10 years of research on energy and performance". In: *The Journal of Supercomputing* 75.2 (2019), pages 808–836. ISSN: 1573-0484. DOI: 10.1007/s11227-018-2613-1.

[6] B. Bermejo, S. Filiposka, C. Juiz, B. Gómez, and C. Guerrero. "Improving the Energy Efficiency in Cloud Computing Data Centres Through Resource Allocation Techniques". In: *Research Advances in Cloud Computing*. Edited by S. Chaudhary, G. Somani, and R. Buyya. Singapore: Springer Singapore, 2017, pages 211–236. ISBN: 978-981-10-5026-8. DOI: 10.1007/978-981-10-5026-8_9.

[7] R. Buyya, C. Vecchiola, and S. T. Selvi. *Mastering Cloud Computing: Foundations and Applications Programming*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013. ISBN: 978-0-12-409539-7.

[8] C. Juiz and B. Bermejo. "The CiS^2: a new metric for performance and energy trade-off in consolidated servers". In: *Cluster Computing* 23.4 (2020), pages 2769–2788. ISSN: 1573-7543. DOI: 10.1007/s10586-019-03043-8.

[9] X. Molero, C. Juiz, and M. Rodeño. *Evaluación y Modelado del Rendimiento del os Sistemas Informáticos*. Pearson, 2004. DOI: 10.2200/S00516ED2V01Y201306CAC024.

Designing practical algorithms through overfitting

Markus Wagner

Optimisation and Logistics Group
University of Adelaide, Australia
markus.wagner@adelaide.edu.au

Since 2017, the teams around Prof. Tobias Friedrich (HPI, Chair for Algorithm Engineering) and Dr Markus Wagner (University of Adelaide, Australia) have explored the concept of automated algorithm configuration to design new search operators. This project builds upon the existing work and will push it toward real-world, interconnected problems.

While the HPI Future SOC Lab infrastructure has already allowed us to perform a great number of experiments, we have barely been able to scratch the surface. Additional experiments will be needed to further explore the field of Data-Driven Search-Based Software Engineering (DSE) that we have founded. DSE requires to be driven by potent hardware. DSE is in its infancy, and it will become increasingly important—see the following claim from our inaugural DSE paper [6]:

> Claim4: Data mining without optimisation should be deprecated. Conclusions reached from an unoptimised data miner can be refuted, just by running the same tuned learner on the same data [...]. Since they can be so easily refuted, this community should stop publishing analytics papers that lack an optimisation component. [6]

COVID19, disclaimer

Due to the pandemic, this project has made no noteworthy progress besides the publication of a paper, https://link.springer.com/article/10.1007/s10732-020-09457-7 (accepted on 02 September 2020). There, we acknowledge the HPI FSOC's team for enabling the research. The remainder of this report is identical to the previous report as it covers the paper that had been in the works back then.

1 Introduction

Data-Driven Search-Based Software Engineering (DSE) [6] and the recent generalisation Data Mining Algorithms Using/Used-by optimisers (DUO) [1] combine insights from Mining Software Repositories (MSR) and Search-based Software Engineering (SBSE). While MSR formulates software engineering problems as data mining problems, SBSE reformulates SE problems as optimisation problems and use

meta-heuristic algorithms to solve them. Both MSR and SBSE share the common goal of providing insights to improve software engineering. The algorithms used in these two areas also have intrinsic relationships. Combining these two fields is useful for situations (a) which require learning from a large data source or (b) when optimisers need to know the lay of the land to find better solutions, faster.

Contributions to date

Prof. Tobias Friedrich and I have employed the DSE concept in 2017 to investigate heuristic search approaches to extensively explore the design space of heuristics for certain optimisation problems. We distilled from the results new designs which our team was able to analyse theoretically [4, 5], proving that the new approaches performed asymptotically better than state-of-the-art approaches.

So far, we have only investigated relatively simple single-objective problems, which can be analysed theoretically, such as linear functions and some submodular functions. What is missing is the connection to single- and multi-objective real-world problems. This is what this project aims to do. We will use automated algorithm configuration to optimise heuristics for classes of problems and to families of problem instances. We are hoping to achieve breakthroughs similar to the ones achieved to date. Recent results include our work on topic modelling and on relating good parameter configurations with properties of the respective text corpora [8].

2 Problems with Interconnected Components

In optimisation research, problems with different characteristics are investigated. To find an appropriate algorithm for a practical problem often assumptions about characteristics are made, and then a suitable algorithm is chosen or designed. For instance, an optimisation problem can have several components interacting with each other. Because of their interaction they form an interwoven system where interdependencies in the design and the objective space exist. An optimal solution for each component independently will in general not be a good solution for the interwoven optimisation problem.

In order to provide an academic interwoven optimisation test problem, the Traveling Thief Problem (TTP) was proposed in 2013 [7] where two well-known subproblems, the Traveling Salesman Problem (TSP) and the Knapsack Problem (KP), interact with each other. As in the TSP problem a so-called thief has to visit each city exactly once. In addition to just travelling, the thief can make profit during his tour by stealing items and putting them in the rented knapsack. However, the thief's traveling speed decreases depending on the current knapsack weight, which then increases the rent that the thief has to pay for the knapsack. The TTP's canonical formulation is a single-objective one, however, because the travelling time and profit represent solutions with different trade-offs, the problem is bi-objective in nature. To date, pretty much no research has considered this.

3 Solution

Our method is based on a genetic algorithm with customisation addressing problem characteristics, because we address the BI-TTP, a bi-objective version of the TTP, where the goal is to minimise the overall travelling time and to maximise the profit of the collected items. We incorporate domain knowledge through a combination of near-optimal solutions of each subproblem in the initial population and a custom repair operation to avoid the evaluation of infeasible solutions. Moreover, the independent variables of the TSP and KP components are unified to a real variable representation by using a biased random-key approach. The bi-objective aspect of the problem is addressed through an elite population extracted based on the non-dominated rank and crowding distance of each solution. We provide a comprehensive study which shows the influence of parameters on the performance of our method and investigate the performance of each parameter combination over time.

Non-Dominated Sorting Biased Random-Key Genetic Algorithm (NDS-BRKGA)

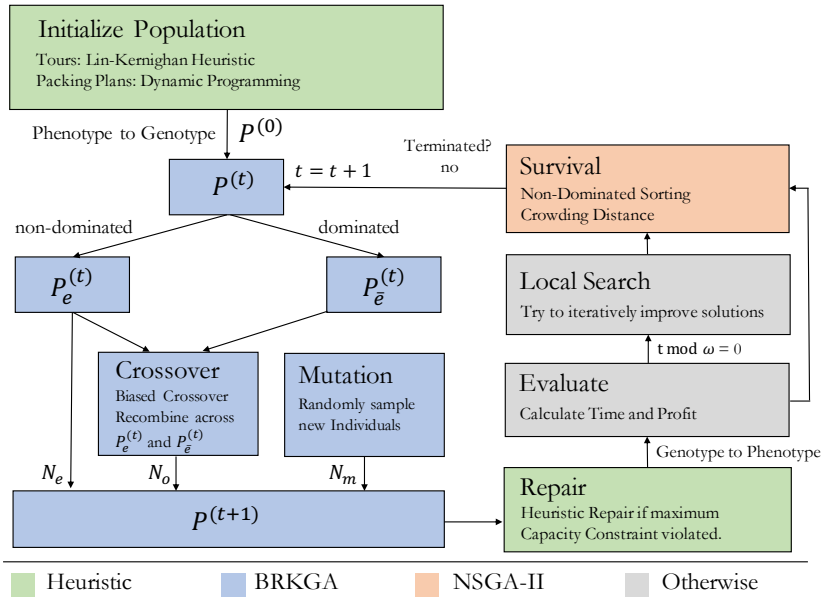


Figure 1: NDS-BRKGA: A customised genetic algorithm

Figure 1 illustrates the overall procedure of our highly-configurable NDS-BRKGA. At first, we generate the initial population using efficient solvers for the subproblems independently. Afterward, we combine the optimal or near-optimal solutions for both subproblems and convert them to their genotype representation which results in the initial population. For the purpose of mating, the population is split into an

elite population $P_e^{(t)}$ and non-elite population $P_{\bar{e}}^{(t)}$. The individuals for the next generations $P^{(t+1)}$ are a union of the elite population $P_e^{(t)}$ directly, the offspring of a biased crossover and mutant individuals. In case an individual violates the maximum capacity constraint, we execute a repair operation. Then, we convert each individual to its corresponding phenotype and evaluate it on the problem instance. In order to insert an explicit exploitation phase in our algorithm, we apply at some evolutionary cycles a local search procedure in some elite individuals. Finally, the survival selection is applied and if the termination criterion is not met, we increase the generation counter t by one and continue with the next generation. In the following, we describe the purpose of each of the design decisions we have made and explain what role it plays during a run of the algorithm.

4 Implementation

We have used the so-called 1000-core cluster of the HPI Future SOC Lab for the exhaustive hyper-parameter Study. Running all 3072 combinations (see table 1) on 9 instances, and performing 10 independent runs of 5 hours each consumed over 160 CPU years in the final experiments.

The parameters were population size N, elite population size N_e, mutant population size N_m, elite allele inheritance probability p_e, fraction $\alpha$of the initial population created from TSP and KP solvers, and the frequency ω for local search.

5 Evaluation

In figure 2, we visualise the best parameter configurations at six different execution times. In each plot the best obtained parameter configuration regarding hypervolume is highlighted in red and parameter configurations up to 0.1 % worse than the best are highlighted in blue. Note that the importance of values of each parameter among

Table 1: Parameter values considered during the experiment. The values highlighted in red have been added since the last report, and their addition resulted in the use of an additional 100 CPU years of runtime.

Parameter	Values
N	100, 200, 500, 1000
N_e	$0.3N$, $0.4N$, $0.5N$, $0.6N$
N_m	$0.0N$, $0.1N$, $0.2N$
ρ_e	0.5, 0.6, 0.7, 0.8
α	0.0, 0.1, 0.2 , 0.3
ω	1, 10, 50, 100

the best parameter configurations is indicated by the intensity of the blue color once some parameter configurations share some parameter values. The following can be observed:

1. **More execution time, better results:** The number of parameter configurations that are capable of generating large hypervolume values increases as the execution time of our algorithm increases. This means that in some runs even though the parameters have not been set appropriately, the algorithm is still able to converge.

2. **Importance of TSP and KP solvers:** It has influence on the overall performance of the algorithm if TSP and KP solvers are used for initialization which is determined by α. The best results are obtained if at least 10 % of the initial solutions are biased towards those solutions found a TSP and KP solvers.

3. **Trends when execution time increases**: We can see a trend as the execution time increases. Our method performs better with a large population, a large survival rate, a small or no explicit diversification through mutant individuals, a small influence of single-parent inheritance, an minor influence of a good initial population, and significant influence of local search procedure.

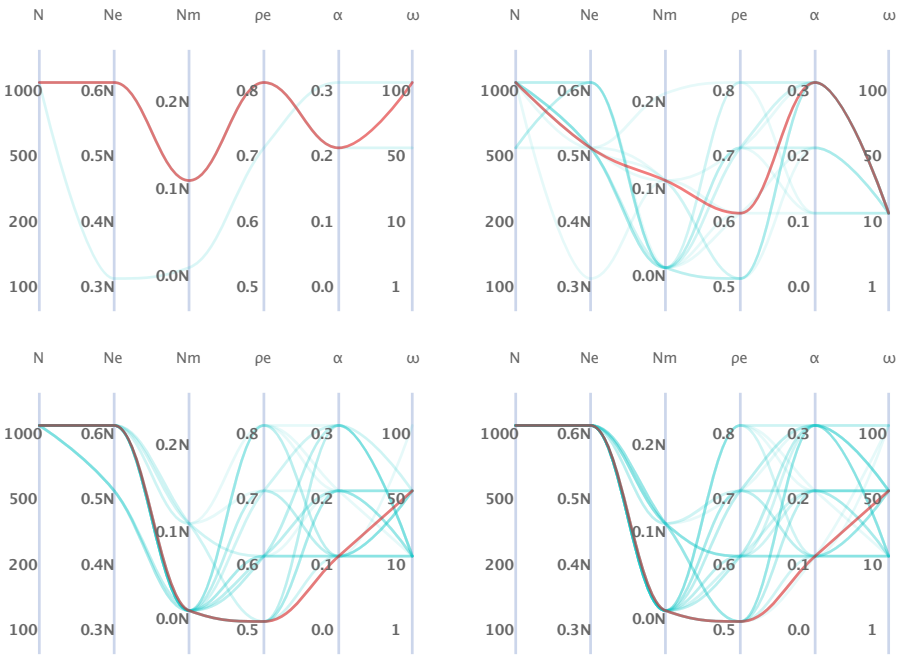


Figure 2: Best parameter configurations over all instances with varying execution times. From top left to bottom right: 600/1800/7200/18000 seconds.

Comparison with single-objective TTP solutions

As the BI-TTP is related to the original TTP (which has been subject to many studies), we now build the bridge to it. Therefore, we compare our results with the best known single-objective TTP objective scores, which come from a comprehensive comparisons of a variety of algorithms reported in [9]. The computational budgets of the approaches which have obtained the best known solutions might vary.

Table 2 compares for each instance the best known score of the TTP with the best score found by our algorithm when it optimised the BI-TTP. Note that despite the strong connection of the BI-TTP to the single-objective TTP, maximising the single-objective TTP objective score is not an explicit goal of the BI-TTP. Nevertheless, NDS-BRKGA has found better scores for the three smallest instances with 280 cities with up to 2790 items.

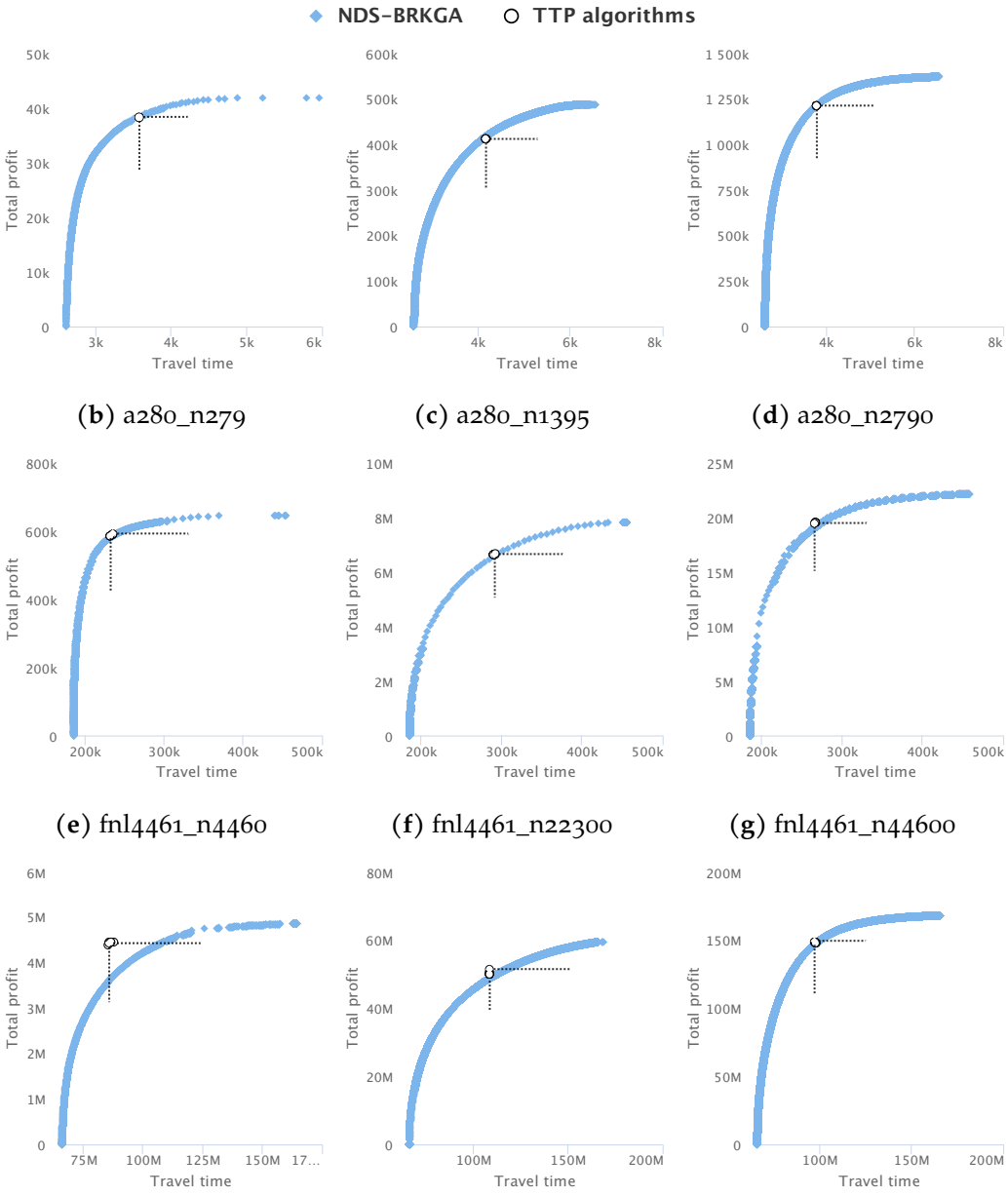
Table 2: Single-objective comparison of the TTP objectives scores

Instance	TTP score*	NDS-BRKGA
a280_n279	18 470.000[a]	18 603.120
a280_n1395	110 147.219[b]	115 445.521
a280_n2790	429 081.783[c]	429 085.353
fnl4461_n4460	263 040.254[d]	257 394.821
fnl4461_n22300	1 705 326.000[a]	1 567 933.421
fnl4461_n44600	6 744 903.000[a]	6 272 240.702
pla33810_n33809	1 863 667.592[d]	1 230 174.003
pla33810_n169045	15 634 853.130[e]	12 935 090.876
pla33810_n338090	58 236 645.120[f]	55 688 288.508

* Available at https://cs.adelaide.edu.au/~optlog/research/combinatorial.php.
[a] Obtained from CS2SA approach proposed by [2] [b] obtained from C3 approach [c] obtained from C4 approach [d] obtained from S5 approach [e] obtained from S1 approach [f] obtained from C6 approach. All these last approaches have been proposed by [3].

In figure 3, we plot the 100 % attainment surface for each instance showing the values of the two objectives of all non-dominated solutions found by our algorithm,[1] as well as for the TTP solutions found by running 10 times the best algorithm as identified in [9] for each instance. In order to better analyse the results, we delimit the region dominated by the TTP solutions by dotted lines. For each instance, the TTP solutions found have presented similar quality concerning their scores, as we can see by the superimposition of the TTP solutions in the plots. Note that in almost all

[1] Available at https://github.com/jonatasbcchagas/nds-brkga_bi-ttp.



(**b**) a280_n279 (**c**) a280_n1395 (**d**) a280_n2790

(**e**) fnl4461_n4460 (**f**) fnl4461_n22300 (**g**) fnl4461_n44600

(**h**) pla33810_n33809 (**i**) pla33810_n169045 (**j**) pla33810_n338090

Figure 3: NDS-BRKGA solutions and the best known TTP solutions

instances, few or none of our solutions have been dominated. The largest number of dominated solutions has been obtained for the instance *pla33810_n33809*, in which our algorithm has had the worst performance concerning the other solutions approaches.

6 Future Work

While the infrastructure allowed us to perform a great number of experiments, we have barely been able to scratch the surface. Additional experiments will be needed to further explore the data-driven design of our custom approach to this problem.

Among the next steps is the write-up of the preliminary findings in a scientific article—which will be submitted following HPI's review of the draft. We are currently finalising the article for submission to the Journal of Heuristics.

We plan to apply for an extension of the current project, as DSE requires to be driven by potent hardware. DSE is in its infancy, and it will become increasingly important—see the following claim from our inaugural DSE paper:

> Claim4: Data mining without optimisation should be deprecated. Conclusions reached from an unoptimised data miner can be refuted, just by running the same tuned learner on the same data [...]. Since they can be so easily refuted, this community should stop publishing analytics papers that lack an optimisation component. [6]

References

[1] A. Agrawal, T. Menzies, L. L. Minku, M. Wagner, and Z. Yu. *Better Software Analytics via "DUO": Data Mining Algorithms Using/Used-by Optimizers*. 2018. arXiv: 1812.01550 [cs.SE].

[2] M. El Yafrani and B. Ahiod. "Population-based vs. single-solution heuristics for the travelling thief problem". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM. 2016, pages 317–324.

[3] H. Faulkner, S. Polyakovskiy, T. Schultz, and M. Wagner. "Approximate approaches to the traveling thief problem". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM. 2015, pages 385–392.

[4] T. Friedrich, A. Göbel, F. Quinzan, and M. Wagner. "Heavy-Tailed Mutation Operators in Single-Objective Combinatorial Optimization". In: *Parallel Problem Solving from Nature – PPSN XV*. Edited by A. Auger, C. M. Fonseca, N. Lourenço, P. Machado, L. Paquete, and D. Whitley. Cham: Springer International Publishing, 2018, pages 134–145. ISBN: 978-3-319-99253-2.

[5] T. Friedrich, F. Quinzan, and M. Wagner. "Escaping Large Deceptive Basins of Attraction with Heavy-tailed Mutation Operators". In: *Proceedings of the Genetic and Evolutionary Computation Conference*. GECCO '18. Kyoto, Japan: ACM, 2018, pages 293–300. DOI: 10.1145/3205455.3205515. URL: http://doi.acm.org/10.1145/3205455.3205515.

[6] V. Nair, A. Agrawal, J. Chen, W. Fu, G. Mathew, T. Menzies, L. Minku, M. Wagner, and Z. Yu. "Data-driven Search-based Software Engineering". In: *Proceedings of the 15th Int. Conf. on Mining Software Repositories*. MSR '18. Gothenburg, Sweden: ACM, 2018, pages 341–352. DOI: 10.1145/3196398.3196442. URL: http://doi.acm.org/10.1145/3196398.3196442.

[7] S. Polyakovskiy, M. R. Bonyadi, M. Wagner, Z. Michalewicz, and F. Neumann. "A Comprehensive Benchmark Set and Heuristics for the Traveling Thief Problem". In: *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*. GECCO '14. Vancouver, BC, Canada: ACM, 2014, pages 477–484. DOI: 10.1145/2576768.2598249. URL: http://doi.acm.org/10.1145/2576768.2598249.

[8] C. Treude and M. Wagner. "Predicting Good Configurations for GitHub and Stack Overflow Topic Models". In: *Proceedings of the 16th International Conference on Mining Software Repositories*. MSR '19. Montreal, Quebec, Canada: IEEE Press, 2019, pages 84–95. DOI: 10.1109/MSR.2019.00022. URL: https://doi.org/10.1109/MSR.2019.00022.

[9] M. Wagner, M. Lindauer, M. Mısır, S. Nallaperuma, and F. Hutter. "A case study of algorithm selection for the traveling thief problem". In: *Journal of Heuristics* 24.3 (June 2018), pages 295–320. ISSN: 1572-9397.

Behavior-based authentication

Feature engineering and performance evaluation based on large user profiles

Vera Weidmann and Marvin Mirtschin

neXenio GmbH
vera.weidmann@nexenio.com
marvin.mirtschin@nexenio.com

1 Introduction

Our project contributes to the growing interest of mobile and behavior-based authentication systems. Next to fingerprints and facial features, another meaningful authentication method is a person's gait pattern. Our smartphones are equipped with a variety of sensors and these can measure the environment and behavior surrounding our phones. We carry these phones with us around the clock, even closely attached to our bodies. Accordingly, data corresponding to our body movement is generated. Research shows that these recorded data signals can be shaped to a unique signature of the phone's owner. The authentication accuracy, though, is highly dependent on the users' physiological and environmental circumstances. The more situations that are covered by the user's training state, the more the system's precision is challenged. In this context, statistical investigations and machine learning algorithms are applied.

Our research team at neXenio confronts behavior-, in particular gait-based authentication, with an enormous data set, covering six thousand walking sequences that have been recorded over the last years.

This semester, our fourth period at Future SOC Lab, we refined signal processing and feature engineering. In addition we attached a indoor positioning system to this gait-based authentication approach. The aim was to show a reliable system that enables gait-based authentication while walking towards an access control system.

2 Behavior-based authentication

Nowadays, smartphones can be unlocked via password, fingerprint, or, as more recently introduced, face recognition. The last two methods are biometric authentication methods, which use user related characteristics, e.g. the friction ridges of the finger or facial features, to recognize people.

Nowadays, another biometric generates interest and enthusiasm: gait authentication. Here, the user's motions, such as the leg's forward, backward or sideways

movements, are observed by sensors. The next sections introduce the general methodology for gait-authentication as well as neXenio's course of action in this field.

2.1 Methodology for gait-based authentication

Nowadays, gait-based authentication methods are well researched. Especially walking sequences, recorded by a smartphone or smart watches' inbuilt sensors, such as the accelerometer and gyroscope sensor, are reflected. Sprager and Juric [9] as well as Connor [2] summarized the state of the art methodologies used in this field. Guidelines for general data processing are stated by [1] and [3].

In brief, a walking sequence is cut in pieces and summarized by an average gait cycle. This cycle is used as a template and the dissimilarity of a new step is concluded by different distance metrics, such as Euclidean, Hamming, Manhattan or Tanimoto distance. Other approaches use statistical measurements in the time and frequency domain to gather descriptive information about walking sequences. Calculations such as mean, median, standard deviation, third and fourth order cumulants, skewness and kurtosis, and distribution parameters, such as a ten-bin histogram are used ([5, 7, 8]). Likewise, we found features such as the root mean squared, median absolute deviation, average absolute variation, quantiles, interquartile range, correlations and zero-crossing in several papers. In the frequency domain, Fourier and cepstral coefficients, discrete cousin transformations and wavelets are used [9]. Next to principal component analysis, Gait Dynamic Images [12] or geometric template matching were applied to gait sequences [4].

Besides cycle-based assignments, different machine learning techniques are used for authentication: linear and logistic regressions, k-nearest neighbors, support vector machines (SVM), hidden markov models and convolutional neuronal networks.

2.2 seamless.me @ neXenio

neXenio GmbH is a small company located in the middle of Berlin. They take up the challenge of developing high secure IT-solutions that address data sharing and virtual collaboration needs of digital workspaces. One of their products called SEAMLESSme targets the idea of behaviour-based authentication. The most important difference to other authentication systems and research approaches is that neXenio's implementation focuses on the premise of data privacy and security. Since data is processed locally on the device itself, data is never sent to external computation resources. Thus, the underlying classification approach is required to be a one-classification problem. An algorithm is deployed that only considers data from a single person. This training set is neither polluted by any outlier nor is enriched by data from other users. The algorithm must detect whether a new unknown walking observation fits to the learned pattern. In general, this type of classification shows less precise results than binary or multi-class classifications as the difference between users are not known. Only very few research studies considered this classification type, e.g. by using one-class SVMs.

Another challenge regarding local processing is the device's battery drain. Therefore, neXenio implemented a more efficient outlier technique instead of widely utilized battery-expensive algorithms. This outlier recognition algorithm is based on multi-level hierarchical nested histograms. Each histogram creates a discrete frequency distribution of the sensor data's processed features to a specific grain. Features, mentioned in the previous section 2.1, were evaluated and assembled in a way that the best authentication performances were attained.

3 The Problem of learning from a variety of gait patterns

Real-world gait-based authentication applications have to deal with the variety of natural gait forms. If a specific, but genuine form is unknown by the algorithm, it can hardly be assigned to the user. Gait-affecting factors are of physiological or environmental nature. While physiological factors induce more unconscious circumstances, such as permanent gait abnormalities or temporal changes caused by mood, environmental factors are represented by clothing, shoes, surfaces, slopes or obstacles [9]. Real world behavior-based authentication systems need to be robust for long-term utilization as these factors can change by varying degrees from time to time.

Public datasets for gait recognition do not imply a comprehensive overview about a user's possible walking situations. While OU-ISIR Biometric Database of Osaca University [6] is the largest database in the field of gait recognition, holding nearly 750 subjects, just one environmental factor, inclined terrain, was recorded in one session. Frank et al. [4] published another dataset in cooperation with the McGill University. This database contains just 20 participants, but data is recorded in the wild in two distinct sessions, meaning that people could have worn different clothes and shoes per session. Research that relies on this database, detected the effect of clothes the most. In all analyses the authentication performance suffers in cross-day comparison, particularly when people had a major change in trouser type ([4, 10]). This effect is also shown by the larger dataset of Subramanian et al. [11].

Purpose of Future SOC Lab utilization

In the last years, we collected a great amount of data files, covering this variety of physiological and environmental forms. At the moment, our statistical investigations as well as our novelty detector, which we use for user classification, can be just tested by dividing our dataset into smaller subsets. An evaluation that comprises all walking circumstances is not feasible as the limits of our machines according to RAM are reached.

By applying for utilization of the HPI Future SOC Lab we aimed to improve our authentication approach, refine signal preprocessing, feature engineering and modeling in regard to meet stability for wide user profiles.

4 Usage of Future SOC Lab resources

Again, Future SOC Lab provided us with an isolated server, allowing us to process all of our data in parallel. This was one of its main advantages as it greatly sped up our analyses. Even ad-hoc analyses, which evaluate the impact of e.g. an additional filter method, were possible.

Due to the developments from the last semesters, we still benefit from the whole refactored and optimized data transformation processes. Especially, cross validation and grid search methods made in possible to evaluate parameters. Both benefit from parallel computing, enabled by SOC Lab's core clusters.

5 Evaluation

5.1 Performance metrics

For classification objectives, generally, the true positives, false positives (also called false matches; imposters get authenticated), false negatives (called false non-matches; genuine sequences do not match) and true negatives are calculated for evaluating the performance of an algorithm. Biometric algorithms are ranked by the equal error rate (EER). This metric reflects the closest point of false matches and false non-matches. The lower the equal error rate value, the higher the accuracy of the biometric system.

Including the indoor positioning system, false matches/ false non matches are not just caused by a wrong biometric match, but also by a gate-system error. So, two error rates are complemented to rank one whole system.

5.2 Findings

This semester we developed an approach similar to "eigenfaces" and "eigensteps", combining principal components and support vector machines, but in the context of a one class classification problem. We successfully implemented cycle detection and "eigensteps" in our iOS and Android applications.

Additional, we added a indoor positioning part to our project structure. Here we were also challenged by parameter searching, especially for BLE antennas and smartphones.

We achieved an EER of 16% while keeping the phones' pocket position fixed. Using data of different pockets increases the error to 25%. This problem will be addressed in our future work.

6 Conclusion and future work

The resources of HPI's Future Soc Lab enabled us to intensify our data analysis and improve the authentication model. Only by using the advantage of the server's

parallel computing, could we process our data — and even speed up the total run time.

At the moment, we are implementing a coordinate transformation that converts a phone's orientation in a trouser pocket into a resilient and unbiased device and environment independent coordinate system. We expect to make a statement about whether there is an improvement in the result compared to the calculation of the acceleration's magnitude.

Furthermore, we want to refine our indoor positioning algorithm and want to compare it with k-means clustering and boosting algorithms.

In this regard, we look forward to using the resources of Future SOC Lab in the next semester again. We want to thank the Future SOC Lab team for the great support.

References

[1] A. Buriro, Z. Akhtar, B. Crispo, and S. Gupta. "Mobile Biometrics: Towards A Comprehensive Evaluation Methodology Mobile Biometrics : Towards A Comprehensive Evaluation Methodology". In: *2017 International Carnahan Conference on Security Technology (ICCST)*. 2017. ISBN: 978-1-5386-1585-0. DOI: 10.1109/CCST.2017.8167859.

[2] P. Connor and A. Ross. "Biometric recognition by gait: A survey of modalities and features". In: *Computer Vision and Image Understanding* 167 (2018), pages 1–27. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2018.01.007.

[3] R. Ferrero, F. Gandino, B. Montrucchio, M. Rebaudengo, A. Velasco, and I. Benkhelifa. "On gait recognition with smartphone accelerometer". In: *Proceedings - 2015 4th Mediterranean Conference on Embedded Computing, MECO 2015 - Including ECyPS 2015, BioEMIS 2015, BioICT 2015, MECO-Student Challenge 2015* (2015), pages 368–373. DOI: 10.1109/MECO.2015.7181946.

[4] J. Frank, S. Mannor, J. Pineau, and D. Precup. "Time Series Analysis Using Geometric Template Matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.3 (2013), page 1. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.121.

[5] T. Hoang, D. Choi, and T. Nguyen. "Gait authentication on mobile phone using biometric cryptosystem and fuzzy commitment scheme". In: *International Journal of Information Security* 14.6 (2015), pages 549–560. ISSN: 1615-5270. DOI: 10.1007/s10207-015-0273-1.

[6] T. T. Ngo, Y. Makihara, H. Nagahara, Y. Mukaigawa, and Y. Yagi. "The largest inertial sensor-based gait database and performance evaluation of gait-based personal authentication". In: *Pattern Recognition* 47.1 (2014), pages 228–237. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2013.06.028.

[7] C. Nickel. "Accelerometer-based Biometric Gait Recognition for Authentication on Smartphones". PhD thesis. Darmstadt: Technischen Universität Darmstadt, June 2012. URL: http://tuprints.ulb.tu-darmstadt.de/3014/.

[8] C. Nickel and C. Busch. "Classifying accelerometer data via hidden Markov models to authenticate people by the way they walk". In: *IEEE Aerospace and Electronic Systems Magazine* 28.10 (2013), pages 29–35. ISSN: 0885-8985. DOI: 10.1109/MAES.2013.6642829.

[9] S. Sprager and M. Juric. "Inertial Sensor-Based Gait Recognition: A Review". In: *Sensors* 15.12 (Sept. 2015), pages 22089–22127. ISSN: 1424-8220. DOI: 10.3390/s150922089.

[10] S. Sprager and M. B. Juric. "An Efficient HOS-Based Gait Authentication of Accelerometer Data". In: *IEEE Transactions on Information Forensics and Security* 10.7 (2015), pages 1486–1498. ISSN: 1556-6013. DOI: 10.1109/TIFS.2015.2415753.

[11] R. Subramanian, S. Sarkar, M. Labrador, K. Contino, C. Eggert, O. Javed, J. Zhu, and H. Cheng. "Orientation invariant gait matching algorithm based on the Kabsch alignment". In: *2015 IEEE International Conference on Identity, Security and Behavior Analysis, ISBA 2015* (2015), pages 1–8. DOI: 10.1109/ISBA.2015.7126347.

[12] Y. Zhong, Y. Deng, and G. Meltzner. "Pace independent mobile gait biometrics". In: *2015 IEEE 7th International Conference on Biometrics Theory, Applications and Systems, BTAS 2015* (2015), pages 1–8. DOI: 10.1109/BTAS.2015.7358784.

Benchmarking the TeraSort Algorithm on Ethernet Cluster

Benchmarking Apache Hadoop based implementation

Marek Nowicki

Department of Computer Science
Faculty of Mathematics and Computer Science
Nicolaus Copernicus University in Toruń, Poland
faramir@mat.umk.pl

The paper shows the evaluation of MapReduce TeraSort implementation on the 1000 Core Cluster at HPI Future SOC Lab. The implementation was done using the Apache Hadoop framework. This is the third report of usage of the Java on the Ethernet Cluster.

1 Introduction

The project called *Benchmarking Java on Ethernet Cluster* checks the scalability of parallel, network-intensive microbenchmarks and applications written in Java, using the PCJ library, on the cluster with high-performance Ethernet — on the 1000 Core Cluster — Ethernet-based cluster — at the Hasso Plattner Institute. The current paper describes the findings done during the second extension of the project in the period from spring 2020 to fall 2020, associated with the TeraSort application running on the cluster using the Apache Hadoop framework. The previous technical report papers [2, 3] presents the performance of some selected microbenchmarks and the sort implementations in the PCJ library.

The rest of the paper is organized as follows. Section 2 contains a brief introduction into Apache Hadoop and MapReduce processing, then section 3 describes the cluster setup, next section 4 presents the benchmark results and the paper is concluded by section 5.

2 Apache Hadoop

Apache Hadoop is a well-known framework for processing huge amount of data written in the Java language. The main data processing technique used in Apache Hadoop is the MapReduce.

Processing data using MapReduce is composed of five steps.

1. loading data from disk as key/value pairs;
2. mapping (transforming) the pair into zero (filtered out), one or many intermediate key/value pairs;

3. grouping all intermediate values with the same key, sorting and shuffling them to load balance data for the reducers;
4. reducing (combining) the data into zero (filtered out) or typically one output (reduced) value (it is also possible to return more output pairs for one key);
5. storing reduced data on the disk.

If the algorithm needs more map/reduce steps, the Apache Hadoop has to store intermediate results on the disk and then load it at the beginning of the next step.

TeraSort algorithm

The Apache Hadoop TeraSort package is a benchmark application that sorts large input data. The input data consists of many 100-byte long records. Each record has 10-byte long key and the rest is the value. The input data can be generated using a `teragen` application. The `terasort` application is the main benchmark application. The output of sort data can be validated using a `teravalidate` application to check if the data is properly sorted.

The TeraSort algorithm is a one-step algorithm. It starts with the sampling input data to generate the split points. Split points are used for partitioning the data between reducers to ensure that all elements in one reducer are less than each element in the subsequent reducers. The mapping and reducing functions are identity functions. In the end, the sorted data, i.e. the returned key/value pairs, is stored back on the disk. The directory with results contains multiple output files — one file per *reducer*.

Data is loaded and stored using the HDFS (Hadoop Distributed File System).

Full code of the benchmark is available at GitHub [1].

3 Cluster setup

This section presents results obtained on the HPI Future SOC Lab 1000 Core Cluster during Spring 2020 period.

Hardware and software

Hardware

The 1000 Core Cluster consists of 25 computational nodes. The computing nodes are equipped with the 4 Intel Xeon E7-4870 processors, with max. 2.40 GHz clock speed. Each processor contains 10 cores, each core can run 2 threads in hyper-threading mode. In total it is possible to run 80 threads per node. There is installed 1 TB of RAM on each node, that at least more than 800 GB should be available for the user. Nodes are connected using 10-Gigabit Ethernet using Intel 82599ES 10-Gigabit Ethernet Controller. The benchmark uses the `/tmp` directory to store the HDFS data. The

`/tmp` directory is exclusive for each node and is located on SSD drive attached to each node.

The benchmarks have used the maximum of 8 nodes for the computation as this is the standard limit on the submission system on the cluster.

Software
The operating system installed on the nodes was *Ubuntu* Linux in version 18.04.4 LTS (*Bionic Beaver*).

The *SLURM* (version 18.08.7) was used for submitting batch jobs to the cluster.

Java Virtual Machine from *Oracle JDK 13* package was used for the benchmarks of Java codes. Apache Hadoop in version 3.2.1 was used for running Hadoop cluster.

Apache Hadoop configuration Almost all default values of Apache Hadoop configuration were left unmodified. The configuration changes are presented in table 1.

SLURM scripts As the HPI cluster is a general-purpose cluster and uses the SLURM submission system, it was not possible to start the Hadoop cluster using the standard Apache Hadoop mechanism. This mechanism uses an *ssh* connection assuming there is a passwordless connection between nodes for the user, that is not true for the SLURM submission system configuration. To overcome the limitation, it was necessary to use the SLURM mechanism to start the required daemons on allocated nodes. The *namenode*, *secondarynamenode* and *resourcemanager* daemons were run in the background on the job master node. To run *datanode*s and *nodemanager*s daemons on all allocated nodes the `srun` application was used.

Listings 1, 2, and 3 present an *SBATCH* script for preparing, starting and shutting down the cluster.

Listing 1 set up the script environment. First lines ensure that whole 8 nodes will be exclusively allocated for the job, and no other *Hadoop-Cluster* job can run at the same time. Next, the user configuration follows. The path to the Java and Apache Hadoop installation directories, as well as the directory for storing Apache Hadoop data (*HADOOP_DATA*). There is also prepared directory with template configuration that will be copied and filled in later steps. Moreover, it is possible to clear the old Apache Hadoop data directory on the allocated nodes. The user configuration options and the template of Apache Hadoop configuration are all the user has to set up to start the cluster. Next, the normal execution occurs. First of all, there is a prepared job directory, where all the script and applications output files will be stored. Then the configuration is prepared from the template. During the next step, the *HADOOP_DATA* directory is (optionally) cleared and the output directories created on each node. The last step is preparing DFS directories if not yet exists.

Listing 2 starts all the necessary Apache Hadoop daemons. First of all, the *namenode*, *secondarynamenode* and *resourcemanager* daemons are started on the job master node. Then the `srun` is used for starting *datanode*s and *nodemanager*s services on all nodes. The job is run in the background. The script checks if all *datanode*s started up checking the number of *Live datanodes* reported by `dfsadmin` application. The

Table 1: Apache Hadoop configuration template. The text written using `TYPEWRITER` font means values changed during cluster setup.

yarn-site.xml	
yarn.resourcemanager.hostname	`MASTER_NODE`
yarn.nodemanager.aux-services	mapreduce_shuffle
yarn.nodemanager.resource.memory-mb	821 600 MB
yarn.scheduler.maximum-allocation-mb	821 600 MB
yarn.scheduler.minimum-allocation-mb	128 MB
yarn.nodemanager.vmem-check-enabled	false
yarn.nodemanager.resource.cpu-vcores	80
yarn.scheduler.maximum-allocation-vcores	80
yarn.nodemanager.disk-health-checker-.max-disk-utilization-per-disk-percentage	98.5
mapred-site.xml	
mapreduce.framework.name	yarn
mapreduce.map.resource.memory-mb	40 000 MB
mapreduce.reduce.resource.memory-mb	40 000 MB
yarn.app.mapreduce.am.resource.memory-mb	128 000 MB
core-site.xml	
fs.defaultFS	`hdfs://MASTER_NODE:9000`
hdfs-site.xml	
dfs.replication	1
dfs.namenode.name.dir	`NAMENODE_PATH`
dfs.datanode.data.dir	`DATANODE_PATH`

Listing 1: Apache Hadoop cluster—SLURM `sbatch` script. Part 1: Setup.

```
#!/bin/bash -l
#SBATCH --job-name=Hadoop-Cluster
#SBATCH --dependency=singleton
#SBATCH --nodes=8
#SBATCH --ntasks-per-node=80
#SBATCH --exclusive

# User configuration
JAVA_HOME=/home/marek.nowicki/jdk-13
HADOOP_HOME=/home/marek.nowicki/hadoop-3.2.1
HADOOP_DATA=/tmp/marek.nowicki/hadoop-data
CONFIG_TEMPLATE="/home/marek.nowicki/terasort-test/hadoop-conf"
#CLEAR_DATA=1 # uncomment if should clear HADOOP_DATA dir

# Other script variables
JOB_ID=${SLURM_JOB_ID:-$(date +"%Y%m%d_%H%M%S")}
JOB_NUM_NODES=${SLURM_JOB_NUM_NODES:-1}
PATH="${JAVA_HOME}/bin:$PATH"
MASTER_NODE=$(hostname -s)
NAMENODE_PATH="${HADOOP_DATA}/namenode"
DATANODE_PATH="${HADOOP_DATA}/datanode"

# Directory for job files
mkdir job-${JOB_ID} && cd job-${JOB_ID} || exit 1

export HADOOP_LOG_DIR="${HADOOP_DATA}/logs"
export HADOOP_CONF_DIR="$(pwd)/hadoop-conf"
HADOOP_LOOP="$(pwd)/HADOOP_LOOP"

# Preparing configuration from template for a job
cp -r ${CONFIG_TEMPLATE} ${HADOOP_CONF_DIR}

sed -i "s#MASTER_NODE#${MASTER_NODE}#g" ${HADOOP_CONF_DIR}/*.xml
sed -i "s#NAMENODE_PATH#${NAMENODE_PATH}#g" ${HADOOP_CONF_DIR}/*.xml
sed -i "s#DATANODE_PATH#${DATANODE_PATH}#g" ${HADOOP_CONF_DIR}/*.xml

# Clearing HADOOP_DATA directory on all allocated nodes
srun -N ${JOB_NUM_NODES} -n ${JOB_NUM_NODES} \
  bash -c "exec > >(sed 's/^/\$(hostname): /');
         exec 2> >(sed 's/^/\$(hostname) (stderr): /' >&2);
         [ ! -z ${CLEAR_DATA+x} ] && rm -rf ${HADOOP_DATA};
         mkdir -p ${HADOOP_DATA}/logs"

# Preparing DFS directories if not exists
if [ ! -d ${NAMENODE_PATH} -o ! -d ${DATANODE_PATH} ]
then
  ${HADOOP_HOME}/bin/hdfs namenode -format
fi
```

Listing 2: Apache Hadoop cluster—SLURM `sbatch` script. Part 2: Launching and monitoring.

```
# Starting namenode, secondarynamenode and resourcemanager daemons
${HADOOP_HOME}/bin/hdfs --config "${HADOOP_CONF_DIR}" \
   --daemon start namenode

${HADOOP_HOME}/bin/hdfs --config "${HADOOP_CONF_DIR}" \
   --daemon start secondarynamenode

${HADOOP_HOME}/bin/yarn --config "${HADOOP_CONF_DIR}" \
   --daemon start resourcemanager

# Starting datanodes and nodemanagers on all allocated nodes
srun -N ${JOB_NUM_NODES} -n ${JOB_NUM_NODES} \
  -o worker-%N.out -e worker-%N.err \
  bash -xc "\
    ulimit -S -n unlimited ; \
    ${HADOOP_HOME}/bin/hdfs --config \"${HADOOP_CONF_DIR}\" \
       --daemon start datanode ; \
    ${HADOOP_HOME}/bin/yarn --config \"${HADOOP_CONF_DIR}\" \
       nodemanager" &
workersPID=$!

# Waiting at most 5 minutes for applications to launch
for i in `seq 1 60`; do
  sleep 5
  [ "$(${HADOOP_HOME}/bin/hdfs dfsadmin -report \
      | grep -Po '(?<=Live datanodes \()(\d+)')" \
            == "${JOB_NUM_NODES}" ] && break
done

# Monitoring loop with a guard
echo 1 > ${HADOOP_LOOP}
i=0
# Checking if cluster should stop (every 1 second)
while [ "$(cat ${HADOOP_LOOP} 2>/dev/null )" == "1" ]; do
  i=$(( $i + 1 ))
  # Checking datanodes status (every 10 minutes)
  if [ $(( $i % 600 )) -eq 0 ]; then
    dfsReport=$(${HADOOP_HOME}/bin/hdfs dfsadmin -report)
    liveNodes=$(grep -Po '(?<=Live datanodes \()(\d+)' <<< "$dfsReport")

    if [ "$liveNodes" != "${JOB_NUM_NODES}" ]; then
      echo "Not all datanodes are alive: $liveNodes:"
      awk '/Dead datanodes/{dead=1};
          dead && /^Name: /{print}' <<< "$dfsReport"
    fi
  fi
  # Printing report about DFS (every 1 hour)
  if [ $(( $i % 3600 )) -eq 0 ]; then
    ${HADOOP_HOME}/bin/hdfs dfsadmin -report
  fi
  sleep 1
done
```

SLURM job finishes when it comes to the end of script processing. To prevent it, the loop with a guard is executed. The loop checks for *HADOOP_LOOP* file content. If the file does not contain single *1* or even does not exist, the loop finishes. Additionally, the loop checks the DFS state and report of its usage and the number of live *datanode*s.

Listing 3: Apache Hadoop cluster — SLURM `sbatch` script. Part 3: Stopping and finalizing.

```
101 # Stopping datanodes and nodemanagers
102 kill $workersPID
103
104 # Stopping resourcemanager, secondarynamenode and namenode daemons
105 ${HADOOP_HOME}/bin/yarn --config "${HADOOP_CONF_DIR}" \
106    --daemon stop resourcemanager
107
108 ${HADOOP_HOME}/bin/hdfs --config "${HADOOP_CONF_DIR}" \
109    --daemon stop secondarynamenode
110
111 ${HADOOP_HOME}/bin/hdfs --config "${HADOOP_CONF_DIR}" \
112    --daemon stop namenode
113
114 # Removing HADOOP data from ${HADOOP_DATA} directories
115 # ...only when HADOOP_LOOP file was also removed
116 [ ! -f ${HADOOP_LOOP} ] && \
117   srun -N ${JOB_NUM_NODES} -n ${JOB_NUM_NODES} \
118    bash -c "exec > >(sed 's/^/\$(hostname): /');
119           exec 2> >(sed 's/^/\$(hostname) (stderr): /' >&2);
120           rm -rf ${HADOOP_DATA}"
121
122 # EOF
```

Listing 3 stops all the Apache Hadoop services. First of all the `srun` background job is killed, so the SLURM also send kill signal to the children processes on all nodes and the *nodemanager*s and *datanode*s are stopped. Next, the *resourcemanager*, *secondarynamenode* and *namenode* daemons are stopped on the job master node. The last step is clearing the *HADOOP_DATA* directory, but only when the *HADOOP_LOOP* file does not exists.

The SLURM submission system was used to ensure that no Hadoop job will overlap with any other Hadoop job. Listing 4 presents the sample script for running Hadoop job from SLURM job. The configuration uses the *Hadoop-Cluster* job configuration. However, `hadoop jar` and `hdfs dfs` commands contain also parameters for pointing *resourcemanager* (`-jt ${MASTER_NODE}:8032`, where 8032 is the default *resourcemanager* port) and default file system (`-fs hdfs://${MASTER_NODE}:9000`) and those can also be used.

Listing 4: Submitting job to the Apache Hadoop cluster

```
#!/bin/bash -l
#SBATCH --job-name=Hadoop-Job
#SBATCH --dependency=singleton
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1

# User configuration
JAVA_HOME=/home/marek.nowicki/jdk-13
HADOOP_HOME=/home/marek.nowicki/hadoop-3.2.1
DATA_DIR=/home/marek.nowicki/hadoop-data
BENCHMARK_NAME="TeraSort_1e9"
NODES=8
THREADS=80

# Other script variables
JOB_ID=${SLURM_JOB_ID:-$(date +"%Y%m%d_%H%M%S")}
JOB_NUM_NODES=${SLURM_JOB_NUM_NODES:-1}
PATH="${JAVA_HOME}/bin:$PATH"

# Directory for job files
mkdir job-${JOB_ID} && cd job-${JOB_ID} || exit 1

MASTER_JOBID=$(\squeue -o "%i" --name Hadoop-Cluster | grep -v JOBID)
MASTER_NODE=$(scontrol show job ${MASTER_JOBID} \
            | grep -Po '(?<=BatchHost=)(.*)')
export HADOOP_CONF_DIR="$(realpath ..)/job-${MASTER_JOBID}/hadoop-conf"

echo "Master: ${MASTER_NODE} (job:${MASTER_JOBID})"

# directories on HDFS
INPUT_DIR="${DATA_DIR}/input/${BENCHMARK_NAME}"
OUTPUT_DIR="${DATA_DIR}/output/${BENCHMARK_NAME}/${JOB_ID}"

# Starting benchmark
${HADOOP_HOME}/bin/hadoop jar \
  ${HADOOP_HOME}/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar \
  terasort \
  -Dmapreduce.job.maps=$(( $NODES * $THREADS )) \
  -Dmapreduce.job.reduces=$(( $NODES * $THREADS )) \
  -Dmapreduce.job.running.map.limit=$(( $NODES * $THREADS )) \
  -Dmapreduce.job.running.reduce.limit=$(( $NODES * $THREADS )) \
  "${INPUT_DIR}" \
  "${OUTPUT_DIR}"

# Removing output files
${HADOOP_HOME}/bin/hdfs dfs -rm -r "${OUTPUT_DIR}"
# EOF
```

4 Results

The results presented here show the total execution time of the benchmark as reported by the applications—e.g. the time elapsed between *terasort.TeraSort: starting* and *terasort.TeraSort: done* messages. The result is an average taken from at least 5 runs.

The Apache Hadoop example package contains not only TeraSort application but also a *trivial Sort* application (`org.apache.hadoop.examples.Sort`). To use it with the data generated using `teragen` application it is necessary to provide input format, output format, key output and value output classes. Providing format classes from the *TeraSort* package produces proper output. However, removing the partitioning code from `TeraInputFormat`, just the code for storing record, resulted in producing the wrong output sequence—the validation failed.

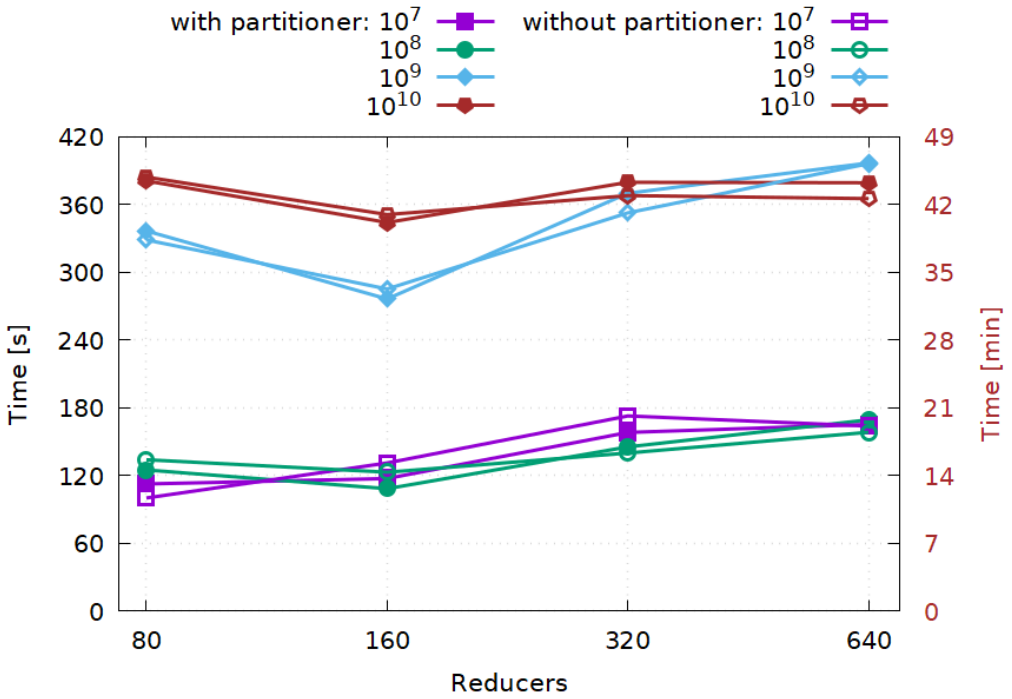


Figure 1: Strong scaling of the *trivial Sort* application. Execution on 80, 160, 320 and 640 reducers. The axis for 10^{10} plot is on the right.

Figure 1 present total time needed to sort and store 10^7, 10^8, 10^9 and 10^{10} records respectively using the *trivial Sort* application. The performance is roughly the same, regardless of the usage of the partitioner or not.

Figure 2 present total time needed to sort and store 10^7, 10^8, 10^9 and 10^{10} records respectively using the *TeraSort* application. However, comparing results with the PCJ results from [3], the results here were much worse, even though the PCJ results were not using HDFS, but SSD drive directly. That led to an idea to change the maximum

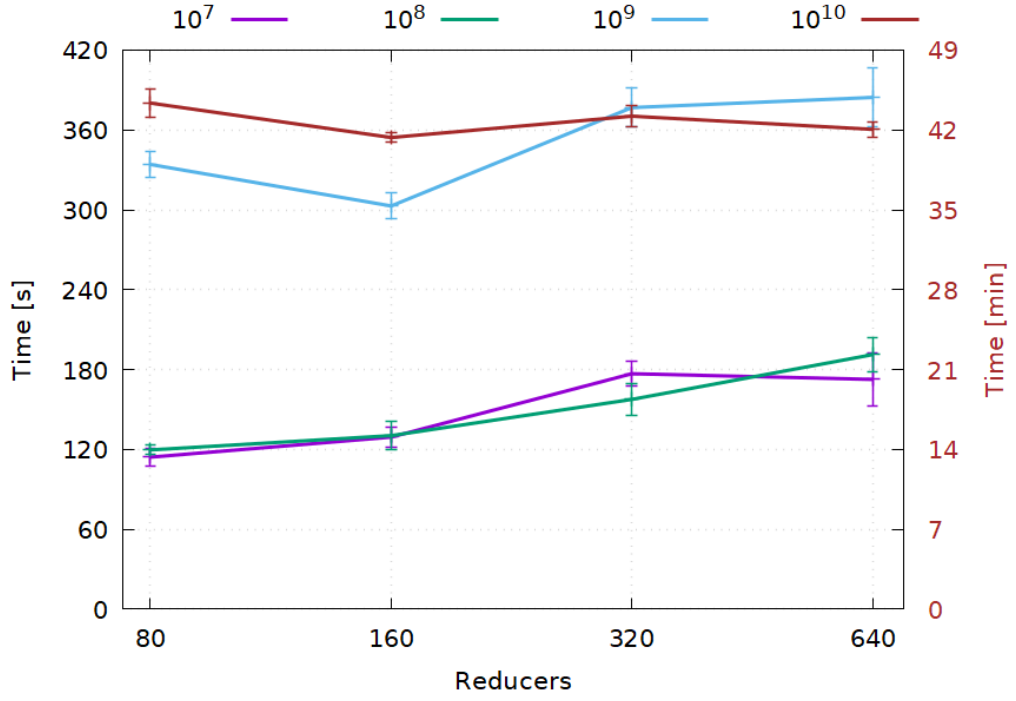


Figure 2: Strong scaling of the *TeraSort* application. Execution on 80, 160, 320 and 640 reducers. The axis for 10^{10} plot is on the right.

memory for a map and reduce tasks for Apache Hadoop. As the node has about 800 GB of RAM, and 80 reduce tasks, 40 GB of maximum memory usage per task could be too big oversubscription.

Figure 3 presents the performance obtained when running TeraSort application using 640 reducers, with the maximum memory of map and reduce tasks set to 5 GB, 10 GB, 20 GB and 40 GB. Due to `OutOfMemoryError` there is no point for 5 GB and 10^{10} records.

The smallest execution time is for the 10 GB maximum memory usage per task. The performance is similar to the previously obtained PCJ results.

5 Conclusion and future steps

The project shows that it is possible to run Apache Hadoop cluster even on the general-purpose Ethernet cluster. It is necessary to perform benchmarks to set up the cluster properly. One of the easiest parameters to tune is the maximum memory used for a map and reduce tasks. The properly set up cluster shows very good performance.

In the future steps, the detailed performance comparison with the PCJ reading from and writing to the HDFS is planned. Moreover, the maximum memory used for a map and reduce tasks is not the only configuration option that can be changed.

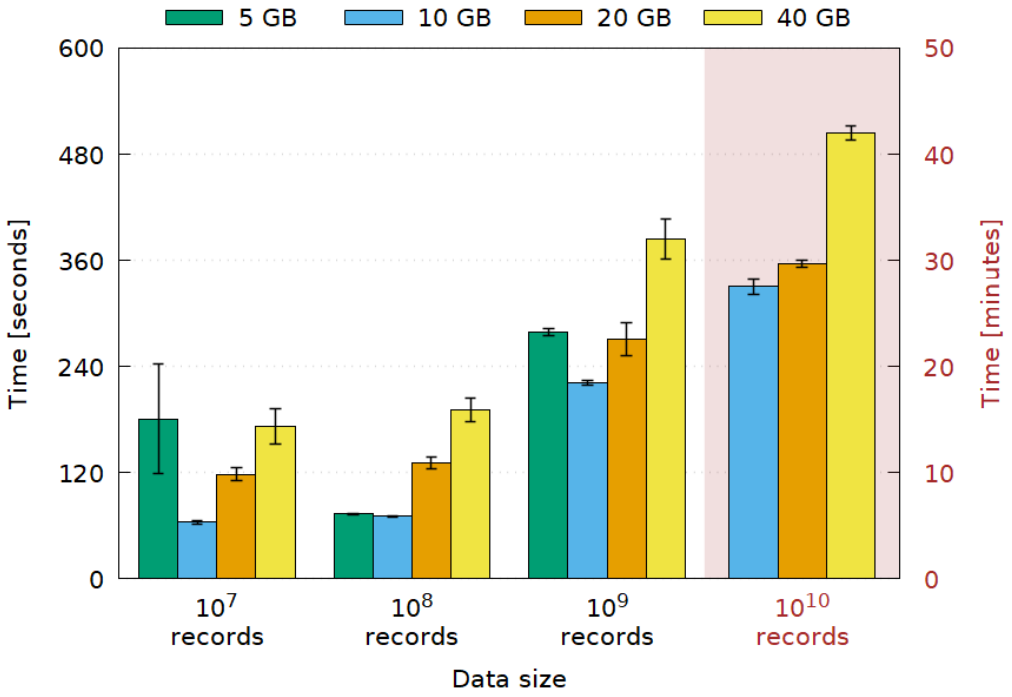


Figure 3: Performance of the *TeraSort* application for different values of maximum memory usage per task: 5 GB, 10 GB, 20 GB and 40 GB. Execution on 640 reducers. The axis for 10^{10} plot is on the right.

Tuning all Apache Hadoop properties might be necessary to get even better performance.

Acknowledgements

The author would like to thank Future SOC Lab, Hasso Plattner Institute for Digital Engineering for awarding access to the 1000 Core Cluster.

References

[1] *Apache Hadoop TeraSort at GitHub*. Accessed: 28.10.2020. URL: https://github.com/apache/hadoop/tree/rel/release-3.2.1/hadoop-mapreduce-project/hadoop-mapreduce-examples/src/main/java/org/apache/hadoop/examples/terasort (last accessed 2020-01-01).

[2] M. Nowicki. "Benchmarking Java on Ethernet Cluster". In: *HPI Technical Report*. 2021. in press.

[3] M. Nowicki. "Benchmarking the Sort Algorithm on Ethernet Cluster". In: *HPI Technical Report*. 2021.

Measurement-Based Software Performance Engineering for Microservices and Multi-Core Systems

Project Report for HPI Future SOC Period Spring 2020

André van Hoorn[1], Markus Frank[1], Sandro Speth[1], Andrea Janes[2], and Matteo Camilli[2]

[1] University of Stuttgart, Germany
[2] Free University of Bozen-Bolzano, Italy

This report provides a summary of our project "Measurement-Based Software Performance Engineering for Microservices and Multi-Core Systems" conducted during the HPI Future SOC Lab period spring 2020.

1 Project Idea

Our project was initially divided into two subprojects, namely (1) "DevOps-oriented Load Testing for Microservices" and (2) "Software Performance Engineering for Multi-Core Systems". Both subprojects are a direct continuation of the works that we started in the previous periods. During the period, we have added a third subproject, namely (3) "Automated Cross-Component Issue Classification".

In the remainder of this report, we will provide some more details about the project context (sections 1.1 and 1.2), list the granted Future SOC Lab resources (section 2), and provide a brief description of our activities and findings (section 3).

1.1 DevOps-oriented Load Testing for Microservices

Modern software engineering paradigms and technologies — such as DevOps [3] (including automation as part of continuous delivery) and microservices [16] — are gaining more and more attraction in the software and services engineering communities. Of particular interest are quality-of-service concerns, for instance, w. r. t. performance and reliability. While established approaches for classic contexts (i.e., which do not use DevOps and microservices) exist, their adoption to DevOps and microservices requires considerable research efforts [4, 5, 6, 15].

Load testing is a measurement-based approach to assess performance-related properties of software systems. In this project, we focus on the experimental evaluation of DevOps-oriented approaches for load testing microservices.

In the past periods, we have already conducted experiments for our newly developed approaches in the HPI infrastructure (e.g., [1, 2, 17]). The activities on load testing conducted during this period were a direct continuation of the activities started during the previous periods.

1.2 Software Performance Engineering for Multi-Core Systems

Multicore systems are a permanent part of our daily life. Regardless of whether we consider nowaday's desktop PCs, notebooks, or smart phones—all devices are running on multicore CPUs. To use these hardware features in an efficient way, developers need to build parallel-enabled software. However, the development of such software is more complex than developing sequential software.

To handle the rising complexity, it is necessary to develop software in an engineering-like way. In such a process, software architects plan and analyze software designs on a model level. Software architects can use tools like Palladio to simulate and analyze early-phase software designs. Unfortunately, current approaches and tools lack the ability to consider multicore systems. Therefore, in this subproject, we aim to find performance prediction methods for multicore systems in the context of our ongoing research [7, 9, 10, 11, 13, 14].

In the past periods, we intensively used the HPI infrastructure to perform experiment-based performance evaluations for parallel applications [7, 14]. We used the resulting measurements to extract performance curves. These performance curves can be used by software architects to increase the performance predictions for parallel applications run on multicore environments [12].

1.3 Automated Cross-Component Issue Classification

Microservice architectures consist of many independently developed services, which offer their functionality via externally defined interfaces and use other services via their interfaces. Although each service is developed independently, dependencies arise via the interfaces. Thus, issues in a service may not only be dependent on issues of the same service but also issues of other services. The goal of this project is to develop an approach that uses machine learning to identify and predict relationships between issues in the same service or across the boundaries of a microservice. Such relationship prediction should make it easier for the developer to identify the origin of an issue and to fix it. However, large amounts of RAM and cores are required for the machine learning methods used to predict the relationships.

2 Used Future SOC Lab resources

We requested and received dedicated (root) access to the following computing resources (servers):

1. 896 GB RAM, 80 cores
2. 32 GB RAM, 24 cores

Dedicated access has been given to us due to our expected high resource demands.

3 Findings

In this section, we will provide a summary of the experiments and results.

3.1 DevOps-oriented Load Testing for Microservices

To support the ongoing development of our performance testing tool PPTAM [1],[3] we tested it using the Trainticket application,[4] which is a benchmark microservice system comprised of 41 microservices. To test this system, one needs 24 GB of memory and 50 GB disk in Linux,[5] which makes it hard to test on single computers due to the lack of resources there. In the past period, we have initially setup the test infrastructure. Further experimentation within the HPI infrastructure will help us to better understand the feasibility of the developed PPTAM approach.

3.2 Software Performance Engineering for Multi-Core Systems

Based on the process proposed by Wert et al. [20], we continued the controlled experiments started in the previous period [8] to measure the behavior of performance-influencing factors of highly parallel software on multi-core architectures. Therefore, we executed various resource demands raging from processor-intensive to I/O-intensive on different hardware configurations. We used four different parallelization paradigms and measured memory bandwidth, cache behavior, and speedup. We used the measurements to extract performance curves using linear regression (see table 1).

Table 1: Extracted Performance Curves for Dedicated Machines. Based on the Speedup Behaviour of the Demands

	$f(x)$ for **Stage**		
Demand Type	**1**	**2**	**3**
CountNumbers	$0.473x$	$-0.217x + 0.653$	$-0.00100x + 0.213$
MatrixMultiplication	$0.387x$	$0.066x + 0.347$	$-0.00700x + 0.432$
FibonacciNumbers	$0.416x$	$0.027x + 0.416$	$0.00003x + 0.435$
PrimeNumbers	$0.445x$	$0.329x + 0.109$	$0.00080x + 0.548$
SortArray	$0.379x$	$0.131x + 0.250$	$-0.01800x + 0.537$
MandelSet	$0.455x$	$0.405x$	$0.00080x + 0.849$

[3] https://github.com/pptam/pptam-tool (last accessed 2020-01-01).
[4] https://github.com/FudanSELab/train-ticket (last accessed 2020-01-01).
[5] https://github.com/FudanSELab/train-ticket/wiki/Installation-Guide (last accessed 2020-01-01).

Table 2: Shows the Accuracy Gain (in %) of the Performance Curve Approach in Comparison to the Pure Palladio Approach

	Benchmark												
Threads	imagick	botsalgn	smithwa	nab	bt311	fma3d	swim	bwaves	kdtree	md	botsspar	applu311	Mean
10	−2.29	−10.51	−7.48	15.78	3.70	11.90	13.19	9.37	16.66	1.92	11.67	11.67	6.30
40	8.48	−3.20	−0.18	13.95	11.61	6.36	4.57	6.16	12.05	1.38	10.92	6.53	6.55
80	51.21	39.41	7.25	52.24	46.37	22.23	17.58	27.20	54.49	26.90	−25.57	72.33	32.64
90	47.17	41.52	8.02	55.43	46.45	23.99	18.96	30.03	37.94	31.00	−32.05	61.72	30.85
Mean	26.14	16.81	1.90	34.35	27.03	16.12	13.58	18.19	30.28	15.30	−8.76	38.06	19.08

Further, we evaluated the performance prediction models—using our performance curves vs. not using the performance curves. As example use cases, we executed SPEC Benchmarks on the HPI infrastructure. As conclusion we can state that using the performance curves increases the prediction accuracy of model-based performance predictions by up to 60% [12] (see table 2).

3.3 Automated Cross-Component Issue Classification

Issues have been a fundamental part of software development for years. There are different classes of issues, such as bug reports or feature requests, but also more fine-grained classifications. In reality, however, it is often the case that many issues are not classified or are even classified falsely in the issue trackers. It can happen that the developers easily overlook an existing bug report, and a developer does not fix the bug for a long time. An incorrect issue classification can therefore lead to more difficult bug fixing and thus increased maintenance costs. Especially in microservice architectures, where services meet via externally defined interfaces, errors in one service can lead to faults in the service that depends on it. Issues can, therefore propagate across interfaces along the call chain. When troubleshooting in microservice architectures, it is, therefore, essential to explicitly record when an issue affects the interface of a service.

We have developed an automated issue classifier to increase the quality of issue management. The issue classifier uses many different classification algorithms, weights the results of all algorithms, and uses bagging to assemble the results to achieve an accuracy of almost 100% for the trained issue classes and labels. However, to train these classification algorithms, models must be created from several 1000 meticulously selected issues. As much data as possible must be loaded into RAM at the same time to ensure sufficient training times during development. With the help of such issue classifier, conventional issues as well as cross-component issues, as those presented in [18, 19], can be classified efficiently and correctly.

Acknowledgment

We are thankful to the HPI Future SOC Lab for having granted us access to the computing infrastructure. The environment eases the joint work of different organizations on the platform, which has so far been hindered by university-internal access constraints — apart from the fact that an equipment comparable to that of the HPI Future SOC Lab has not been available to us, and that enables extensive performance configuration tests needed to reach our goals.

References

[1] A. Avritzer, V. Ferme, A. Janes, B. Russo, A. van Hoorn, H. Schulz, D. Menasché, and V. Rufino. "Scalability Assessment of Microservice Architecture Deployment Configurations: A Domain-based Approach Leveraging Operational Profiles and Load Tests". In: *Journal of Systems and Software* 165 (2020).

[2] A. Avritzer, V. Ferme, A. Janes, B. Russo, H. Schulz, and A. van Hoorn. "A quantitative approach for the assessment of microservice architecture deployment alternatives using automated performance testing". In: *Proceedings of the 12th European Conference on Software Architecture (ECSA 2018)*. LNCS. Springer, 2018.

[3] L. J. Bass, I. M. Weber, and L. Zhu. *DevOps : A Software Architect's Perspective*. SEI series in software engineering. Addison-Wesley, 2015. ISBN: 978-0-13-404984-7.

[4] C. Bezemer, S. Eismann, V. Ferme, J. Grohmann, R. Heinrich, P. Jamshidi, W. Shang, A. van Hoorn, M. Villaviencio, J. Walter, and F. Willnecker. "How is Performance Addressed in DevOps? A Survey on Industrial Practices". In: *Proceedings of the ACM/SPEC International Conference on Performance Engineering (ICPE 2019)*. ACM, 2019, pages 45–50.

[5] A. Brunnert, A. van Hoorn, F. Willnecker, A. Danciu, W. Hasselbring, C. Heger, N. Herbst, P. Jamshidi, R. Jung, J. von Kistowski, A. Koziolek, J. Kroß, S. Spinner, C. Vögele, J. Walter, and A. Wert. *Performance-oriented DevOps: A Research Agenda*. Technical report SPEC-RG-2015-01. SPEC Research Group — DevOps Performance Working Group, Standard Performance Evaluation Corporation (SPEC), 2015.

[6] S. Eismann, C.-P. Bezemer, W. Shang, A. van Hoorn, and D. Okanović. "Microservices: A Performance Tester's Dream or Nightmare?" In: *Proceedings of the 11th ACM/SPEC International Conference on Performance Engineering (ICPE 2020)*. ACM, 2020.

[7] M. Frank, S. Becker, A. Kaplan, and A. Koziolek. "Performance-influencing Factors for Parallel and Algorithmic Problems in Multicore Environments".

In: *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering* (*ICPE '19*).

[8] M. Frank, S. Becker, A. Kaplan, and A. Koziolek. "Performance-Influencing Factors for Parallel and Algorithmic Problems in Multicore Environments: Work-In-Progress Paper". In: *Companion of the 2019 ACM/SPEC International Conference on Performance Engineering* (*ICPE 2019*). ACM, 2019, pages 21–24. DOI: 10.1145/3302541.3313099.

[9] M. Frank and M. Hilbrich. "Performance Prediction for Multicore Environments:An Experiment Report". In: *Proceedings of the Symposium on Software Performance* (*SSP 2016*). 2016.

[10] M. Frank, M. Hilbrich, S. Lehrig, and S. Becker. "Parallelization, Modeling, and Performance Prediction in the Multi-/Many Core Area: A Systematic Literature Review". In: *Proceedings of the 7th IEEE International Symposium on Cloud and Service Computing* (*SC2 2017*). 2017.

[11] M. Frank, F. Klinaku, and S. Becker. "Challenges in Multicore Performance Predictions". In: *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering* (*ICPE 2018*). ACM, 2018. DOI: 10.1145/3185768.3185773.

[12] M. Frank, L. Schmid, A. Kaplan, L. Greiner, A. Koziolek, and S. Becker. "Performance Curves for better Performance Predictions of Parallel Applications in Multicore Environments". In: *Companion of the 2020 ACM/SPEC International Conference on Performance Engineering*. ACM. 2020.

[13] M. Frank, S. Staude, and M. Hilbrich. "Is the PCM Ready for ACTORs and Multicore CPUs? — A Use Case-based Evaluation". In: *Proceedings of the 8th Symposium on Software Performance* (*SSP 2017*). 2017.

[14] P. Gruber and M. Frank. "Modelling and Predicting Memory Behavior in Parallel Systems with Network Links: Palladio-based Experiment Report". In: *Proceedings of the 10th Symposium on Software Performance* (*SSP 2019*). 2019.

[15] R. Heinrich, A. van Hoorn, H. Knoche, F. Li, L. E. Lwakatare, C. Pahl, S. Schulte, and J. Wettinger. "Performance Engineering for Microservices: Research Challenges and Directions". In: *Companion of the 8th ACM/SPEC International Conference on Performance Engineering* (*ICPE 2017*). ACM, 2017. ISBN: 978-1-4503-4899-7. DOI: 10.1145/3053600.3053653.

[16] S. Newman. *Building Microservices*. O'Reilly Media, Inc., 2015.

[17] H. Schulz, T. Angerstein, D. Okanović, and A. van Hoorn. "Microservice-tailored Generation of Session-based Workload Models for Representative Load Testing". In: *Proceedings of the 27th IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems* (*MASCOTS 2019*). To appear. 2019.

[18] S. Speth. "Issue management for multi-project, multi-team microservice architectures". Master's thesis. 2019.

[19] S. Speth, U. Breitenbücher, and S. Becker. "Gropius: A Tool for Managing Cross-component Issues". In: *European Conference on Software Architecture*. Springer. 2020, pages 82–94.

[20] A. Wert, J. Happe, and D. Westermann. "Integrating Software Performance Curves with the Palladio Component Model". In: *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering* (*ICPE 2012*). ACM, 2012, pages 283–286.

Online Speech Recognition for German Medical Speech

Our path to data collection

Erik Ziegler, Marcel Schmidberger, and Mats Pörschke

Hasso Plattner Institute for Digital Engineering
erik.ziegler@student.hpi.de
marcel.schmidberger@student.hpi.de
mats.poerschke@student.hpi.de

Medical documentation is a huge burden across all healthcare professions. In many cases, speech recognition could provide great benefits but even modern speech processing models struggle with medical terms. By creating a medical data set and training a German model we aim to improve that. Furthermore, we want to evaluate the performance of the three major open-source speech recognition frameworks Kaldi, DeepSpeech, and the wav2letter++ speech processing toolkit, for our use case. Unfortunately, we did not manage to create a sufficient speech data set as of now but finally cleared the path to do so. Therefore, we cannot evaluate the results of our models here but will discuss our developments and considerations as well as our next steps.

1 Introduction

Speech processing has been established in many consumer applications and although most known speech recognition engines are proprietary, there are toolkits and models freely available, e.g. Kaldi, DeepSpeech, and wav2letter++. Most of these open-source toolkits already produce models that reach a Word Error Rate (WER) below 10 % for English on various test sets but they lack pre-trained models for other languages. This is primarily because there is significantly more annotated speech data openly available for the English language (>10 000 hours). Although by a larger margin, German is the language with the second most speech data (≈1000 hours) after English. But even models trained on a large amount of English data available are not suitable for medical applications, because the underlying data in both the acoustic and language model lacks utterances of medical terms.

The speech recognition toolkit wav2letter++ enables high accuracy with relatively little training data: an existing German acoustic wav2letter++ model based on [3] by Zamia achieved a WER of <4 % with only 412 hours of annotated speech data [17]. We aim to add speech data including utterances of medical terms to the training set and train a wav2letter++ *streaming convnet* [15] model and evaluate its accuracy and compare it to DeepSpeech and Kaldi models, trained on our dataset.

2 Speech recognition frameworks

2.1 DeepSpeech

DeepSpeech is an open-source Speech-To-Text engine firstly introduced in Baidu's Deep Speech research paper [7]. Based on that, the Mozilla Foundation provides an implementation using Google's TensorFlow machine learning framework internally [5]. The speech recognition pipeline uses extracted audio features as inputs for a single-directional Recurrent Neural Network (RNN) consisting of a multi-layer Long Short-Term Memory (LSTM) [9] cell. Each application of the RNN cell outputs the recognized character within a given short audio snippet. An overlapping windowing approach ensures that all characters in the provided audio are extracted. Using a language model, DeepSpeech finally ranks potential words based on the recognized character sequence and outputs the most likely overall transcription. All models support online as well as offline audio transcriptions.

With every official release, the Mozilla Foundation provides pre-trained English model versions. Moreover, there are DeepSpeech models available in a variety of other languages, actively developed by the open-source community. The German DeepSpeech models [1, 6] were trained using an open-source German Speech Corpus released by the University of Hamburg [13].

2.2 Kaldi

Kaldi [11] is an open-source speech recognition toolkit actively developed by its community. It provides pre-trained models for different languages trained on a variety of corpora. In contrast to DeepSpeech or Wav2letter++, Kaldi does not implement an end to end speech recognition pipeline using a deep neural network. Instead, Kaldi's model can be divided into two main components: The first part is an acoustic model deep neural network [14] that transcribes the audio features into a sequence of context-dependent phonemes. The second part is the decoding graph. It takes the previously generated phonemes and turns them into lattices. Lattices are lists of word-sequences that are likely for a particular audio part. Using a language model, the most likely sentence is then determined as the final transcription. Similar to DeepSpeech, Kaldi also supports online transcriptions as well as the transcription of prerecorded files.

2.3 Wav2letter++

The speech processing toolkit wav2letter++ is developed by the Speech team at Facebook AI Research and was open-sourced in 2019. It was built to facilitate research in end-to-end models for speech recognition. As of today, it incorporates recipes for approaches of 7 speech recognition papers ([3, 8, 10, 12, 15, 16, 18]).

Transcribing speech in real-time from an audio stream is known as online speech recognition. Most speech recognition research focuses on offline speech recognition (transcribing audio files) without the constraint of performing the task in real-time.

In January 2020, Facebook AI released and open-sourced the wav2letter@anywhere inference platform. As wav2letter itself, it aims to allow running inference for various types of speech recognition models. But as of now, the inference platform only supports models build with time-depth separable (TDS) convolutions and connectionist temporal classification (CTC) criterion.

The mentioned German wav2letter++ model that was provided by Zamia [2] is not compatible with the inference platform. This is why we want to train a compatible model, more specifically a *streaming convnet* [15] model that is built with TDS convolutions and uses the CTC criterion. Such a model would also be well suited for offline use for example in mobile applications due to its low latency and small size.

3 Training data for medical speech processing

Creating a speech data set for medical documentation poses an especially difficult data protection challenge. This is due to the nature of speech data, as not only the content underlies personal data protection laws but also the spoken word itself.

Therefore the speech data needs to be pseudonymized to protect the identity of the speaker. To generate disjoint test and training data sets, it is still necessary to be able to connect all recordings from one speaker. So all recordings are saved under a unique identifier that can not be traced back to the person. Data still can only be collected with explicit approval by each individual.

Furthermore, the patient's data needs to be protected. Therefore all spoken documentation entries need to be anonymized. This can be achieved in two ways: Either you let medical personal readout anonymized text or anonymize the audio of real documentation. First, we built a platform to record anonymized texts based on medical documentation provided by our partners 3.3 and contribute them to our data pool. But the vast amount of audio data necessary to achieve great model performance also showed us, that explicit data collection will not be sufficient to create a large enough data set. It would require too much additional time investment from our partner's medical staff. So additionally we build a system to record real medical documentation and anonymize it automatically.

3.1 Automatic speech data anonymization

To collect audio data from real medical documentation we need to extract the patient's name from the audio and thereby create anonymized audio. Ideally, this needs to be performed automatically to reduce manual labor. Our automatic name extraction pipeline works as follows:

1. Record Speech
2. Transcribe speech using the current speech model
3. Detect names in Text using Named Entity Recognition (NER) and Regular expressions

4. Use word timestamps in the audio file from the speech model to cut the audio
5. Store audio files on a central server
6. Manually screen recordings verify anonymization and correct transcriptions

Named entity recognition of names nowadays is incredibly good. We are using the open-source library CoreNLP [4] from Stanford for our purposes. It even comes with pre-trained entities for names. As names pose a considerable challenge to speech recognition engines it was important for us to be allowed to use the spoken names for training as well. With our consulting data protection officer we gained permission to do so by splitting salutation, first and last name apart, and thereby make the name unidentifiable.

3.2 Why should the speech data be created by medical personal?

This has two reasons: On one side these medical terms can not easily be read or correctly pronounced by anyone. On the other side, each professional field has a special demographic, and each demographic their way of speaking. For our model to perform to its fullest potential the training data has to be created by the people, the speech engine is used by. We decided to focus our model on nursing and care, thereby the speech recognition needs to be able to deal with different dialects, especially from eastern Europe.

3.3 Data collection partners

We wanted to generate speech data in cooperation with Zentrum für Psychatrie (ZFP). They are one of Germany's largest institutions for psychiatry care. With over 2600 medical staff they create thousands of medical documentation entries every day. Unfortunately due to Covid-19 and other circumstances we could not start the project with them and thereby were unable to generate the needed audio data. But we did get access to thousands of anonymized written reports which we used to build language models and texts to read out in our speech data collection tool.

After this setback we reached out to senior care facilities and found two senior care chains as new cooperation partners that did not only provide us with access to millions of documentation from the last years but will also support us in collecting speech data from their caregivers.

4 What we planned to use the Future SOC Lab for

Training acoustic models for speech recognition is computationally very demanding. The training time for the desired wav2letter model on a powerful desktop machine (with one NVIDIA GTX 1080 GPU) would exceed 4 months, according to the Zamia team. To experiment with different hyperparameters and training sets to compare our three desired models, significantly more resources are needed. The NVIDIA

DGX-1 would be the perfect tool for the job. With its 8 powerful GPUs, it would cut down training time to weeks instead of months.

5 Conclusion

The creation of a high-quality data set is a major challenge in any machine learning application. After many hurdles and set backs we are now on the right path to overcome this challenge. But we underestimated the time it would take and therefore unfortunately did not manage to finish our model training within this Future Soc Lab cycle. We will continue to collect speech recordings over the following months and start training and evaluation of our models as soon as possible.

Table 1: Project task list

Part	done
Find medical partners	yes
Clear legal requirements	yes
Build data collection tools	yes
Build model training pipelines	yes
Collect Data	just started
Train models	no
Model evaluation	no

6 Outlook

Use case-specific training in speech processing has huge implications across all industries. Each field requires a set of domain-specific terms to be detected that generally are not included in common speech. Therefore their vocabulary is not well represented in available data sets resulting in poor WER performance. A process of adding domain-specific terms to existing data sets that yields better recognition will be part of our research and can be adopted from our medical use case.

References

[1] A. Agarwal and T. Zesch. "German End-to-end Speech Recognition based on DeepSpeech". In: *Preliminary proceedings of the 15th Conference on Natural*

Language Processing (KONVENS 2019): Long Papers. Erlangen, Germany: German Society for Computational Linguistics & Language Technology, 2019, pages 111–119.

[2] G. Bartsch. *Zamia Speech*. Apr. 11, 2020. URL: https://zamia.org/asr/ (last accessed 2020-01-01).

[3] R. Collobert, C. Puhrsch, and G. Synnaeve. *Wav2letter: an end-to-end convnet-based speech recognition system*. 2016. arXiv: 1609.03193.

[4] *corenlp*. Nov. 1, 2020. URL: https://stanfordnlp.github.io/CoreNLP/ (last accessed 2020-01-01).

[5] *deepspeech*. Nov. 1, 2020. URL: https://github.com/mozilla/DeepSpeech (last accessed 2020-01-01).

[6] *deepspeech-polyglot*. Nov. 1, 2020. URL: https://gitlab.com/Jaco-Assistant/deepspeech-polyglot (last accessed 2020-01-01).

[7] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Ng. *DeepSpeech: Scaling up end-to-end speech recognition*. Dec. 19, 2014. arXiv: 1412.5567.

[8] A. Hannun, A. Lee, Q. Xu, and R. Collobert. *Sequence-to-sequence speech recognition with time-depth separable convolutions*. 2019. arXiv: 1904.02619.

[9] S. Hochreiter and J. Schmidhuber. "Long Short-Term Memory". In: *Neural Comput.* 9.8 (Nov. 1997), pages 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735.

[10] W.-N. Hsu, A. Lee, G. Synnaeve, and A. Hannun. *Semi-Supervised Speech Recognition via Local Prior Matching*. 2020. arXiv: 2002.10336.

[11] *Kaldi*. Nov. 1, 2020. URL: https://github.com/kaldi-asr/kaldi (last accessed 2020-01-01).

[12] T. Likhomanenko, G. Synnaeve, and R. Collobert. *Who needs words? lexicon-free speech recognition*. 2019. arXiv: 1904.04479.

[13] *Open Source Acoustic Models for German Distant Speech Recognition*. Nov. 1, 2020. URL: https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/acoustic-models.html (last accessed 2020-01-01).

[14] V. Peddinti, D. Povey, and S. Khudanpur. "A time delay neural network architecture for efficient modeling of long temporal contexts". In: *Interspeech 2015*. 2015, pages 3214–3218. DOI: 10.21437/Interspeech.2015-647.

[15] V. Pratap, Q. Xu, J. Kahn, G. Avidov, T. Likhomanenko, A. Hannun, V. Liptchinsky, G. Synnaeve, and R. Collobert. "Scaling Up Online Speech Recognition Using ConvNets". In: (2020), pages 3376–3380. DOI: 10.21437/Interspeech.2020-2840.

[16] G. Synnaeve, Q. Xu, J. Kahn, E. Grave, T. Likhomanenko, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert. *End-to-end ASR: from Supervised to Semi-Supervised Learning with Modern Architectures*. 2019. arXiv: 1911.08460.

[17] *Zamia Speech*. Nov. 1, 2020. URL: http://zamia-speech.org/ (last accessed 2020-01-01).

[18] N. Zeghidour, N. Usunier, I. Kokkinos, T. Schaiz, G. Synnaeve, and E. Dupoux. "Learning filterbanks from raw speech for phone recognition". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pages 5509–5513.

Information Retrieval for Cultural Heritage Data

Christian Bartz[1], Martin Lorenz[2], and Christoph Meinel[1]

[1] Internet Systems and Technologies
Hasso Plattner Institute for Digital Engineering
{firstname.lastname}@hpi.de
[2] Wildenstein Plattner Institute
{firstname.lastname}@wpi-art.org

The Wildenstein Plattner Institute is undertaking a massive digitization project, intending to make millions of previously unpublished cultural heritage information available to the broader public. The goal of this project is to use computer vision and machine learning methods for the automatic extraction of semantic metadata from scanned documents. We use the 1000 core cluster of the Future SOC Lab for the recognition of printed texts. Furthermore, we use the GPU cluster for preliminary experiments that shall lead to the creation of novel image synthesis methods, producing images that can be of use as annotated training data for state-of-the-art image segmentation methods.

1 Introduction

The Wildenstein Plattner Institute (WPI) is undertaking a massive digitization project, intending to make millions of previously unpublished cultural heritage information available to the broader public. After digitization, these millions of documents are only available in the form of digital images. While already allowing further conservation of such archival material is already possible, e.g., the search for keywords is not. The only possibility is to enrich the digitized documents with semantic metadata about the content of each document. Because of the mass of information, such metadata creation can only be done with the help of automated methods. The data of the WPI is manifold and contains many kinds of metadata. In this project, we are interested in examining printed and handwritten text contained in the digitized documents of the WPI. In the area of analyzing printed text, we use the off-the-shelf Optical Character Recognition (OCR) Print-OCR tool Tesseract [9]. In the area of handwriting analysis, there are no off-the-shelf tools available, yet. For handwriting analysis, we are developing new methods based on deep learning, which promise very good recognition capabilities on challenging data. The biggest challenge in the area of handwriting analysis with deep learning is the availability of annotated training data. The archive of the WPI, for instance, does not contain any annotations that we can use for the training of deep learning models. Our solution to this problem is the synthetic generation of training data that is as close as possible to the real data from the archive. The synthesis of training data for the analysis of handwriting will

also help us to improve the recognition results on printed text. The contributions made in our project can be summarized as follows:

1. Design of a pipeline for massive parallel application of OCR on scanned documents from the archive of the WPI (see section 2).
2. Development of a novel image reconstruction framework based on the StyleGAN framework [6, 7] that is later to be used for the generation of synthetic training data (see section 3).

We conclude our findings in section 4.

2 Massive Parallel OCR

The first task of our project is to use methods of print OCR for the extraction of textual content from 330 235 scanned pages of auction catalogues from the archive of the Wildenstein Plattner Institute. The amount of available pages makes it strictly necessary to use automated analysis and also a high-performance cluster that can handle many jobs in parallel. The Future SOC Lab cluster is very well suited for this task since it offers more than 1000 cores for parallel processing of our data. For recognizing the textual content of the printed text in the given auction catalogues, we use Tesseract OCR [9] and create a full processing pipeline. In the following, we describe the architecture of the pipeline we used for the analysis of the 330 235 scanned pages.

2.1 Pre-Processing

Before making use of Tesseract for text recognition, we need to perform several pre-processing steps. These pre-processing steps are necessary because Tesseract expects a specific input format and also further metadata about the provided image.

Image Loading The first step is to load the image containing the scanned page. Since we are dealing with a large number of images, the file containing all known metadata about each scanned page is large (> 20 MB). Loading the file for the analysis of each file would produce a large overhead. Therefore, we developed a RESTful server that provides one set of metadata for each worker on request. A set of metadata consists of the image path, country and city of the corresponding auction.

Language Guessing Tesseract requires the user to supply it with information about the language contained in the document to analyse. The data provided by the WPI is quite heterogeneous when it comes to languages. Most of the documents are in French, with others written, e.g., in English, or Italian. Thanks to the metadata provided by the WPI, we can guess the most probable language contained in a given scan, by making use of city and country supplied as metadata for each scanned page.

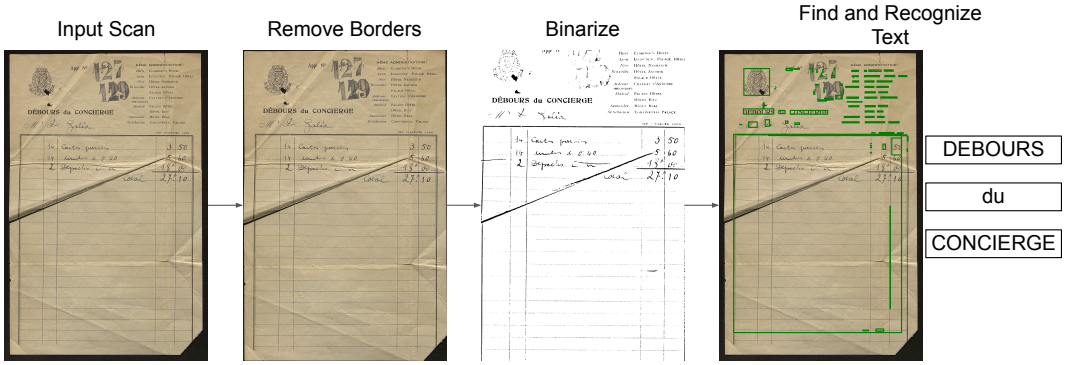


Figure 1: Overview of our pipeline for recognizing textual content of scans. The green boxes in the most-right image show the regions that Tesseract handles as text regions. The example shows that the method can keep printed text, but many other unwanted types of data, such as lines of the table, handwriting, and stamps, are not discarded. In our future work, we wish to address this problem.

Margin Removal and Binarization Another pre-processing step is to binarize the scanned page. Binarization is the process of classifying each pixel of a digital image into foreground or background, resulting in a black and white image with only two possible values for each pixel. Before applying binarization, we remove the black margin of the scanning device, which can be found in each scanned page. Margin removal is a crucial step because otherwise, we can not perform binarization successfully.

Margin removal involves a few simple steps. First, the input needs to be blurred to remove disturbing noise introduced while scanning. The next step is to perform Canny edge detection [4], followed by detection of all contours using the border following the approach by Suzuki et al. [10]. The found contours are now sorted based on their area. In this sorted list of contours, we now look for the pair that has the greatest difference in size. We then take the contour of that pair with the larger area as the contour of our scanned page without the scanning border and crop this box from the given scan.

After we received the image without the scanning margin, we again blur the image and perform adaptive gaussian thresholding for binarization of the input image.

Orientation Correction Scans of documents are not guaranteed to be perfectly axis-aligned. Tesseract expects given documents to be as axis-aligned as possible. To get an axis-aligned image, we use the orientation and script detection module of tesseract and rotate the input image based on the output of this module.

2.2 Text Recogniton

After pre-processing, we receive a binarized page that is axis-aligned. We also receive information about the language of the content of the document. Using this information, we provide tesseract with our scanned page. Tesseract returns the content of each printed text instance found and also the location of each text instance. We return the extracted data to the WPI in the form of a *JSON* document. In figure 1, we provide a structural overview of the proposed analysis pipeline.
Running the full pipeline for one image takes about 10 seconds per image. If we were to run our recognition pipeline using only one job, this analysis would take ca. 38 days. Using the power of the 1000 Core Cluster of the Future SOC Lab, we were able to cut this time to 2 days.

2.3 Future Work

The described approach works well for the given data. However, there are some parts of the pipeline that should be improved in the future. The most important part that needs to be improved is the binarization step. Binarization using adaptive gaussian thresholding is a simple approach and leads to noisy artefacts in the resulting image. These noisy artefacts decrease the performance of Tesseract, as they introduce many false-positive recognition results. Furthermore, handwritten annotations contained in a scanned page are used as input to Tesseract, too. Using handwritten annotations as input is a problem since Tesseract is not able to recognize handwriting. In the future, we want to develop a better binarization strategy that decides for each pixel, whether it belongs to handwriting, printed text, an image, or the background.

3 Synthetic Data for Handwriting Recognition

As already indicated in section 2.3, the performance of the used binarization method is not optimal. Our simple binarization approach produces false-positive text regions because it is not able to discard regions containing handwritten text. In the following, we briefly describe our idea and first results of a novel binarization method based on deep learning, using synthetic training data.

In recent years novel mechanisms for semantic segmentation of images have been proposed [5, 8]. These methods are based on deep learning and train a deep neural network to decide for each pixel of a given input image, which class this pixel belongs to. During training, an input and a segmentation image (the expected result) need to be provided to the model. Using such an approach would be the best solution for binarizing the data in the archive of the WPI. However, we do not have access to the required segmentation images. A possible solution is to use recent advances in neural image generation using Generative Adversarial Networks (GANs) [6, 7] for the synthetic generation of segmentation images. To this end, we used the resources of the Future SOC Lab (GPU cluster) to train preliminary models that can synthesize images. On the one hand, these images should look as real as possible, compared to

our original images. On the other hand, we also managed to find a faster approach for the reconstruction of original images based on [1, 2]. We explain our preliminary findings in detail in [3]. Based on our approach, we plan to use the resources of the Future SOC Lab for building a model that can not only synthesize images that look as similar as possible to scanned documents from the archive of the WPI, but that is also able to produce a segmentation image at the same time.

4 Conclusion

In our project, we tackle the analysis of scanned documents of the Wildenstein Plattner Institute to make millions of previously unpublished cultural heritage information available to the broader public. We make use of computer vision algorithms and extract semantic metadata describing textual content from scanned documents. To this end, we used the infrastructure provided by the Future SOC Lab to perform OCR on more than 300 000 scanned pages in a massively parallel way, cutting the processing time from weeks to only two days. Furthermore, we used the GPU infrastructure of the Future SOC Lab for developing novel image synthesis methods based on deep learning, work that we wish to continue in the next usage period.

References

[1] R. Abdal, Y. Qin, and P. Wonka. "Image2StyleGAN: How to Embed Images Into the StyleGAN Latent Space?" In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pages 4432–4441. URL: http://openaccess.thecvf.com/content_ICCV_2019/html/Abdal_Image2StyleGAN_How_to_Embed_Images_Into_the_StyleGAN_Latent_Space_ICCV_2019_paper.html (last accessed 2020-05-06).

[2] R. Abdal, Y. Qin, and P. Wonka. "Image2StyleGAN++: How to Edit the Embedded Images?" In: *CVPR 2020*. Nov. 2019. arXiv: 1911.11544 [cs.CV].

[3] C. Bartz, J. Bethge, H. Yang, and C. Meinel. *One Model to Reconstruct Them All: A Novel Way to Use the Stochastic Noise in StyleGAN*. 2020. arXiv: 2010.11113 [cs.CV].

[4] J. Canny. "A Computational Approach to Edge Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (Nov. 1986). ISSN: 1939-3539. DOI: 10.1109/TPAMI.1986.4767851.

[5] K. He, G. Gkioxari, P. Dollar, and R. Girshick. "Mask R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pages 2961–2969.

[6] T. Karras, S. Laine, and T. Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *CVPR 2019*. Mar. 2019. arXiv: 1812.04948 [cs.NV].

[7] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. *Analyzing and Improving the Image Quality of StyleGAN*. Dec. 2019. arXiv: 1912.04958 [cs.CV].

[8] O. Ronneberger, P. Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. 2015.

[9] R. Smith. "An Overview of the Tesseract OCR Engine". In: *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*. 2007, pages 629–633.

[10] S. Suzuki and K. be. "Topological structural analysis of digitized binary images by border following". In: *Computer Vision, Graphics, and Image Processing* 30.1 (Apr. 1985), pages 32–46. ISSN: 0734-189X. DOI: 10.1016/0734-189X(85)90016-7.

Tooling for big data extraction

Analysis of the chronological usage of JavaScript libraries across the WWW

Robert Beilich

Technische Hochschule Brandenburg
beilich@th-brandenburg.de

The decision for the best JavaScript library to use in web applications is difficult, due to the vast number and the short lifespan of a lot of them. This project strives to apply the decisions made in the past to the future, by analysing a dataset of web crawls, provided by Common Crawl, what may causes libraries to crumble and others to rise.

1 Introduction

Working with big datasets is a challenge, which reaches more and more aspects of our lives. Starting from the obvious candidates like analysing the shopping habits of customers or monitoring ones health condition, with the help of smart wearables and finishing at not so obvious examples like optimizing fueling in the transport industry. [6] Efficiently gathering information from unstructured data like web pages is an even more demanding process, as it also includes finding the relevant information instead of just utilising it.

1.1 Motivation and Idea

The goal is to extract the usage of JavaScript libraries from historical data. This usage data should then get analysed to display, for example, the most used libraries of a given year or to make predictions about the success of new libraries, based on information about existing libraries. These insights could then enable developers to decide, if it is worth to invest time into a new library, or if it has no future.

1.2 The dataset

The Common Crawl Foundation created "an open repository of web crawl data that can be accessed and analyzed by anyone"[2]. This crawl data dates back to 2008 and is currently extended on a monthly basis. Since 2013 the data is saved as Web Archives (WARCs)[3]. The WARC format aims at joining metadata and actual data of a chained list of records in one archive. [5] For the Common Crawl dataset this results in three types of WARC records: response, request, and metadata. The response records include the HTTP response that was crawled, which is the target data for this thesis. The other two types, which include information about the crawl

process and how the data was requested, are ignored. [4] Ignored as well are the provided Web Archive Transformation (WAT) and WARC Encapsulated Text (WET) files, which include extracted metadata and plain text respectively.

One crawl consists of round about 60.000 WARCs of around 50 to 60 TiB of compressed data, which includes roughly 2.5 billion web pages. These WARCs are stored "on Amazon S3 as part of the Amazon Public Datasets program"[3]. This has multiple benefits besides the public availability of the data, such as easier access to the data or — which is highly important for the maintainer of the data — the coverage of the storage cost.[1] Access to the data is provided either from inside Amazon Web Services(AWS) by directly accessing the S3 instances or openly for everyone via simple https calls.

2 Execution environment

During the prototyping phase the project ran on a small privately-owned server, with 32 GB of RAM and 8 hyper-threaded cores. This minimal setup was sufficient for initial testings, but not for actual executions. Therefore the project was proposed to participate in the Future SOC Lab program of the Hasso-Plattner-Institut (HPI).

Future SOC Lab resources

Of the resources provided by the Future SOC Lab, the multi-core machines are the ones used for this project. These machines are housing 1 TB of RAM and 40 cores in a non-uniform memory access (NUMA) architecture, where memory is physically located on different busses. One of these machines was solely used for the database and one other machine to run multiple instances of the parser, that extracts the JavaScript libraries from the web pages saved in the Common Crawl dataset.

3 Findings

The main tasks of the parser are to extract the data from the web pages and to persist it to the database. Increasing the performance of these two tasks results in the most benefits in terms of execution time. The major bottleneck was identified as network bandwidth. The high amount of processing power, provided by the Future SOC Lab, cannot get exhausted, because the limiting factor of this project is the immense data source, that is accessed over the internet. As the whole Future SOC Lab is bound to one outgoing connection, the amount of data, that can get processed, is bound to the amount of data, that can get downloaded. Because of this bottleneck and the extrapolated time — of approximately nine years — needed to process the whole dataset, the processing was stopped, when the size of database reached 1 TB. This accounts for 0.0046 % of the current dataset — which increases monthly — and by that 829 million processed web pages.

Listing 1: Basic Beautiful Soup example

```
from bs4 import BeautifulSoup

# ... retrieval of web page here ...

soup = BeautifulSoup(page_content, 'lxml')
scripts = soup.find_all('script')
for script in scripts:
   print(script.get('src'))
```

3.1 Parser

In table 1 different iterations of the used extraction algorithm are shown.

Table 1: Comparision of source extraction methods

Method	time for 500 urls	time for 1 crawl	working
Beautiful Soup	28s	3.5y	yes
Beautiful Soup (filter)	23s	2.9y	yes
Regex	0.002s	9d	no
lxml	1s	46d	yes

Beautiful Soup [7] may be the go to solution, when it comes to parsing web pages in Python, but it also comes with some issues. Extracting the sources from a web page with Beautiful Soup is as simple as shown in listing 1, but also quite slow. Even applying filters to Beautiful Soup gives no significant performance boost.

Processing HTML with regular expressions proved to be fast, but unreliable, as the regex would not catch edge cases until they are added to it. The edge cases of malformed HTML are endless and therefore regexes are no viable solution to extract the used JavaScript libraries from web pages.

Ultimately the underlying parser used by Beautiful Soup is tested and gives a reasonable extraction time, when the processing is parallelized.

3.2 Analysis

To exemplary analyse the data, a ranking of the top ten used libraries and for each of them a timeline of usage is generated. This is shown in figure 1 and figure 2 respectively. The timeline is displayed as percentage of total web pages processed for each timestamp, as the actual amount of web pages varies across the timespan.

The graphs only display the results of 0.0046 % of the whole dataset, but aggregation on even less data showed similar results, which indicates, that it stays mostly

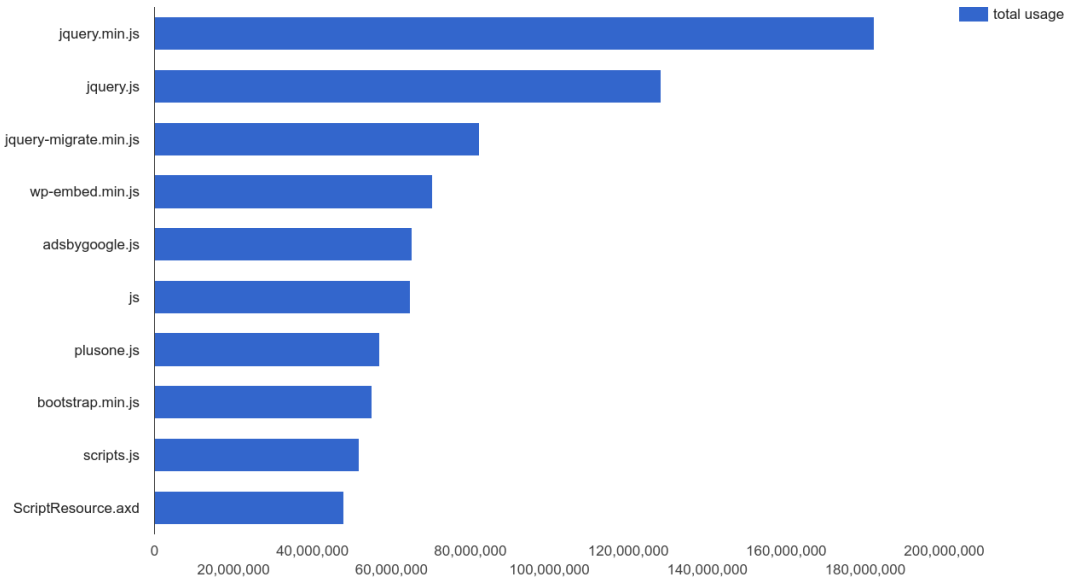


Figure 1: Top 10 libraries by occurrences

true for rest as well. This means that even nowadays jQuery is used on 45 % of all web pages. This number is the sum of `jquery.js` and `jquery.min.js`, which both represent the same library, but are as of now identified as separate ones. The timeline also shows that since May 2017 `jquery.js`, `jquery.migrate.min.js` and `wb-embed.min.js` are increasing and decreasing at the same rate, which could signify, that the three are used in conjunction.

4 Next steps

Going forward, the main focus will be on the extraction process, the persisting of the data and the data model. Enhancing these, increases the performance the most, when it comes to processing the data.

As the network bandwidth is the major bottleneck, the instances of the parser need to get distributed across multiple networks. The executing machines itself don't need to be as potent as the ones used in the Future SOC Lab.

To better analyse the data, further cleaning is necessary to better identify JavaScript libraries and different versions of the same one. Additionally, libraries and frameworks, which use webpack[1] or similar tools to flatten the code, produce single files with generated names. This hides the information which libraries are used, as these libraries a bundled together with the application code and the name of the script file does not contain the name of the library anymore. This has to be taken into account and further investigation into the process of identifying them has to be done.

[1] https://webpack.js.org/ (last accessed 2020-01-01).

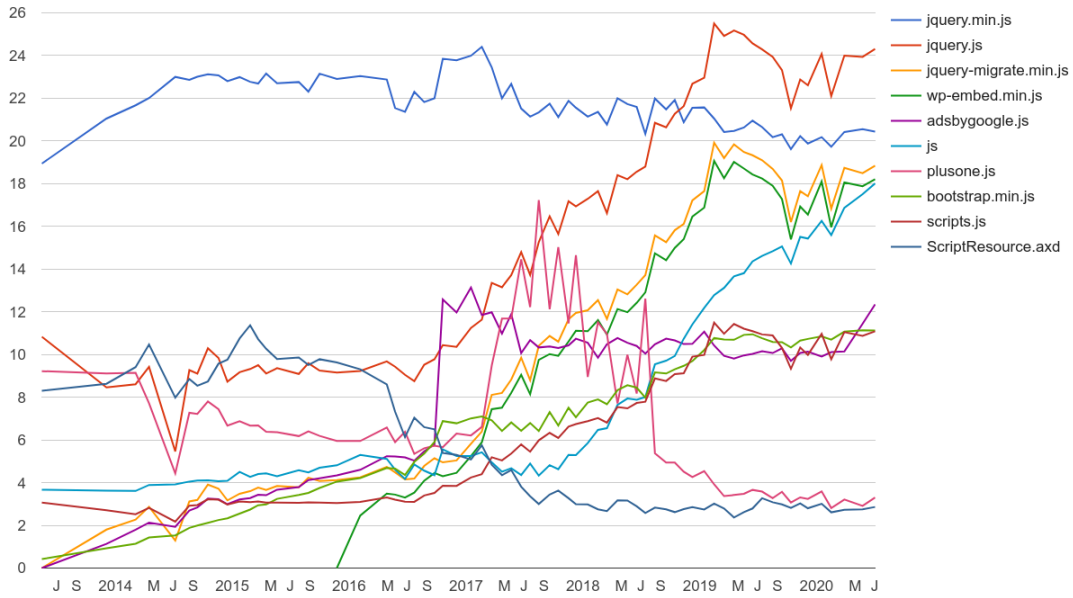


Figure 2: Timelines of Top 10 libraries by percentage of parsed web pages

References

[1] Amazon. *Open Data on AWS*. URL: https://aws.amazon.com/opendata/ (last accessed 2020-07-30).

[2] Common Crawl. *Common Crawl*. URL: https://commoncrawl.org (last accessed 2020-07-30).

[3] Common Crawl. *So you're ready to get started.* URL: https://commoncrawl.org/the-data/get-started/ (last accessed 2020-07-30).

[4] Common Crawl. *WARC Format*. URL: https://commoncrawl.org/the-data/get-started/#WARC-Format (last accessed 2020-07-30).

[5] ISO. *ISO 28500:2017 Information and documentation — WARC file format*. URL: https://www.iso.org/standard/68004.html (last accessed 2020-07-30).

[6] M. Rice. *17 Big Data Examples and Applications*. URL: https://builtin.com/big-data/big-data-examples-applications (last accessed 2020-10-10).

[7] L. Richardson. *Beautiful Soup: We called him Tortoise because he taught us*. URL: https://www.crummy.com/software/BeautifulSoup/ (last accessed 2020-07-30).

Fast and Non-Invasive Diagnosis of SARS-COV-2 Infection via Raman Spectroscopy and Deep Learning

A preliminary study

Dario Bertazioli[1], Cristiano Carlomagno[2], Marzia Bedoni[2], and Vincenzina Messina[1]

[1] DISCO, University of Milano Bicocca
d.bertazioli@campus.unimib.it,vincenzina.messina@unimib.it
[2] LABION, Fondazione Don Gnocci
{ccarlomagno,mbedoni}@dongnocchi.it

The pandemic of the coronavirus disease in 2020 (COVID-19) is continuously spreading, becoming a worldwide emergency. Early, fast and precise identification of subjects with a current or past SARS-CoV-2 infection must be achieved to slow down the epidemiological spreading and to limit the number of new infections. Nowadays, available tests present limitations including the performance time, discrimination power, information obtained, costs and availability for a continuous monitoring of the population. We present a Raman-based approach for the analysis of saliva, able to significantly discriminate the signal from patients with a current infection by COVID-19 from healthy subjects and/or with a past infection. Our results demonstrated the differences in saliva biochemical composition of the three experimental group involved (COVID, CTRLs, COV-NEG). According to the preliminary evaluation, the developed Deep Learning Pipeline achieve an accuracy score of 87 % in the patient discrimination. These findings have implications for the creation of a potential Raman-based diagnostic tool, using saliva as minimal invasive and highly informative biofluid, demonstrating the efficacy of the classification model.

1 Introduction

SARS-CoV-2 is a coronavirus belonging to betacoronavirus genus lineage B, related to SARS-like viruses with identified differences in genetic material and surface proteins. At present, COVID-19 diagnosis is mainly based on epidemiological history, clinical manifestations, CT and nucleic acid test results. Many studies on the well characterized SARS-CoV, and theoretically translable on the SARS-CoV-2, demonstrated the early infection of epithelial cells of the major and minor salivary gland ducts with a primary accumulation of virus bodies into saliva [5]. With the infection progression, the viral load into saliva grows exponentially, converging into saliva different secretion coming from other districts including upper and lower respiratory tracts, nasopharingeal and blood vessels through crevicular fluid. Raman Spectroscopy (RS) is a vibrational-based detection method able to provide biochemical information from a target biofluid in a fast, sensitive, non-destructive and automatable way.

The range of biological molecules simultaneously detectable using RS vary from proteins to lipids, nucleic acids, hormones, metabolites, cells, bacteria and viruses, obtaining information regarding their presence, concentrations, interactions, environment and mutations. The identification of a SARS-CoV-2 Raman signature carry information about the molecular characterization of the analyzed sample (saliva in this study) and when using a portable Raman easily could represent an extremely useful, fast and low cost point-of-care (POC) for the viral infection diagnosis.

Contributions

Concretely, this work deals with the technical implementation of a diagnostic procedure that could be applied to several pathologies. In particular, to demonstrate the power and the flexibility of the general pipeline, these methods have been tested on three different neurodegenerative pathologies, Amyothrofic Lateral Sclerosis (ALS), Alzheimer (AD) and Parkinson (PD) diseases [1, 3], and on the recent and widespread infection caused by the epidemic SARS-COV2 coronavirus. This report focuses on this last mentioned application. A high-level overview of the methodology could be the following: when an individual is suspected to be affected by one of the aforementioned diseases, a saliva sample is collected and analysed by means of a Raman Spectrometer, encoding the information contained in the biofluid in the form of spectral data. Given the low-frequency (thus, low-energetic) characteristics of the Raman method, the acquired spectrum is difficult to interpret. At this point, the application of some machine learning (ML) and deep learning (DL) algorithms enable a fast and automatic data-analysis which results in a classification of the involved patients indicating the presence or absence of the suspected disease.

2 Context

Raman Spectroscopy (RS) is a spectroscopic technique that, given a target sample to be investigated, typically a molecular system, allow to study and observe low-frequency modes, also called rotational, vibrational and rotovibrational modes. The "modes" of a molecular aggregate could be seen as a collective behaviour of its components in response to a perturbation of the system (in terms of energy): intuitively, the components of the molecules (i.e. atoms), excited by a laser beam, vibrates and rotates collectively until a phonon, a quantum of rotovibrational energy, is released after the de-exitation. The physics of RS is rather simple: when an incident photon, part of a laser beam, is scattered by the target, its energy is modified with respect to the initial conditions. Filtering out other components of the scattering process, such as the Rayleigh scattering or the infrared absorption components, this energy shift between the incident and the scattered photon can be associated with a rotational or vibrational mode of the target, as previously mentioned. This process is referred to as Raman scattering, a particular type of inelastic scattering. Therefore, by varying the frequency of the incident laser beam, scientists are able to map the presence and the characteristics of the low-frequency modes at given energy by observing

the energy shifts at which the light is emitted after being Raman Scattered. The obtained mapping allows determining a sort of "fingerprint" (also referred to as *Raman Fingerprint*) directly associated with the molecular composition of the target sample. Since the RS exploits laser excitements, it is non-invasive and non-destructive, i.e. the involved energy quantity is small and no radiations are produced. Due to those characteristics, in addition to being sensitive, rapid and fully automatable, the "Raman Fingerprints", besides their wide use in chemometrics applications, have shown their potential in the biomedical field: RS can, indeed, describe the chemical composition of a sample, producing an overview regarding the presence, concentrations and interactions of molecules contained in a molecular system . In addition, the intrinsic safety and non-destructiveness of RS techniques allows the creation of highly sensitive portable spectrometers which can be easily employed as biosensing point of care in clinical areas. Biomedical applications of RS involve a variety of target samples, typically biofluids such as Cerebrospinal or blood-based (mostly serum) fluids. However, the search for meaningful biomarkers, which could potentially unveil the presence or insurgence of a disease in a patient, is hindered by the invasiveness of their collection procedure, which is not even always feasible in cases of severe degenerations and therapy monitoring in late-stage patients. On contrary, in recent research, the analysis of saliva samples has demonstrated the potential of such a biofluid for the identification of the presence and relative concentrations of relevant biomarkers, making saliva one of the most promising analysis-targets, especially considered its extremely easy accessibility and the non-invasiveness of its collection.

3 Background

Carrying out a classification of spectral data belonging to COVID-affected patients, heathy controls, and COVID negativized patients (those who successfully recovered from the disease) we considered both Machine Learning (ML) and Deep Learning (DL) techniques to automate the analysis of the acquired RS data for unveiling hidden patterns that correlate with the pathologies object of this study [2]. In particular, in order to build a suitable ML classification model, the huge amount of information contained in a single Raman spectrum must be processed in order to identify similarity or distance between spectra and characteristic peaks. Spectral pre-processing is, therefore, an essential step for both MultiVariate Analysis (MVA) and Machine Learning and can strongly affect the identification performances. Beside classic ML algorithms, DL approaches, stemmed from Artificial Neural Networks, gained increasing attention in the last few years. DL models can be considered as belonging to a subset of ML techniques characterized by the "Representation Learning" ability: features (data representation) are no-longer "handcrafted" (as required in the most classical ML algorithms) but synthesized along with the main task (i.e. classification). The term "Deep" refers to the multiple-layer structure of the model, where each layer elaborates the input feature so that the next one can deal with a better

representation of the data achieving more structured information. Recently, DL algorithms have been applied to Raman Spectroscopy, demonstrating their superior performances with respect to ML techniques with the additional advantage of being able to directly handle raw data without requiring time-consuming preprocessing. By applying DL techniques it is, therefore, possible to obtain two crucial objectives: reduce the dimensionality of the acquired data and train a classification model to recognize and classify the input spectra collected from the healthy person or from the patient affected by a specific pathology. This suits well the need to represent the complex connections characterizing high dimensional data as RS. In particular, among the various DL architectures, Convolutional neural networks (CNNs), inspired by the biological visual cognition mechanism, have been proved to be very effective for this application.

4 Classification: the Machine Learning Pipeline

In this work, we propose the Convolutional Neural Networks (CNNs) depicted in figure 1 characterized by 3 convolutional layers, each followed by a Max-pooling and Batch Normalization operation, and 3 Fully-Connected layers regularized through the use of Dropout. This choice is the result of an extensive phase of computational

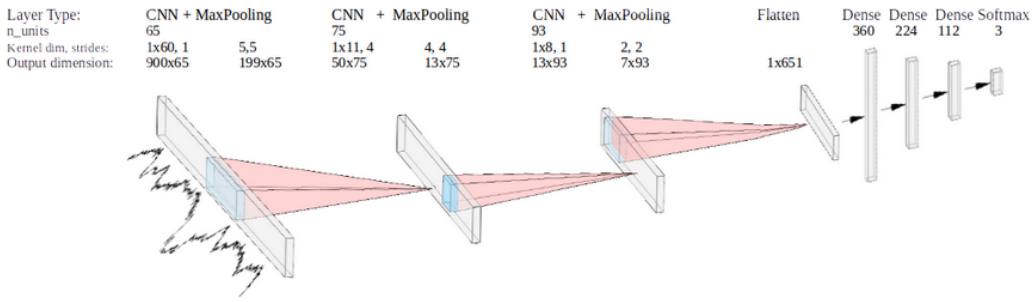


Figure 1: A graphic representation of the best 1D-CNN model configuration obtained through the HPO process. Figure taken from [2].

experiments, where we compared different ML models and DL architectures and, for the most promising candidates, we applied HyperParameter Optimization (HPO) for the automatic selection of their optimal configuration, namely random search for the optimization of ML models while for the DL models, TPE sampling and Gaussian Process-based Sequential Model-based Optimization has been used to optimize our FCNN and CNN, respectively. We considered SVM, RF, and XGB as baseline ML models, while different architectures from the family of DL models, namely Fully Connected Neural Networks (FCNNs) and CNNs, were tested. Indeed ML techniques, such as Support Vector Machine (SVM), Decision Trees and Random Forest,

have been widely applied to the classification of spectral signals. The application of ML algorithms, as well as classical MVA, required an initial preprocessing phase, including outlier removal, spectral realignment, noise removal, despiking, and normalization, followed by a dimensionality reduction step, e.g. applying principal component analysis (PCA). In contrast, DL models show to be potentially capable of directly handling both the preprocessed and the raw data [4]. Moreover, in order to partially overcome the problem of data scarcity given by the relatively reduced number of patients in our dataset, and to avoid overfitting problems boosting the generalization capability of the proposed models, we implemented a Data Augmentation procedure aimed at generating synthetic data samples by applying variations and distortions to the original data in order to maximally explicit their intrinsic invariances. In particular, Data Augmentation applied to Raman Spectroscopy makes it possible to simulate the spectral imperfections and variations that are characteristics of the measuring process. This could be obtained by injecting a rather small contribution of Gaussian noise to the original spectra (at random wavenumbers). Other augmentation steps include the modulation of the offset and the slope.

All these steps have been integrated into a pipeline, characterized by a modular structure providing a flexible and systematic experimental framework, as depicted in figure 2.

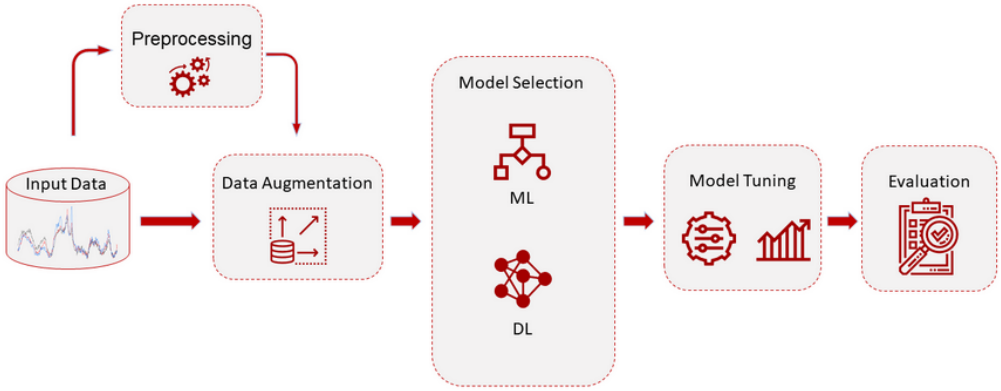


Figure 2: Diagram of the proposed pipeline. An optional preprocessing phase, suitable for ML models, is followed by a data augmentation procedure, after which a model selection step is performed by comparing several ML and DL models. The most promising models undergo hyperparameter optimization. The performances of the optimized algorithms are then evaluated in a Leave-One-Patient-Out Cross-Validation. Figure taken from [2].

As the last step of this pipeline, a rigid performance evaluation scheme has been designed. Since in this work we are mainly interested in discriminating between COV, CTRL and COVNEG patients we considered as the main task the "complete" multiclass classification problem (COV vs CTRL vs COVNEG). For a further analysis

aimed at gaining deeper insight into the classification outcomes, the binary classification cases (COV vs CTRL, COV vs COVNEG, COVNEG vs CTRL) have been investigated. First of all, we built a single spectra classification model, evaluating spectra-level metrics. Afterwards, keeping in mind that we had multiple spectra associated with each patient, we aggregated the classification result at a patient-level: each patient is classified according to the majority label assigned to his/her spectra. Another important consideration that influenced the evaluation phase is that the number of distinct patients in the dataset is rather limited and this could compromise the reliability of the evaluation performance indices in a classic ML evaluation procedure, for example applying a regular hold-out strategy. Therefore, we apply the Leave-One-Patient-Out Cross-Validation, a robust and stable procedure where each test-fold is composed by the entire set of spectra from a single patient, that guarantees a most accurate estimate of the model performances. This procedure also prevents any classification bias given by an arbitrary test-set choice.

5 Results and Conclusions

When applying DL techniques to spectral data, a series of challenges arise. Due to the rather high complexity (and capacity) of DL models, the available data volume should be large enough to guarantee a uniform and coherent sampling from the real-data distribution, enabling the algorithm to succeed in the classification task. In particular, when the focus is on a patient-level classification, the smoothness and goodness of the (sampled) data distribution strictly depend on the number of individuals involved in the data collection process: the higher the number of patients, the most probability for a model to generalize well, and, in contrast, the fewer the patients the harder the training results. Such a consideration forced us in designing a Data Augmentation protocol to generate synthetic spectral examples with a two-fold objective: favouring the network training and improving the classification performances. High complexity and capacity usually lead to a large number of almost equally valid configurations for the components of a DL model. A great effort is therefore required to choose a suitable architecture through the composition of various layers and, especially for CNNs, to select an (at least sub-) optimal "fine hyper-parameter" configuration. A few options are available to accomplish such a task. One is to obtain this design by a trial and error method, which is the popular choice in the related literature but could be time-consuming even for a domain expert and it does not give any guarantee of convergence to an optimal configuration. Another recent option relies on HPO techniques, that recently gained traction in DL model design. In the light of HPO, we optimized the CNN architecture and fine-tuned its hyperparameters through SMBO: our CNN architecture consists of three 1-D convolutional layers for the feature extraction and three fully connected layers for the classification, with the final configuration represented in figure 1.

We noticed that this step empirically resulted in increased model performances. From our initial exploration and machine learning baseline, we deduced that learning problem is a quite difficult one, mainly due to a low-volume high-dimensional

dataset. For this reason, both the ML and DL models have been trained on the data preprocessed along with the pipeline described in the Methods, where an evaluation of the possible normalization methods led us to the choice of L2 normalization technique for our experiments. We also tested the performances of our DL models against on raw unprocessed data, and we obtain that, in good agreement with [4], CNNs are particularly capable of gaining competitive performances while skipping the preprocessing steps. Therefore the "raw-CNN" performances will be reported along with the others. The CNN architecture has been trained on the 3-class classification problem to output, through a soft-max, a probability distribution associated with the class choices, where the highest probability is taken as a prediction of the corresponding label. The model is tested via a Leave-One-Patient-Out Cross Validation method. Practically, the labelling decision for a patient can be taken based on its entire spectral set (since numerous spectral samples are acquired from the same individual). Furthermore, the percentage of spectra belonging to the same patient and correctly classified is itself a confidence indicator. Classification metrics can thus be condensed into a new confusion matrix grouped by patients, where the average accuracy score of the CNN model is 0.87 on raw data (0.84 on preprocessed data). In comparison, SVM, RF, and XGB have, respectively, accuracy rates of 0.717, 0.772, and 0.851 on preprocessed data with the application of PCA.

In conclusion, the preliminary results indicate that a promising Raman-based diagnostic system capable of discriminating COVID-affected patients from healthy ones and even from negativized patient could be considered as a Point-of-Care thought portable Raman spectroscopy. Of course, further analysis and studies are required, in order to asses the generalization capability of our pipeline in proper clinical settings. More salivary samples are expected to be collected in order to further increment the data volume, allowing the models to be trained on a wider amount of experience, thus being likely to provide better and more stable results.

References

[1] C. Carlomagno, P. I. Banfi, A. Gualerzi, S. Picciolini, E. Volpato, M. Meloni, A. Lax, E. Colombo, N. Ticozzi, F. Verde, V. Silani, and M. Bedoni. "Human salivary Raman fingerprint as biomarker for the diagnosis of Amyotrophic Lateral Sclerosis". In: *Scientific Reports* 10.1 (2020), page 10175. ISSN: 2045-2322. DOI: 10.1038/s41598-020-67138-8.

[2] C. Carlomagno, A. Gualerzi, S. Picciolini, P. I. Banfi, A. Lax, D. Bertazioli, V. Messina, J. Navarro, Clerici, Arienti, Bianco, Caronni, and M. Bedoni. "The Raman COVID-19 salivary fingerprint: a fast and sensitive approach for the characterization of COVID-19 onset from saliva." In: *Nature Communications* (2020). Submitted.

[3] C. Carlomagno, S. Picciolini, A. Gualerzi, M. Meloni, P. I. Banfi, Aux, D. Bertazioli, V. Messina, and M. Bedoni. "Characterization of a salivary Parkinson's

disease fingerprint through Raman spectroscopy and Machine Learning". In: *Progress in Neurobiology* (2020). Submitted.

[4] J. Liu, M. Osadchy, L. Ashton, M. Foster, C. J. Solomon, and S. J. Gibson. "Deep Convolutional Neural Networks for Raman Spectrum Recognition: A Unified Solution". In: *Analyst* 142.21 (Aug. 2017), pages 4067–4074. DOI: 10.1039/C7AN01371J. arXiv: 1708.09022.

[5] W.-K. Wang, S.-Y. Chen, I.-J. Liu, Y.-C. Chen, H.-L. Chen, C.-F. Yang, P.-J. Chen, S.-H. Yeh, C.-L. Kao, L.-M. Huang, P.-R. Hsueh, J.-T. Wang, W.-H. Sheng, C.-T. Fang, C.-C. Hung, S.-M. Hsieh, C.-P. Su, W.-C. Chiang, J.-Y. Yang, J.-H. Lin, S.-C. Hsieh, H.-P. Hu, Y.-P. Chiang, J.-T. Wang, P.-C. Yang, S.-C. Chang, and members of the SARS Research Group of the National Taiwan University/National Taiwan University Hospital. "Detection of SARS-associated coronavirus in throat wash and saliva in early diagnosis". In: *Emerging infectious diseases* 10.7 (July 2004), pages 1213–1219. ISSN: 1080-6040. DOI: 10.3201/eid1007.031113. PMID: 15324540.

Current Technical Reports from the Hasso-Plattner-Institut

Vol.	ISBN	Title	Authors/Editors
158	978-3-86956-564-4	**HPI Future SOC Lab – Proceedings 2019**	Christoph Meinel, Andreas Polze, Karsten Beins, Rolf Strotmann, Ulrich Seibold, Kurt Rödszus, Jürgen Müller (Hrsg.)
157	978-3-86956-561-3	**Digital sovereignty: insights from Germany's education sector**	Christoph Meinel, Michael Galbas, David Hagebölling
156	978-3-86956-560-6	**Digitale Souveränität: Erkenntnisse aus dem deutschen Bildungssektor**	Christoph Meinel, Michael Galbas, David Hagebölling
155	978-3-86956-556-9	**Triple graph grammars for multi-version models**	Matthias Barkowsky, Holger Giese
154	978-3-86956-555-2	**Modular and incremental global model management with extended generalized discrimination networks**	Matthias Barkowsky, Holger Giese
153	978-3-86956-551-4	**Human pose estimation for decubitus prophylaxis**	Benedikt Weber
152	978-3-86956-550-7	**RailChain : Abschlussbericht**	Ingo Schwarzer, Said Weiß-Saoumi, Roland Kittel, Tobias Friedrich, Koraltan Kaynak, Cemil Durak, Andreas Isbarn, Jörg Diestel, Jens Knittel, Marquart Franz, Carlos Morra, Susanne Stahnke, Jens Braband, Johannes Dittmann, Stephan Griebel, Andreas Krampf, Martin Link, Matthias Müller, Jens Radestock, Leo Strub, Kai Bleeke, Leander Jehl, Rüdiger Kapitza, Ines Messadi, Stefan Schmidt, Signe Schwarz-Rüsch, Lukas Pirl, Robert Schmid, Dirk Friedenberger, Jossekin Beilharz, Arne Boockmeyer, Andreas Polze, Ralf Röhrig, Hendrik Schäbe, Ricky Thiermann
151	978-3-86956-547-7	**HPI Future SOC Lab – Proceedings 2018**	Christoph Meinel, Andreas Polze, Karsten Beins, Rolf Strotmann, Ulrich Seibold, Kurt Rödszus, Jürgen Müller (Hrsg.)

ISBN 978-3-86956-556-9
ISSN 1613-5652