VU MCS research group - SUPERVISION

TDM - project supervised by
dr. Tiziano DE MATTEIS
t.de.matteis@vu.nl
Keywords: HPC, Accelerators, Programming Languages, Performance
11A-33, NU Building

AI - project supervised by
prof.dr.ir. Alexandru IOSUP
a.iosup@vu.nl
11A-30, NU Building

MC - project supervised by
Michael R. Crusoe
m.r.crusoe@vu.nl

JD - project supervised by
ir. Jesse Donkervliet
jesse.donkervliet@vu.nl
11A-13, NU Building

MJ - project supervised by
Matthijs Jansen, MSc
m.s.jansen@vu.nl
11A-13, NU Building

ST - project supervised by
ir. Sacheendra Talluri
s.talluri@vu.nl
11A-13, NU Building

XC - project supervised by
Xiaoyu Chu
x.chu@vu.nl
11A-13, NU Building

ZR - project supervised by
z.ren@vu.nl
Zebin Ren
11A-19, NU Building

KD - project supervised by
k.doekemeijer@student.vu.nl
Krijn Doekemeijer
11A-19, NU Building

DN - project supervised by
d.niewenhuis@vu.nl
Dante Niewenhuis
11A-13, NU Building

DB - project supervised by
dr. Daniele Bonetta
d.bonetta@vu.nl
11A-30, NU Building

Team website:

http://atlarge.science

For new students:

https://atlarge-research.com/new_students.html

Main publications:

https://atlarge-research.com/publications.html

This document is available here:

https://docs.google.com/document/d/1k-wTpn1JyAtVg72pIrsOZnGdjrWzEz_CIExGq_6RCLI/edit?usp=sharing

For all possible co-supervisors/second readers, see **https://atlarge-research.com/people.html**

For more details, see [https://atlarge-research.com/new_students.html]. Students we supervise have reached the highest level of achievement and acclaim, winning prestigious awards.

Last edits to this document:

- **2024-07-23 [tdm]:** renamed file for next a.y. Cleaned up and added new topics
- **2024-02-04 [db]:** added more projects
- **2024-17-01 [db]:** added a few new projects
- **2023-12-04 [xc]:** add XC-04 project; update xc topics and project status
- **2023-10-1 [tdm]:** added my topics
- **2023-06-19 [atr]** updating my selection of projects with new ideas
- **2023-02-17: [mj]** Update and add several projects
- **2022-12-05: [zr]** Add NVME emulation project
- **2022-10-26: [ai]** Document is linked in the Overview Bachelor & Master projects 2022-2023 - Department of Computer Science. TODO: Ask the Director of Education to change this system.
- **2022-10-10: [team]** Collective update.
- **2022-10-07: [animesh]** Refreshed my project offers and on-going project status.
- **2022-03-10:** Added AT-msc-11 and 12, a new msc thesis work.
- **2022-01-28: Added AT-msc-10, a new msc thesis work.**
- **2021-09-28: Added Matthijs Jansen, projects TBA**
- **2021-09-24: Addition of LitSurv projects, which will lead to detailed MSc thesis topics -- topics through AI-20. 4 topics for literature study from Animesh.**
- **2021-09-24: Update of AI-1 through AI-17.**
- **[...]**

What's this about?

Here is a list of project ideas, survey work, and course projects that a student can take to work on. Depending upon the scope and skills of each student, these ideas can be narrowed down or expanded in their scope. If in doubt, consider setting up a meeting to discuss in detail with the project supervisor.

Are these all the projects our team has to offer?

Surely not. First, we have formulated extensive vision-documents with many more topics, both topical [R1] and methodological [R2]. Second, the field is truly dynamic and growing, so this document is constantly being worked on -- check periodically to find new projects to work on. Third and last, we appreciate working on a diverse set of problems, so if you have your own idea or question come over and talk to us.

Where can I find hot research topics to work on?

You can follow work in top conferences such as SOSP, OSDI, NSDI, FAST, USENIX ATC, PLDI, SoCC, EuroSys, ICDCS, Middleware, Supercomputing, HPDC. Other conferences of interest might include Systor, Apsys, SIGCOMM, SIGMETRICS, VLDB, SIGMOD, ICAC, ICPE, CCGRID, OOPSLA, PLDI, Ecoop, PPoPP, etc. Ask us for further references. Some vision-scale workshops might be of interest as well -- HotCloud, HotOS, HotStorage, etc. We are and have been program committee chairs for many of these conferences, so we know the topics and can help you get started (and know the community and can help you link to it).

What is the scale of a project?

The scope of each project is typically shown, either as BSc or MSc, but most MSc projects can be scaled down to the BSc level. Plus, MSc projects can be narrowed down for the Individual Systems Practical (XM_405088) [https://studiegids.vu.nl/en/2020-2021/courses/XM_405088].

What support do we offer?

We offer templates and guidelines for all the core tasks and activities that go into a thesis, which you can use directly or customize. For example, we offer a good thesis template and guidelines including Animesh Trivedi's notes, links to various thesis resources as exemplified here, etc. We have literally written the methodological guidelines for the field for several activities [R2], so you will learn from us the state-of-the-art on how to do things.

[R1] Alexandru Iosup, Alexandru Uta, Laurens Versluis, Georgios Andreadis, Erwin Van Eyk, Tim Hegeman, Sacheendra Talluri, Vincent van Beek, Lucian Toader (2018) **Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems**. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf

[R2] Among other methodological articles, Alexandru Iosup, Laurens Versluis, Animesh Trivedi, Erwin Van Eyk, Lucian Toader, Vincent van Beek, Giulia Frascaria, Ahmed Musaafir, Sacheendra Talluri (2019) **The AtLarge Vision on the Design of Distributed Systems and Ecosystems**. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1902.05416.pdf

TDM - projects supervised by Tiziano DE MATTEIS

Topics for BSc and MSc thesis topics (2024/2025):

My research interests are in abstractions, systems, and tools for High-Performance Computing (HPC). More specifically, I work on Spatial/Dataflow Devices (e.g., FPGAs – Field Programmable Gate Arrays –, CGRAs) for HPC, Parallel programming frameworks/models, Energy Efficiency. I'm also interested in everything that is performance-related, and I like to build systems that work. If you are interested in having a chat about the listed topics, or related ones, feel free to drop me an email!

All offered projects are challenging, require good systems programming and knowledge. I assume that at the Bachelor's level you have taken (or you are familiar with the topics of) the Concurrency and Multithreading (X_401031) and at the Master's level you have taken/or you are familiar with at least some of Programming Large-Scale Parallel Systems (XM_40017), Systems Seminar (XM_0122), Programming Multi-core and Many-core Systems (XMU_40018), Accelerator-Centric Computing Ecosystems (XM_0171)

Note: Related literature study subjects are available for MSc thesis proposals.

Productive programming of AMD/Xilinx Versal Devices

The Versal Adaptive Compute Acceleration Platforms (ACAP) is a programmable heterogeneous device that combines general-purpose CPU cores, programmable logic (PL), and specialized AI Engine (AIE) compute processors in the same System-on-Chip. Versal devices are being shipped in the market and are being included in commercially available processors (https://www.amd.com/en/technologies/xdna.html), but how to efficiently program and exploit them are still open questions.

In this context, multiple theses are available covering (but not limited to) topics such as:

- Explore the capability of AI-Engines in commodity processors (https://riallto.ai/index.html, https://www.amd.com/en/technologies/xdna.html)
- how to efficiently map stencil computations on such devices, develop interprocessor communication libraries etc.
- Reduce and AllReduce Primitives on ACAP devices
- Porting specific workloads (e.g., radio-astronomy, in collaboration with ASTRON)
- Evaluate productivity of machine learning framework for running specific workloads (in collaboration with ASTRON)
- Implementing Streaming Analytics operators
- Implementing Linear Algebra routines (BLAS – ongoing project here: https://github.com/atlarge-research/AIE-BLAS)
- ….

Level: MSc, or motivated BSc

Relevance: With this project, you will get significant experience in spatial architectures and in programming one of the prominent, and commercially available, implementation of spatial computer. The knowledge you acquire will be transferable to other spatial architectures, e.g., CGRAs like those from Cerebras. In addition, you will get the opportunity to contribute to the community with publicly open-source and reusable libraries.

Desired Requirements:

- Good programming skills (C/C++).
- Some experience with build system (e.g., [C]Make).
- Some knowledge of parallel programming and parallel architectures

References:

- https://www.xilinx.com/products/silicon-devices/acap/versal-ai-core.html
- Xilinx. "Versal ACAP AI Engine Programming Environment User Guide" https://docs.xilinx.com/r/en-US/ug1076-ai-engine-environment

FPGA vs GPU: the AI battle

FPGAs could be a more energy efficient alternative to GPUs, especially when considering Machine Learning inference tasks. In this thesis, we want to evaluate their performance (also in terms of energy consumption), and compare them with the one achieved by GPUs. For this project we will use Pytorch, and Vitis AI, a tool to conveniently port your ML model to FPGAs.

Level: MSc

Evaluating a commerical disaggregation platform

Hardware disaggregation is becoming a popular solution in data centers to seamingless share resources (e.g., compute, memory) between servers, reducing the total infrastructure and operational costs. In this project we want to analyze, evaluate and discuss the potential of the Liqid composable infrastructure. This infrastructure can deliver scalable architecture built from pools of disaggregated resources, such as compute, networking, storage, GPU, FPGA, and memory devices.

Note: this project will leverage the SURF Experimental Platform Technologies (https://www.surf.nl/en/etp). Plannin ahead is necessary to get access to the infrastructure.

Level: MSc

Exploring Spatial Architectures

Spatial (or Reconfigurable Dataflow) architectures are being offered today as a way to overcome some of the limitations of classical von Neuman devices (CPUs and GPUs). Many of the commercially available solutions are being offered today as Machine-Learning accelerators, but how to use them productively for more generic scientific computing is yet to be understood. In this research line, we want to explore these dimensions by:

- Benchmarking such devices
- Understanding their programming environment
- Implementing specific applications
- Developing reusable libraries
- …

These projects can be offered as MSc thesis or Research Projects (6/12-CFUs)

Currently available:

- Programming AMD/Xilinx Versal Devices (see previous point)
- Benchmark NextSilicon Maverick: this can be done under the SURF experimental platform (https://www.surf.nl/en/news/surf-partners-with-nextsilicon-for-faster-and-energy-efficient-research-workflows). Be aware, this requires planning time and it is subject to SURF availability.

Reduced precision algebra

Traditionally, scientific computing used double-precision floating point formats (usually 64 bits) to carry on computations. While in some cases also single-precision (32 bits) can be a viable options, modern hardware (GPU, ML accelerators) is increasingly supporting in hardware half-precision formats (e.g., BFloat16), or even fixed point arithmetics (e.g., INT8). In this project we want to first review current approaches and how these are supported in software. Then, we would like to understand what are the performance and power benefit of using a reduced precision format, and what are the tradeoff (e.g., in terms of accuracy of the produced results). We will mainly target CPU, but the work can also be extended to consider GPUs.

Level: MSc, BSc

Relevance:
Reduced precision formats are quite popular today thanks do their wide adoption in Machine Learning workload. With this project you will get to understand their fundamentals, pros and cons. You will also get to know/use benchmarking methods (e.g, to measure power consumption) and parallel performance models (e.g., roofline models).

Desired Requirements:

- Experience with C/C++ programming
- Some experience with linear algebra libraries (preferably)
- Some experience with GPU programming (preferably)

References:

- For an overview of the impact of reduced/mixed precision formats in linear algebra please refer to: Higham, N., & Mary, T. (2022). Mixed precision algorithms in numerical linear algebra. Acta Numerica, 31, 347-414. doi:10.1017/S0962492922000022
  Note: this paper focuses on mathematical properties of reduced precision formats, while in this project we want to better understand their performance/power benefits (aka, don't be scared by the maths you will find there!)

Sustainable task graph scheduling

Task graphs are a way of representing computations. The computation is decomposed in smaller components called tasks (the nodes in the graph). Tasks are connected by precedence constraints (aka data dependencies, the edge in the graph). Scheduling a task graph requires deciding on which compute resource execute each task, making sure to respect the precedence constraints.

Traditionally, performance has been the key metric to evaluate a scheduling strategies. But with sustainability being more and more advocated also in the IT sector, it is becoming more relevant to also consider it as a key objective of a scheduling strategy. The goal of this project is to explore this performance vs. sustainability trade off, designing, analysing and evaluating simple sustainable scheduling strategies (e.g., inspired to well known classical scheduling algorithms such as HEFT).

This project will be developed together with Dante Niewenhuis as daily advisor, and OpenDC (a data center simulator developed in the lab) can be used for the evaluation

Level: BSc/MSc

Desired Requirements:
- Knowledgable of graph and some basic graph theory
- Basic Programming experience
- OpenDC knowledge (preferably and only if the evaluation target will be OpenDC)

Sparse matrix formats and sparse linear algebra on reconfigurable accelerators

Sparse Matrices are widely used today in scientific applications (e.g. solving linear systems of equations) and in machine learning. Sparse Matrices can be represented using different formats, that differ in terms of memory access layout, and, therefore in the total execution time of the sparse matrix operations (e.g., SpMV, sparse matrix-vector). In this thesis we want to **1)** review the different sparse matrix format available today (suitable for Bachelor thesis, targeting CPU), and **2)** implement basic building blocks for the execution of sparse matrix operations on FPGAs (Master thesis).

[More details to be added soon/discussed]

Level: MSc (objective 1 and 2)

References:

- Olivia Hsu, Maxwell Strange, Ritvik Sharma, Jaeyeon Won, Kunle Olukotun, Joel S. Emer, Mark A. Horowitz, and Fredrik Kjølstad. 2023. The Sparse Abstract Machine. In Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3 (ASPLOS 2023). https://arxiv.org/abs/2208.14610

GPU-Accelerated Parallelization of Quantum Circuit Simulation

Objective: The project's core objective is to evaluate the practical implications of parallelization of quantum circuits, with a primary focus on leveraging GPU processing. The study also aims to assess the generalizability of this acceleration across various instances within quantum computing applications. It will start with a benchmarking phase of currently available solution, followed by an analysis of achieved results and design of improved solution.

Level: MSc or Resarch Project (only the benchmark and analysis part)
This is a joint project together with University of Amsterdam-QuSoft and the Chemistry department.

Desired Requirements:
- Experience with C/C++ programming
- Some experience with linear algebra (preferably)
- Some experience with GPU programming (preferably)

Task Graph Processing on Reconfigurable Devices

TBD

Misc/Your idea

I am generally interested in supervising projects related to HPC, Accelerator, Parallelism, Energy. So you are more encouraged to come up with your idea. Some examples:
- Parallelization of computations on FPGA/GPU/CPU/Novel architectures
- Data plane acceleration (e.g. Network processing) using FPGA or AMD Versal devices
- Implementation of concurrent data structures (for instance, shared queues on GPU…)
- Artifact reproducibility and expansion (for BSc only)
- …

AI - projects supervised by Alexandru IOSUP

Unless otherwise noted, all projects in this set are for a 30-EC MSc project accompanied by a 6-EC Literature Study; they can be scaled down toward a BSc project. Also, all of them can lead to publications. Some can be done in collaboration with industry (ask) or with the international organization SPEC (noted in the description of the work).

All offered projects are challenging. Some require good systems programming. I assume that at the Bachelor's level you have taken and done well in Computer Organization and Computer Networks, and preferably also Advanced Network Programming, and at the Master's level you have done well in Distributed Systems and preferably also Systems Storage. If you have not, let's discuss how to prepare.

Active MSc projects with a literature study (2022-2023):

1. **MSc-thesis-1** Reproducibility in distributed systems
    a. Daily supervision: shared with Jesse Donkervliet
    b. Assigned : Ean-Dan Tjon-Joek-Tjien
2. **MSC-thesis-2** Scalable platform for procedural game-content generation
    a. Daily supervision: shared with Jesse Donkervliet
    b. Assigned: Misha Rigot
3. **MSC-thesis-3** Dynamic Area of Simulation in Minecraft-like Games
    a. Daily supervision: shared with Jesse Donkervliet
    b. Assigned: Jerrit Eickhoff
4. **MSC-thesis-4** Extending classic scheduling interfaces to improve performance and energy-efficiency in datacenters
    a. Daily supervision: shared with Sacheendra Talluri
    b. Assigned: Aratz Manterola-Lasa
5. **MSC-thesis-5** Analysis of the impact of combined operational techniques on datacenter performance and energy-efficiency
    a. Assigned: Tim Pelle
6. **MSc-thesis-6** Generalized workflow language based on CWL
    a. Daily supervision: shared with Michael R. Crusoe
    b. Assigned: Mihai Popescu
7. **MSc-thesis-7** Plantenna - Serverless Edge Computing for Precision Agriculture
    a. Daily supervision: shared with Matthijs Jansen
    b. Assigned:
8. **MSc-thesis-8** Energy-efficient graph processing
    a. Assigned: Paul Puttbach

Active BSc projects (2022-2023):

1. **BSc-thesis-1:** Serverless Persistent Multiplayer Worlds in Minecraft-like Games
    a. Assigned: Victor Gavrilovici
2. **BSc-thesis-2**: Serverless Multiplayer Minigames in Minecraft-like Games
    a. Assigned: Sven Lankester
3. **BSc-thesis-3**: Analysis of energy-efficiency and performance in scientific workflow execution in geo-distributed datacenters
    a. Assigned: Radu Apsan

Active HP projects (2022-2023):

1. David Breitling
2. Tiziano Coroneo
3. Paul Ellsiepen (+ BSc)
4. Dide Poyraz
5. Tudor Roman
6. Lennart Schulz
7. Răzvan Wist

AI-1. Protocol for reproducible experiments in cloud and serverless computing.

Reproducibility of results is a core tenet of any scientific field [1], and ever more so in fields that develop critical technology, for example, in computer science [2] and its application to cloud (and serverless) computing. Like other leaders in the field [3], we envision fostering a domain where everything we develop is tested and benchmarked, reproducibly. Together with the global organization SPEC [4], our group has developed the first set of methodological principles for reproducible experiments in cloud computing, both real-world [5] and simulation [6]. Your thesis will extend this work conceptually with new principles and an overall protocol, and technically with the necessary instruments to conduct experiments and test the degree of reproducibility for quantifiable principles.

[1] M. Baker, “Is there a reproducibility crisis?” Nature, vol. 533, no. 7604, pp. 452–454, 2016.
[2] P. J. Denning, “The science in computer science,” Commun. ACM, vol. 56, pp. 35–38, 2013.
[3] D. G. Feitelson, “Experimental computer science: The need for a cultural change,” The Hebrew University of Jerusalem, Tech. Rep., 2005.
[4] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[5] Alessandro Vittorio Papadopoulos, Laurens Versluis, André Bauer, Nikolas Herbst, Jóakim von Kistowski, Ahmed Ali-Eldin, Cristina Abad, J. Nelson Amaral, Petr Tuma, and Alexandru Iosup (2019) Methodological Principles for Reproducible Performance Evaluation in Cloud Computing - A SPEC Research Technical Report. SPEC RG Technical Report SPEC-RG-2019-03, v.1.0. April 2019. [Online] Available at: http://bit.ly/ReproducibleCloudExperiments
[6] Fabian Mastenbroek, Georgios Andreadis, Soufiane Jounaid, Wenchen Lai, Jacob Burley, Jaro Bosch, Erwin Van Eyk, Laurens Versluis, Vincent van Beek, Alexandru Iosup (2021) OpenDC 2.0: Convenient Modeling and Simulation of Emerging Technologies in Cloud Datacenters. CCGRID 2021: 455-464.

AI-1b. Protocol for reproducible experiments in big data. Reproducibility of results is a core tenet of any scientific field [1], and ever more so in fields that develop critical technology, for example, in computer science [2] and its application to big data. Like other leaders in the field [3], we envision fostering a domain where everything we develop is tested and benchmarked, reproducibly. Together with the global organization SPEC [4], our group has developed the first set of methodological principles for reproducible experiments in cloud computing, both real-world [5] and simulation [6]. Your thesis will extend this work conceptually toward the principles and the overall protocol needed for big data, and technically with the necessary instruments to conduct experiments and test the degree of reproducibility for quantifiable principles.

[1] M. Baker, “Is there a reproducibility crisis?” Nature, vol. 533, no. 7604, pp. 452–454, 2016.
[2] P. J. Denning, “The science in computer science,” Commun. ACM, vol. 56, pp. 35–38, 2013.

[3] D. G. Feitelson, “Experimental computer science: The need for a cultural change,” The Hebrew University of Jerusalem, Tech. Rep., 2005.
[4] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[5] Alessandro Vittorio Papadopoulos, Laurens Versluis, André Bauer, Nikolas Herbst, Jóakim von Kistowski, Ahmed Ali-Eldin, Cristina Abad, J. Nelson Amaral, Petr Tuma, and Alexandru Iosup (2019) Methodological Principles for Reproducible Performance Evaluation in Cloud Computing - A SPEC Research Technical Report. SPEC RG Technical Report SPEC-RG-2019-03, v.1.0. April 2019. [Online] Available at: http://bit.ly/ReproducibleCloudExperiments
[6] Fabian Mastenbroek, Georgios Andreadis, Soufiane Jounaid, Wenchen Lai, Jacob Burley, Jaro Bosch, Erwin Van Eyk, Laurens Versluis, Vincent van Beek, Alexandru Iosup (2021) OpenDC 2.0: Convenient Modeling and Simulation of Emerging Technologies in Cloud Datacenters. CCGRID 2021: 455-464.

AI-1c. Collaborative protocol for reproducible experiments in distributed systems.
Reproducibility of results is a core tenet of any scientific field [1], and ever more so in fields that develop critical technology, for example, in computer science [2] and its large field of distributed systems. Like other leaders in the field [3], we envision fostering a domain where everything we develop is tested and benchmarked, reproducibly. Together with the global organization SPEC [4], our group has developed the first set of methodological principles for reproducible experiments in cloud computing, both real-world [5] and simulation [6]. However, we have not conducted collaborative experiments to test across multiple institutions the reproducibility of claims made in cloud computing, and more generally in distributed systems. Your thesis will help address this situation, and thus provide a pragmatic contribution to enable reproducibility in distributed systems.

[1] M. Baker, “Is there a reproducibility crisis?” Nature, vol. 533, no. 7604, pp. 452–454, 2016.
[2] P. J. Denning, “The science in computer science,” Commun. ACM, vol. 56, pp. 35–38, 2013.
[3] D. G. Feitelson, “Experimental computer science: The need for a cultural change,” The Hebrew University of Jerusalem, Tech. Rep., 2005.
[4] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[5] Alessandro Vittorio Papadopoulos, Laurens Versluis, André Bauer, Nikolas Herbst, Jóakim von Kistowski, Ahmed Ali-Eldin, Cristina Abad, J. Nelson Amaral, Petr Tuma, and Alexandru Iosup (2019) Methodological Principles for Reproducible Performance Evaluation in Cloud Computing - A SPEC Research Technical Report. SPEC RG Technical Report SPEC-RG-2019-03, v.1.0. April 2019. [Online] Available at: http://bit.ly/ReproducibleCloudExperiments
[6] Fabian Mastenbroek, Georgios Andreadis, Soufiane Jounaid, Wenchen Lai, Jacob Burley, Jaro Bosch, Erwin Van Eyk, Laurens Versluis, Vincent van Beek, Alexandru Iosup (2021) OpenDC 2.0: Convenient Modeling and Simulation of Emerging Technologies in Cloud Datacenters. CCGRID 2021: 455-464.

AI-1d. Protocol for reproducible experiments in scheduling for large-scale clusters and datacenters.
Reproducibility of results is a core tenet of any scientific field [1], and ever more so in fields that develop critical technology, for example, in computer science [2] and its application to big data. Like other leaders in the field [3], we envision fostering a domain where everything we develop is tested and benchmarked, reproducibly. Together with the global organization SPEC [4], our group has developed the first set of methodological principles for reproducible experiments in cloud computing, both real-world [5] and simulation [6]. Your thesis will extend this work conceptually toward the principles and the overall protocol needed for scheduling for large-scale clusters and datacenters, and technically with the necessary instruments to conduct experiments and test the degree of reproducibility for quantifiable principles. Among other goals, you could design a global competition for such schedulers, e.g., around OpenDC as simulation platform (see directly AI-8).

[1] M. Baker, “Is there a reproducibility crisis?” Nature, vol. 533, no. 7604, pp. 452–454, 2016.
[2] P. J. Denning, “The science in computer science,” Commun. ACM, vol. 56, pp. 35–38, 2013.

[3] D. G. Feitelson, “Experimental computer science: The need for a cultural change,” The Hebrew University of Jerusalem, Tech. Rep., 2005.
[4] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[5] Alessandro Vittorio Papadopoulos, Laurens Versluis, André Bauer, Nikolas Herbst, Jóakim von Kistowski, Ahmed Ali-Eldin, Cristina Abad, J. Nelson Amaral, Petr Tuma, and Alexandru Iosup (2019) Methodological Principles for Reproducible Performance Evaluation in Cloud Computing - A SPEC Research Technical Report. SPEC RG Technical Report SPEC-RG-2019-03, v.1.0. April 2019. [Online] Available at: http://bit.ly/ReproducibleCloudExperiments
[6] Fabian Mastenbroek, Georgios Andreadis, Soufiane Jounaid, Wenchen Lai, Jacob Burley, Jaro Bosch, Erwin Van Eyk, Laurens Versluis, Vincent van Beek, Alexandru Iosup (2021) OpenDC 2.0: Convenient Modeling and Simulation of Emerging Technologies in Cloud Datacenters. CCGRID 2021: 455-464.

AI-2. Understanding distributed systems and ecosystems.

The modern lifestyle depends on computer ecosystems. We engage increasingly with each other, with governance, and with the Digital Economy through diverse computer ecosystems comprised of globally distributed systems, developed and operated by diverse organizations, interoperated across diverse legal and administrative boundaries. These computer ecosystems create economies of scale, and underpin participation and innovation in the knowledge-based society: for example, in the European Union, information and communication technology (ICT), for which all services are migrating to distributed systems and ecosystems, accounts for nearly 5% of the economy and accounts for nearly 50% of productivity growth. Correspondingly, ICT receives about 25% of all business R&D funding and is at the core of EU’s H2020 programme. In this thesis, you will help address a deceivingly simple, yet fundamental question: *What are distributed systems and ecosystems?* Starting from our radical new take on such computer systems [1], you will help define these terms and identify the conceptual gray-area that separates them, catalog typical system structures and programming models, and design a method to compare their fundamental capabilities and (non-functional) properties.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf

AI-3. OpenDC: Exploring the design space for resource management and scheduling in the cloud-edge continuum.

ICT infrastructure is important for the digital society [1, Section 6.1]. Key to ICT operation is its scheduler and resource manager, which takes on behalf of users and datacenter-engineers decisions about workload and resources. Our group has created a novel reference architecture for datacenter scheduling [2], to help manage the staggering complexity and performance challenges that appear in this context. (This work got the field excited and led to our work being accepted in the prestigious SC flagship conference.) In this thesis, you will use this architecture to explore the design space for scheduling and resource management, focusing on the cloud-edge continuum. You will learn about schedulers and their components, and design, implement, and validate a simulation-based instrument to estimate their complex operations and interactions in the datacenter. The instrument you develop will be based on our already existing simulation toolkit, OpenDC [https://opendc.org] [3,4]. Validation experiments will take place in our DAS infrastructure and using public-cloud resources.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf
[2] Andreadis et al. (2018) A reference architecture for datacenter scheduling: design, validation, and experiments. SC 2018: 37:1-15. [Online] Also available as extended technical report: https://arxiv.org/pdf/1808.04224.pdf
[3] Iosup et al. (2017) The OpenDC Vision: Towards Collaborative Datacenter Simulation and Exploration for Everybody. ISPDC 2017: 85-94. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/opendc-vision17ispdc_cr.pdf
[4] Alexandru Iosup, Georgios Andreadis, Vincent van Beek, Matthijs Bijman, Erwin Van Eyk, Mihai Neacsu, Leon Overweel, Sacheendra Talluri, Laurens Versluis, Maaike Visser (2021) The OpenDC Vision: Towards Collaborative Datacenter Simulation and Exploration for Everybody. ISPDC 2017: 85-94
[5] Bal et al. (2016) A Medium-Scale Distributed System for Computer Science Research: Infrastructure for the Long Term. IEEE Computer 49(5): 54-63.

AI-4. A reference architecture for serverless operations in the cloud-edge continuum.

ICT infrastructure is vital for the digital society. Recently, datacenters, which are the core computational workhorse of ICT infrastructure, have started to host serverless operations, offered as service to clients as Function-as-a-Service (FaaS) [1, Section 6.5 of the technical report, "The Future of Apps: Serverless, Service- and Workflow-based Ecosystems"]. Together with the global organization SPEC [2], our group has developed an understanding of what serverless and FaaS are -- for example, FaaS offers desirable properties as an operational model: focus on the application logic with minimal operational logic, event-driven operation that corresponds to an always-connected world, and granular billing and thus unprecedented efficiency. [3] In this thesis, you will help extend the reference architecture for FaaS cloud computing conceptually, by adding to it not only new elements that have emerged in the past couple of years (as the pace of innovation in this domain is break-neck), but also tracing the evolution of each component [4] and thus helping to explain why each part of the stack is there. Further, you will design, implement, and validate a simulation-based instrument to estimate the complex operations and interactions in the FaaS ecosystem.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf
[2] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[3] van Eyk et al. (2018) A SPEC RG Cloud Group's Vision on the Performance Challenges of FaaS Cloud Architectures. ICPE. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/spec-rg-cloud-serverless-performance-challenges-icpe-2018.pdf
[4] van Eyk et al. (2018) Serverless is More: From PaaS to Present Cloud Computing. IEEE Internet Computing 22(5): 8-17. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/serverless-history-now-future18ieeeic.pdf

AI-4b. A reference architecture for AI/ML/DL in the datacenter. Datacenter infrastructure is important for the digital society. Stakeholders across industry, government, and academia employ diverse cloud services hosted by datacenter infrastructure, and expect services to be reliable, high speed, and low cost. In turn, datacenter operators must maintain efficient operation at unprecedented scale, combining systems with unprecedented heterogeneity (diversity). This enormous complexity has risen sharply in the past decade, making many of the existing designs and tools obsolete (in this field, every order of magnitude leads to a different problem for the engineer). The Big Tech companies employ armies or high-quality engineers and scramble to find pragmatic approaches that can help them cope another day [1,2]. To conquer complexity, our approach is to focus on an architectural approach, based on reference architectures -- the pattern for all datacenters we can envision. Our group has proposed a novel reference architecture for datacenters [3, Section 6.1]. In this thesis, you will extend this architecture conceptually by adding to it the (many) systems and ecosystems supporting artificial intelligence, machine learning, and deep learning applications, and by designing, implementing, and validating a simulation-based instrument to estimate their complex operations and interactions in the datacenter.

[1] Beyer et al., Site Reliability Engineering [at Google]. O´Reilly, 2016.
[2] Beyer et al., Site Reliability Workbook. O´Reilly, 2018.
[3] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf

AI-4c. A reference architecture for the banking ecosystem. Banking is a vital component of modern industries, especially for knowledge-based societies: banking facilitates transferring, depositing, and lending capital. As a heavily regulated industry, banking has been traditionally slow to uptake novel technology and often operates with multi-decade legacy ICT systems. However, since 2008 the industry has seen a significant change, combining two contrary directions: (i) more regulation in terms of increased liability and lower tolerance for risk, with (ii) increased openness of the market aiming to provide better service for (retail-)consumers. For example, new regulation appeared in the banking and financial industry, in response to the 2007-2008 financial crisis, including the Basel series of stress-tests. To open the market, since 2008 a Single Euro Payments Area (SEPA) helps harmonizing bank transactions in Europe. To further open the market, in 2015 but with effective implementation in 2018, the European Union has passed into law the second Payment Services Directive (PSD2), which opens up the retail-banking market for more service providers (e.g., Mint for account information), fintech companies (e.g., Adyen and Klarna for payments, Tink for budget management, OurCrowd for crowdfunding), and even the traditional consumer-facing brands (e.g., Google, Apple, telcos, who can combine online retail with banking functionality). We thus see the future of banking as becoming increasingly dependent on complex ecosystems [1, Section 6.4 of the technical report, “The Future of Banking”]. But how to understand and improve this ecosystem? In this thesis, you will create the first reference architecture for banking ecosystems -- a vital instrument that structures our understanding of the tens to hundreds of distributed systems integrated into a dynamic ecosystem. Further, you will design, implement, and validate a simulation-based instrument to estimate the complex operations and interactions in this ecosystem.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf

AI-4d. A reference architecture for precision agriculture. We thus see the future of agriculture as becoming increasingly dependent on complex ecosystems [1]. But how to understand and improve this ecosystem? In this thesis, you will create the first reference architecture for agriculture ecosystems. Further, you will design,

implement, and validate a real-world and/or simulation-based instrument to estimate the complex operations and interactions in this ecosystem.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf

AI-5. The Distributed Systems Memex.

The implications of archiving large amounts of daily information for scientific and societal purposes are clear since at least the 1940s, when Vannevar Bush defined the concept of the personal memex as an individual's device for storing and accessing privately all information and communication involving that individual. Among the benefits are learning about and eradicating humankind diseases, enabling human beings more creative and thought-related time by eliminating tasks that can be automated, etc. Similarly, we posit that a Distributed Systems Memex, archiving large amounts of operational traces collected from the many distributed systems and ecosystems that underpin our society, would be beneficial for understanding and tuning today's distributed systems, and for designing better distributed systems in the future. Our group has pioneered building the Distributed Systems Memex from scratch, and in particular has established five archives for workload and operational traces [1-5]. All these archives have become de facto standards in our community, helping many researchers and engineers. Your thesis will help design and prototype the entire Distributed Systems Memex. We are about to release the eighth, the Cloud Failure Archive.

[1] Iosup et al. (2008) The Grid Workloads Archive. Future Generation Comp. Syst. 24(7):672-686.
[2] Kondo et al. (2010) The Failure Trace Archive: Enabling Comparative Analysis of Failures in Diverse Distributed Systems. CCGRID 2010: 398-407. Best paper award.
[3] Zhang et al. (2010) The peer-to-peer trace archive: design and comparative trace analysis. In Proceedings of the ACM CoNEXT Student Workshop (CoNEXT '10 Student Workshop). Article 21.
[4] Guo and Iosup (2012) The Game Trace Archive. NetGames.
[5] Versluis et al. (2021) The Workflow Trace Archive. TPDS.

AI-6. KnowDS: The Catalog of Distributed Systems Knowledge.

Distributed systems have existed since the 1950s, and distributed ecosystems are now the digital factories of our economy. Over 15,000 new articles on topics in distributed systems and ecosystems appear every year in peer-reviewed venues. This raises a seemingly simple but fundamental question for the field: What we know and how we know it? The objective of this thesis is to find a process, method, or technique to address this question, thus overcoming numerous challenges in understanding systematically the field -- What are the key topics and why? What are the key approaches to address them and how much they are in use? What are the known unknowns? Can we identify new unknowns? Etc. The basis of this work will be to contrast a novel quantitative approach to the qualitative approaches that have appeared in technology since the seminal study of Vincenti [1].

[1] W. G. Vincenti, What Engineers Know and How They Know It. JohnS Hopkins University Press, 1990.

AI-7. Generalized portfolio scheduling for distributed systems.

Schedulers make, take, and enforce decisions that balance helping clients with optimizing resource use, already at scales and frequencies that exceed by far what a team of human experts can do. Thus, scheduling is at the core of every modern distributed system. Scheduling has significant monetary and sustainability implications in modern datacenters, cloud computing, and big data. However desirable good scheduling is, currently good scheduling relies primarily by creating smart but specialized algorithms; when the conditions for which these algorithms change, the algorithms perform poorly or can even create adverse conditions in the system. Instead, we have pioneered portfolio scheduling [1,2,3], an approach in which the system can automatically and dynamically select the active scheduling algorithm (policy), from the set with which the portfolio is configured. We have built portfolio schedulers and evaluated them in simulation and in complex real-world scenarios. We found portfolio scheduling to be flexible across a large variety of domains [4, Section 6.6], and our work over the past 5 years has already led to numerous successes and to identifying new challenges. In this thesis, you will address one such challenge, starting from the seemingly simple question -- *What are the fundamental limitations to portfolio scheduling in general distributed systems?*

[1] Deng et al. (2013) Exploring portfolio scheduling for long-term execution of scientific workloads in IaaS clouds. SC 2013: 55:1-55:12. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/2013-deng-exploring.pdf
[2] Ma et al. (2017) Ananke: A Q-Learning-Based Portfolio Scheduler for Complex Industrial Workflows. ICAC 2017: 227-232.
[3] Voinea et al. (2018) POSUM: A Portfolio Scheduler for MapReduce Workloads. BigData 2018: 351-357.
[4] Iosup et al. (2019) The AtLarge Vision on the Design of Distributed Systems and Ecosystems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1902.05416.pdf

AI-8. ScheDC: A global competition and education platform for datacenter resource management and scheduling.

Datacenter infrastructure is vital for the digital society. At the core of every datacenter is a (distributed) scheduler, which makes, takes, and enforces decisions that balance helping clients with optimizing resource use, at scales and frequencies that exceed by far what a team of human experts can do. Consequently, datacenter schedulers tend to become complex, barely accessible systems, and few people actually get the opportunity to familiarize themselves with the scheduling process in real datacenters. This is where a simulation platform such as OpenDC [1] can help: it can provide the opportunity for low-barrier experimentation with datacenter scheduling concepts. Your thesis will help create a global competition and education platform for scheduling in datacenters, with OpenDC at the center. You will learn about the modular and flexible reference architecture for datacenter scheduling proposed in our recent work [2], and extend it and the simulation toolkit offered by OpenDC to support many common scenarios and problems that appear in datacenter scheduling.

[1] Iosup et al. (2017) The OpenDC Vision: Towards Collaborative Datacenter Simulation and Exploration for Everybody. ISPDC 2017: 85-94. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/opendc-vision17ispdc_cr.pdf
[2] Andreadis et al. (2018) A reference architecture for datacenter scheduling: design, validation, and experiments. SC 2018: 37:1-15. [Online] Also available as extended technical report: https://arxiv.org/pdf/1808.04224.pdf

AI-9. TraceIT: A comprehensive benchmark for serverless platforms.

Recently, datacenters have started to host serverless operations, offered as service to clients as Function-as-a-Service (FaaS) [1, Section 6.5 of the technical report, "The Future of Apps: Serverless, Service- and Workflow-based Ecosystems"]. Together with the global organization SPEC [2], our group has developed an understanding of what serverless and FaaS are -- for example, FaaS offers desirable properties as an operational model: focus on the application logic with minimal operational logic, event-driven operation that corresponds to an always-connected world, and granular billing and thus unprecedented efficiency. [3] We have analyzed and compared qualitatively over 50 serverless platforms, and found significant differences between them (under submission, available upon request). However, until now the community has not created a comprehensive benchmark for serverless platforms, and consequently there is little quantitative knowledge about their operation and performance. Your thesis will change both. You will design, implement, and validate a benchmark for serverless platforms, and use it to compare several of the state-of-the-art competitors in this field.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf
[2] SPEC RG Cloud https://research.spec.org/working-groups/rg-cloud.html
[3] van Eyk et al. (2018) A SPEC RG Cloud Group's Vision on the Performance Challenges of FaaS Cloud Architectures. ICPE. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/spec-rg-cloud-serverless-performance-challenges-icpe-2018.pdf

AI-10. How do the people of distributed systems think?

We live in the Golden Age of Distributed Systems and Ecosystems, one in which such systems underpin almost every activity vital to or popular in our society and industry. The design principles of such systems have evolved greatly over the past few decades, but are rooted in the significant intellectual advances made in the 1990s. The leaders of the 1990s generation are now recently retired or about to retire so, if we want to preserve the trove of expertise they have accumulated and developed, we must act now. In this thesis, you will combine social sciences and computer sciences methods to conduct interviews with experts in the field, to do (graph) analysis of their professional education and affiliation, and overall to capture their knowledge and professional genealogy. You will develop understanding, map real-world processes, and understand research and practice. You will collect and use primary evidence, including interviews with prominent and award-winning people in the field. You will also develop the tools to conduct this work systematically, and to query it programmatically. This work is inspired by similar projects in mathematics [1] and by the Mathematics Genealogy Project [2].

[1] Donald J. Albers (2011) Fascinating Mathematical People: Interviews and Memoirs. Princeton University Press. ISBN13 9780691148298.

[2] Alexandru Iosup's (incomplete) entry on the Mathematics Genealogy Project: https://www.genealogy.math.ndsu.nodak.edu/id.php?id=131962

~~**AI-11. CUPs 2.0: A Pattern Language for Datacenters and their Applications.** Datacenters are at the core of our society. However, their use as cloud services and their applications remain difficult to describe, and in particular the discussion between experts and non-experts is cumbersome, and the discussion between experts and experts leaves many aspects under-specified or omitted. Our pioneering work with the SPEC RG Cloud group has resulted in the definition of a language for specifying Cloud Usage Patterns (CUP) [1]. Your thesis will extend this work to address many emerging issues: Which design patterns appear at the intersection of datacenter systems and their applications (as cloud services)? How to express them? How to employ them into larger designs and ecosystems? Etc. You will design the language extensions, and an instrument that can help with deploying applications using as input only the basic specification. You will further quantify the performance and cost losses due to inexpert specification.~~

~~[1] Aleksandar Milenkoski, Alexandru Iosup, Samuel Kounev, Kai Sachs, Diane E. Mularz (MITRE), Jonathan A. Curtiss (MITRE), Jason Ding (Salesforce), Florian Rosenberg (IBM), Piotr Rygielski (2018) CUP: A Formalism for Expressing Cloud Usage Patterns for Experts and Non-Experts. IEEE Cloud Computing 5(3): 65-76.~~

<MagnaData: social scheduling>
<MagnaData: performance comparison of resource managers>

AI-12. Opencraft: Towards Scalable Minecraft-like Environments.

Minecraft is a block-based online virtual environment with over 120M active users, that was developed as an indie project and later was bought for >$2.5 billion by Microsoft. Today, the Minecraft ecosystem includes tens of thousands of community-developed mods, a thriving emulation and server-replica community, and -- the highest level of flattery -- hundreds of Minecraft-like commercial games and education environments. Our Opencraft project aims to make Minecraft-like environments scalable. To do this, we adapt existing and invent new distributed systems concepts and designs [1,2], such as dynamic conits, elastic hosting for online games, etc., and we adopt rigorous scientific and experimental computer science methods [3,4]. For example, ours is one of the few experimental studies in distributed systems to have received the coveted ACM reproducibility badge for Artifacts Evaluated Functional [5,6]. Your thesis will continue this tradition, by further scaling Opencraft and/or making multi-game hosting more elastic.

[1] Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1802.05465.pdf
[2] Iosup et al. (2019) The AtLarge Vision on the Design of Distributed Systems and Ecosystems. ICDCS. [Online] Also available as extended technical report: https://arxiv.org/pdf/1902.05416.pdf
[3] D. G. Feitelson, "Experimental computer science: The need for a cultural change," The Hebrew University of Jerusalem, Tech. Rep., 2005.
[4] van der Sar et al. (2019) Yardstick: A Benchmark for Minecraft-like Services. ICPE 2019: 243-253. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/jvdsar-yardstick-benchmark-icpe-2019.pdf
[5] Jesse Donkervliet, Jim Cuijpers, Alexandru Iosup (2021) Dyconits: Scaling Minecraft-like Services through Dynamically Managed Inconsistency ICDCS 2021. https://atlarge-research.com/pdfs/icdcs21-dyconit-paper.pdf

[6] Jerrit Eickhoff, Jesse Donkervliet, Alexandru Iosup (2022) Meterstick: Benchmarking Performance Variability in Cloud and Self-hosted Minecraft-like Games. ISPASS 2022: 147-149

AI-12b. Benchmarking Minecraft-like Environments. Minecraft is a block-based online virtual environment with over 50M active users, that was developed as an indie project and later was bought for >$2.5 billion by Microsoft. Today, the Minecraft ecosystem includes tens of thousands of community-developed mods, a thriving emulation and server-replica community, and -- the highest level of flattery -- hundreds of Minecraft-like commercial games and education environments. Our recent work on benchmarking Minecraft-like systems has been highly appreciated by the community (18% acceptance ratio) [1] and provided one of the few experimental studies in distributed systems to have received the coveted ACM reproducibility badge for Artifacts Evaluated Functional [2]. Your thesis will extend the benchmark with several key capabilities: benchmarking under realistic and diverse mobility and activity models for avatars (e.g., adding the SAMOVAR [3,4] and related models), and benchmarking under computationally expensive aspects of Minecraft-like environments (e.g., complex control with redstone). You will conduct experiments in both our DAS [5,6] and public-cloud infrastructure.

[1] van der Sar et al. (2019) Yardstick: A Benchmark for Minecraft-like Services. ICPE 2019: 243-253. [Online] Also available on our team's website: https://atlarge-research.com/pdfs/jvdsar-yardstick-benchmark-icpe-2019.pdf
[2] First article of Session 10, in the ICPE 2019 conference https://icpe2019.spec.org/conference-program.html
[3] Shen and Iosup (2014) Modeling Avatar Mobility of Networked Virtual Environments. MMVE@MMSys 2014: 2:1-2:6
[4] Shen et al. (2014) Characterization of Human Mobility in Networked Virtual Environments. NOSSDAV.
[5] Bal et al. (2016) A Medium-Scale Distributed System for Computer Science Research: Infrastructure for the Long Term. IEEE Computer 49(5): 54-63.
[6] Jerrit Eickhoff, Jesse Donkervliet, Alexandru Iosup (2022) Meterstick: Benchmarking Performance Variability in Cloud and Self-hosted Minecraft-like Games. ISPASS 2022: 147-149

AI-12c. Understanding the workloads of Minecraft-like services through longitudinal analysis. Minecraft is a block-based online virtual environment with over 50M active users, that was developed as an indie project and later was bought for >$2.5 billion by Microsoft. Today, the Minecraft ecosystem includes tens of thousands of community-developed mods, a thriving emulation and server-replica community, and -- the highest level of flattery -- hundreds of Minecraft-like commercial games and education environments. Our Opencraft project aims to make Minecraft-like environments elastic, using cloud computing concepts and designs, such as elastic hosting for online games [1], autoscaling [2], modern definitions of elasticity and other performance metrics [3], etc., and we adopt rigorous scientific and experimental computer science methods [4,5]. Key to making Minecraft-like services efficient and sustainable when attracting rapidly increasing amounts of users is to understand the historical workload characteristics, and to use this knowledge to tune existing and design new autoscalers. Your thesis will provide this missing knowledge. You will analyze our longitudinal dataset of server occupancy in Minecraft and other popular online games; longitudinal data is the best a researcher can hope for, and we have spent many years collecting such data. You will develop instruments for statistical characterization and modeling, and observe and make new findings that could change the field. If the results are of exceptional quality, you will also become a co-founder of the new Game Uptime Archive.

[1] Nae et al. (2011) Dynamic Resource Provisioning in Massively Multiplayer Online Games. IEEE Trans. Parallel Distrib. Syst. 22(3): 380-395.
[2] Ilyushkin et al. (2018) An Experimental Performance Evaluation of Autoscalers for Complex Workflows. TOMPECS 3(2): 8:1-32. Special Issue with Best Paper Nominees ICPE 2017.

[3] Herbst et al. (2018) Quantifying Cloud Performance and Dependability: Taxonomy, Metric Design, and Emerging Challenges. TOMPECS 3(4): 19:1-36.
[4] Feitelson, “Experimental computer science: The need for a cultural change,” The Hebrew University of Jerusalem, Tech. Rep., 2005.
[5] van der Sar et al. (2019) Yardstick: A Benchmark for Minecraft-like Services. ICPE 2019: 243-253. [Online] Also available on our team’s website: https://atlarge-research.com/pdfs/jvdsar-yardstick-benchmark-icpe-2019.pdf

AI-13. The Cloud Uptime Archive: Understanding the availability patterns of popular cloud services.

Cloud computing is present in our daily lives and underpins our knowledge economy. Cloud services have become so ubiquitous and so interconnected, that the failure of a service we know almost nothing about in the infrastructure of Google and Microsoft can slow down our smartphone and can even crash our daily workflow. Current approaches to testing do not catch enough of these failures, as indicated by the fact that all 5 Big Tech companies in the US have suffered significant and highly visible downtime in the past few years. Thus, correcting the faults leading to slowdowns and failures also requires other approaches, based on detailed understanding of what these failures are, when they occur, what is (seems to be) their root cause, and how quickly they are corrected. Your thesis will attempt to provide this information. Clouds publish such information, but often only partially or temporarily; thankfully, our team has collected such information for nearly a decade , similarly to what we did for general distributed systems[1].

In our previous projects, we have analyzed the longitudinal data and the correlations between failure events (e.g., some failures are periodic, when the Amazon EC2 infrastructure failed, the Netflix services deployed on this infrastructure failed as well). We analyzed cloud services (which are different in nature from the systems we have studies in the past), different kinds of correlations [2,3], and applying machine learning algorithms to predict correlated failures. Your work will extend this ongoing work, with a focus on analyzing the impact of cloud-service unavailability on operations – availability across services, impact on performance and cost, etc. You will design and conduct specific experiments. If your results are of exceptional quality, you will also become a co-founder of the new Cloud Uptime Archive.

[1] Javadi et al. (2013) The Failure Trace Archive: Enabling the comparison of failure measurements and models of distributed systems. J. Parallel Distrib. Comput. 73(8): 1208-1223.
[2] Gallet et al. (2010) A Model for Space-Correlated Failures in Large-Scale Distributed Systems. Euro-Par: 88-100.
[3] Yigitbasi et al. (2010) Analysis and modeling of time-correlated failures in large-scale distributed systems. GRID 2010: 65-72.

~~**AI-13b Cloud Failures Archive [COMPLETED]:** Cloud services such as Facebook, PayPal, and Github are important for society. Users of such services generally assume they work most of the time. But, we know that these services are often unresponsive and sometimes completely unavailable. We collected data about the problems experienced by these services over several years. The data was collected from different sources and is in widely different formats. You would identify the similarities and differences between data from different sources. You would also assess the validity of the data by comparing and correlating data from different sources. Finally, you would summarize the unique statistical properties of the dataset. Two final artifacts from the project will be an open easy to use dataset of all the data we collected and a set of open source tools for analyzing the dataset.~~

~~AI-14. Fission Workflows+: Optimizing serverless applications based on non-functional requirements (NFRs) [COMPLETED]~~

~~The increasingly popular field of serverless computing further disconnects application logic from operational logic. In theory, this allows serverless platforms [1,2] and providers to abstract away virtually all infrastructure. In practice, however, the users currently still frequently need to specify system-level requirements (e.g., concurrency level, memory/cpu requirements, pool size), which require intricate understanding or intuition of how these configurations affect the performance (and cost) of the serverless application. Extending on previous work in datacenters [3], this project will focus on bridging this gap between user-level (non-functional) requirements and system-level configuration. *Which user-level requirements are suitable? how do we accurately inform the user of the (performance-cost) trade-offs that they are making? How do we build a generalizable (control) system that attempts to satisfy these NFRs using system-level requirements in serverless platforms?* You will implement the results in Fission Workflows [4,5], which is a popular open-source serverless platform.~~

~~This project can be done in collaboration with Platform9 Systems (contact Erwin van Eyk through Alexandru Iosup)~~

~~[1] van Eyk et al. (2018) A SPEC RG Cloud Group's Vision on the Performance Challenges of FaaS Cloud Architectures. ICPE. [Online] Also available on our team's website:~~
~~https://atlarge-research.com/pdfs/spec-rg-cloud-serverless-performance-challenges-icpe-2018.pdf~~
~~[2] van Eyk et al. (2018) Serverless is More: From PaaS to Present Cloud Computing. IEEE Internet Computing 22(5): 8-17. [Online] Also available on our team's website:~~
~~https://atlarge-research.com/pdfs/serverless-history-now-future18ieeeic.pdf~~
~~[3] Shen, Siqi, et al. "An availability-on-demand mechanism for datacenters." 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. IEEE, 2015.~~
~~[4] https://github.com/fission/fission-workflows~~
~~[5] https://fission.io/workflows/~~

AI-15: Energy-efficiency and sustainability for 21st century ICT infrastructure

We are just starting to realize the climate impact of our ICT infrastructure. Studies in advanced economies including the Netherlands, reveal alarming trends, with datacenters already consuming about 3% of the electricity portfolio and growing rapidly, and the impact in greenhouse gas emissions and water consumption is also becoming known. Even the production of computer systems draws significant resources and affects the climate. Waste can be high, and techniques to manage efficiently for both dynamic management and long-term capacity planning can be very diverse.

Starting with a literature study, you will survey and deepen existing knowledge on energy-efficiency and sustainability for 21st century infrastructure, and how they are impacted by the fine-grained operation of large-scale computer systems.

Start reading: Andreadis, Iosup, et al. (2022) Capelin: Data-Driven Compute Capacity Procurement for Cloud Datacenters Using Portfolios of Scenarios. IEEE TPDS. Accepted, in print. URL: https://arxiv.org/abs/2103.02060

~~AI-16 Implement and analyze a key value store which separates durability from ordering~~

~~Key value stores like etcd/zookeeper present a consistent view of data by only storing those entries whose ordering has already been agreed upon. Recent research suggests there are performance gains to be had by agreeing on the order of writes *after* or simultaneous along with storing the data.~~
~~Analyze the performance of etcd with increasing key sizes. Implement a prototype kv store which can store data before, and simultaneously along with ordering. Analyze the performance, particularly under performance variability prevalent in the cloud.~~
~~Start reading: Ganesan et. al. (2021) Exploiting Nil-Externality for Fast Replicated Storage. ACM SOSP http://pages.cs.wisc.edu/~ra/nilext.pdf~~

~~AI-17 Trace guided failure injection into storage systems for bug detection~~

~~Distributed storage systems are composed of different services (metadata, durability, etc.) which interact with each other in complex ways. Errors in the interaction between different services, or different components of the same service can lead to unavailability, consistency violation, or even catastrophic data loss.~~
~~Detecting bugs in these systems manually is hard, and they are too large to use formal verification. A popular automated way to find bugs is to subject the system to different environment errors and timing delays, and see how the system reacts.~~
~~Storage systems have the interesting property that the data that moves through the system is provided by the user. Implement a tracer which can follow a known value through the storage system and generate a trace. Inject failures and delays into the system and try to extract new traces for the same operation; for example, a write.~~

~~Start reading: Liu et. al. (2017) DCatch: Automatically Detecting Distributed Concurrency Bugs in Cloud Systems. ASPLOS https://people.cs.uchicago.edu/~shanlu/paper/asplos17-preprint.pdf~~

AI-18: Design of serverless computing platforms

In the late-1950s, leasing time on an IBM 704 cost hundreds of dollars per minute. Today, cloud computing, that is, using IT as a service, on-demand and pay-per-use, is a widely used computing paradigm that offers large economies of scale. Born from a need to make platform as a service more accessible, fine-grained, and affordable, serverless computing promises a cloud environment with NoOps properties (users and especially developers don't need to focus on servers and other operations), yet are billed only for what they use ("pay-as-you-go" model), and can expect the environment to still give the illusion of infinite resource capacity (through elasticity, managed by the cloud). Serverless computing has garnered interest from both industry and academia, and today all major cloud providers have designed and developed their serverless platforms.

Starting with a literature study, you will explore and deepen the state-of-the-art in the design of serverless computing platforms, extending the prior work of the international SPEC RG and the joint effort of the Dagstuhl Seminar on Serverless Computing.

Start reading: Herbst, Iosup, et al. (2021) Dagstuhl Seminar on Serverless Computing. Sep 2021. URL: https://doi.org/10.4230/DagRep.11.4.34

AI-19: Big Graph processing systems for societal and business workloads

Graphs are, by nature, ‘unifying abstractions’ that can leverage interconnectedness to represent, explore, predict, and explain real- and digital-world phenomena. Although real users and consumers of graph instances and graph workloads understand these abstractions, future problems will require new abstractions and systems. What are the most important new techniques emerging in computer systems to help scale such systems effectively and efficiently? What needs to happen in the next decade for big graph processing to continue to succeed? In our EU project GraphMassiziver, together with partners from all around Europe, we are trying to build the next-generation serverless systems for Big Graph processing, applied to societal and business workloads. Starting with a literature study, you will join this effort.

Start reading: Sakr, Iosup, et al. (2021) The Future Is Big Graphs: A Community View on Graph Processing Systems. CACM Sep 2021. https://arxiv.org/abs/2012.06171

AI-20: Societal impact for computer systems research

In the Netherlands, computer systems and in particular datacenters enable over 3.3 million jobs and over 60% of the GDP, and every €1 invested in computer systems such as datacenters generates €15 in added value. Sustained investments in capable networking infrastructure have made the Netherlands home to one of the largest data hubs

in the world. However, the impact of ICT infrastructure on the Dutch society is still relatively unknown. What direct and indirect, technological, economic, climate-related, and more broadly societal impact do our advances have? How and how much does the Dutch research in computer systems contribute?

Starting with a literature study, you will deepen the views on the matter of a broad consortium of computer systems experts in the Netherlands. You will need both deep and broad knowledge about computer systems, and also knowledge or ability to learn quickly about how it is used in practice.

Start reading: Iosup, Trivedi, Wang, et al. (2021) Future Computer Systems and Networking Research in the Netherlands: A Manifesto. Sep 2021 URL:
https://drive.google.com/file/d/1cP6-W3ndBgOGWYvaJWl9MNloS6J2ZsI3/view?usp=sharing

AI-21: Grand experiments in computer systems

Grand experiments define modern science, from Galileo to Sir Isaac Newton, from Ivan Pavlov to Robert Millikan, from Lise Meitner to Jennifer Doudna. In contrast to sciences where the core grand experiments have already been done, computer systems still has a large variety of experiments still open. But which are the most important and beautiful experiments in large-scale computer systems? And what new grand experiments should we aim for? In this project, you will break new ground in answering these questions, in close collaboration with prof.dr.ir. Alexandru Iosup.

Start reading: Iosup et al. (2018) Massivizing Computer Systems: A Vision to Understand, Design, and Engineer Computer Ecosystems Through and Beyond Modern Distributed Systems. ICDCS 2018. URL:
https://arxiv.org/pdf/1802.05465.pdf
9

TODOs:

AI-15. <CloudMyGraph:elasticity> -- mention first elastic graph processing system in the world
AI-15b. <CloudMyGraph:heterogeneity> -- mention first distributed and CPU+GPU graph processing system in the world
AI-15c. <CloudMyGraph:serverless> -- mention first serverless graph processing system in the world
AI-15d. <CloudMyGraph:unified graph processing>
AI-16. <graph processing: revisiting scalability>

AI-17. <Design:Vision x 3>

AI-18. <Ethics>

MC - projects supervised by M. R. Crusoe

MC-01 Common Workflow Language: Mining user behavior for feature development

Common Workflow Language is a standard for describing command line applications and workflows made from them, typically using (Docker) software containers. CWL is popular in bioinformatics and is gaining traction in other fields such as astrophysics. For flexibility, users are allowed to use Javascript in their CWL tools descriptions to handle complicated command line interfaces, or to generate dynamic configuration files. For maximal language simplicity and ease of parsing/implementation it would be ideal if there was no Javascript allowed in CWL. Therefore the maintainers and contributors to the CWL standard try to monitor how users make use of the Javascript feature and then create new language constructs that fulfill those needs. However the current approach is very manual, infrequent, and non-comprehensive.

1. Quantitative analysis phase

Using online archives of CWL workflows, and by searching GitHub and GitLab: can we characterize, in an automated fashion, how users use the Javascript Expression feature of CWL? Techniques may include Abstract Syntax Trees, which would benefit from domain specific enhancement.

2. Design phase

Propose new features for the discovered JS motifs and evaluate the burden of a new language feature versus the utility to users.

Notes:
All work must be done openly and under the Apache 2.0 license. There is a possibility of the results of the project being used for many years to come!

Prerequisites: Comfort with the Linux/Unix command line

May qualify for summer funding via Google's Summer of Code

~~**MC-02 Common Workflow Language: distributed execution with data streaming**~~

~~Command line scientific analysis tools often support streaming data into or out of the tool. (At the command line we use the unix pipe "|" or named pipes to implement this). This speeds up the analysis by avoiding slow disk/storage IO.~~

~~While the CWL standard supports this approach, no CWL-aware workflow system makes use of this optimization except for one F/OSS vendor and one proprietary vendor.~~

~~You would implement this feature (automatic streaming data in and out of scientific computing tools) to one of the CWL workflow engines, such as Toil (which is Python based).~~

~~The first iteration would stream in and out of object stores (Amazon S3, Google Cloud Storage, etc..). More advanced implementations may feature direct streaming between the tools, but this requires refactoring the job scheduling engine.~~

~~Notes:~~

All work must be done openly and under the Apache 2.0 license. If successful, you will have contributed a major feature to a popular workflow engine!

Prerequisites: Python

May qualify for summer funding via Google's Summer of Code

MC-03 Cross-architecture Single instruction, multiple data (SIMD) analysis

SIMD intrinsics like SSE SSE2 SSE3 SSSE3 SSE4.1 AVX AVX2 AVX-512 are popular with C/C++ programmers for speeding up analysis code in many research domains, including bioinformatics. Alas they are architecture specific and implementing fallbacks and multi-versioning is tedious. For example, SSE is not available on Raspberry Pi which is popular in education and hobbyist settings.

The SIMD Everywhere header-only C/C++ library reduces this burden by using a variety of methods: 1) GCC or clang extensions 2) OpenMP 3) Cilk Plus 4) pure source implementation and 5) cross-architecture SIMD (e.g. implementing SSE2 with NEON ARM intrinsics).

However, the performance of these implementations are not quantified.

In this project you will benchmark the use of the SIMD Everywhere library and these different backends as used by real scientific computing codebases on a variety of hardware. You will gain experience in SIMD programming and you will improve the SIMD Everywhere Open Source project by adding new SIMD instructions and accelerated implementations.

Notes:
All work must be done openly and under the MIT license. If successful, your work will benefit many scientific and research applications!

Prerequisites: C or C++ experience

May qualify for summer funding via Google's Summer of Code.

MC-04 Bioinformatic pipelines on Arm64: performance analysis on AWS Graviton and Apple Silicon

The Intel dominance of the scientific computing market may be weakened with the popularity of Arm64 architecture systems like AWS Graviton and Apple Silicon. While there are initial reports of cost and energy savings for industrial applications, this has not been analyzed for bioinformatic pipelines.

In this project you compare the performance on a cost, time, and energy basis of multiple real world bioinformatic pipelines on Arm64 systems like those from the AWS cloud and Apple Silicon. You will have the opportunity to assist in porting the parts of bioinformatic codebases that prevent their running on Arm64 processors in conjunction with the Debian Med project. You will also gain experience using software container (docker) build and deployment technologies.

Notes:

~~All work must be done openly and under an approved Free/Open Source Software license. If successful, your work will benefit many scientific and research applications!~~

~~May qualify for summer funding via Google's Summer of Code.~~

MC-05 FAIR from the beginning: improving metadata flow through using CWL + BioComputeObjects by capturing vital metadata early

BioComputeObects (https://biocomputeobject.org/ and standardized as IEEE Std 2791-2020) are bioinformatics workflow containers with a FDA driven metadata header.

1) How can we ask researchers as early as possible to provide the necessary metadata needed to produce FAIR final results?

2) How can we transfer this metadata along the research lifecycle with fidelity?

We have a need to convert an existing workflow that is HPC based into a portable (software container based) CWL workflow. The goal is to back-derive the necessary information for the final BCO into metadata annotations in the CWL format of the workflow description itself.

https://github.com/dpastling/DUF1220_simulation is the workflow we'd like to convert from being HPC based (the scripts currently assume a SLURM cluster) to a CWL + software containerized version. We already have a hand-written BCO for the experiment as a whole, but the workflow itself is not portable yet.

~~MC-06 Toward robust, transportable, and provenance-aware bioinformatics workflows with CWL~~

~~See https://docs.google.com/document/d/1lJZcoTI2DM6cz1iCKxXbzfSFXA5ZTws7nCGiGvuVGUA/edit#~~

MC-07 CWL to DASK compiler
https://support.astron.nl/confluence/plugins/servlet/share/content/3523cedc-e0bb-49f7-a362-3783019d4d5c
Please contact Steven van der Vlugt <vlugt@astron.nl> and cc Michael R. Crusoe <michael.crusoe@vu.nl>

Introduction

ASTRON is the Netherlands Institute for Radio Astronomy. We investigate the signals that the Universe emits in the form of radio waves. Our mission is to make discoveries in radio astronomy happen. We therefore not only do fundamental astronomical research. We also design, build and manage some of the world's leading radio telescopes, and push the boundaries of technology to make increasingly better and more sensitive instruments. To increase the sensivity of our telescopes we also strive to develop new more accurate pipelines to calibrate the data and reduce the noise.

The Low-Frequency Array (LOFAR) is a large radio telescope with an antenna network located mainly in the Netherlands and spreading across seven other European countries. It is currently the largest low-frequency radio telescope worldwide, operating at the lowest frequencies observable from Earth.LOFAR is a multipurpose sensor network with an innovative computer and network infrastructure capable of handling extremely large data volumes. It consists of many receiver stations spread across the Netherlands and Europe that stream high volumes of data over private and public wide-area networks to a central data processing facility in the Netherlands. The massive data generated by LOFAR has to be processed offline using pipelines that are run in heterogeneous high-performance computing (HPC) environments.

Common workflow language (hereafter CWL) has been growly adopted in astronomical workflow in radio astronomy for various reasons such as: portability, clear defined typing and easily documented workflows. However, given the current data and performance requirement specifical of the data volumes of radio astronomy use cases the standard pipeline executor, such as toil, struggle to optimize at run time for a specific system. Other framework, such as DASK, provide a very good abstraction for execution of workflows but they lack a well documented structure such as CWL. The possibility to have a translation and optimization given the execution system on the fly from a CWL workflow to a DASK is a promising way forward to obtain the best of both worlds.

Technologies used in this project

- CWL (https://www.commonwl.org/)
- python
- singularity
- DASK (https://www.dask.org/)

Goals of the project

This project has three main goals:

1. Can we automatically translate CWL into DASK?
2. Is it possible to optimize the execution of a pipeline before the execution given the system characteristics?
3. Can this approach be generalized to any CWL pipeline?

What we develop

In this project we will develop a light-weight application that

- takes a CWL workflow a description of a system and generates a DASK workflow
- by providing suggestions or hints can optimize the workflow to have coalesced data access (ex. the copy of the data between nodes is minimized)

JD - projects supervised by Jesse Donkervliet

The projects and ideas stated below are non-exhaustive. I have a general interest in large-scale online gaming and Minecraft-like games. If you have a project proposal related to these areas, please contact me.

Three things to note before reading the project descriptions below:

- The projects are divided into three categories: msc theses, bsc theses, and honours projects. This is meant as an indication. If the project you like is not listed in the category that applies to you, we can have a discussion about if and how we can make this project work on your level.
- Many of these projects mention the word *Metaverse*, even though no metaverse yet exists. If you are having trouble imagining what the project will look like because of this term, please replace the word "metaverse" with "future of online gaming, entertainment, and society" instead.
- Each of the project descriptions below comes with a set of relevant articles to read. Because I have not yet added these to the project descriptions below as references, please ask me directly for the reading material if you are interesting in one of these projects.

MSc Theses

- **Designing a High-Level Networking Library for the Metaverse**
  Creating a multiplayer virtual world today is relatively straightforward because there exist a large number of libraries and frameworks that offer gaming network functionality out of the box. However, once selected, changing to a different framework is typically challenging and requires significant implementation changes, making selecting the right library a difficult problem. Unfortunately, the requirements for the virtual world may not yet be known at early stages of development, change over time, or require flexibility because it supports multiple different types of games and experiences—such as a metaverse. In this project we aim to design and implement a high-level, policy-based networking library for the metaverse that supports multiple important networking architectures (e.g., lockstep simulation, client-server) and allows applications to dynamically switch between these architectures. Through the implementation of a prototype, we evaluate the efficacy of this approach. We aim to support dynamic Area of Simulation [?], and incorporate this prototype into Opencraft 2 and make it available as a stand-alone library via the Unity Marketplace.
- **Large-Scale Serverless Modifiable Virtual Environment for the Metaverse**
  Successful virtual worlds such as Second Life, Minecraft, and Roblox allow players to modify the environment and create and share their own content. We refer to such virtual worlds as *modifiable virtual environments*, or MVEs. There has been a recent resurgence in interest in MVEs in the form of a Metaverse, which promises, among other features, scalability at an unprecedented scale, giving large parts of society access to these virtual worlds at the same time. However, how to design MVEs for such scale and (energy) efficiency is an open research challenge. One of the most promising emerging technologies for systems that need to meet such constraints is *serverless computing*. Although serverless has been successful is several application domains, we are just starting to understand how to leverage it for online gaming and metaverse applications. In this project, you design and prototype a scalable and serverless MVE. You build your prototype on top of the successful open-source research platform Opencraft, making it available to people worldwide to explore the possibilities of creating an open, large-scale metaverse. A successful prototype would allow 10,000 players in the same virtual environment at the same time.
- **Designing an Actor-Based Large-Scale Online Game**
  Although we live in a golden age of computer systems and ecosystems, successfully building large-scale distributed systems remains a complex and challenging task. One important reason for this is that distributed systems have to manage a large amount of complexity. Where are the computers running the system physically located? How are they connected to each other? Where is the data the system manages, and who controls it? And so on. An important choice during the design of such system is that of the

programming model. Choosing the right programming model can make building such systems much easier. However, choosing the wrong programming model unfortunately can make it much harder. One of the promising programming models for distributed systems is the *actor system*. When using this programming model, a system is represented as individual and (partially) independent actors, who control their own data and communicate with other actors by exchanging messages.

In this project, we explore how the actor model can be leveraged to support large-scale online gaming and metaverse applications. Specifically, we aim to design and prototype a online game which operates through actors to facilitate high scalability and efficient operation. We aim to implement this prototype on top of the successful open-source research platform Opencraft, making it available to people worldwide to explore the possibilities of creating an open, large-scale metaverse. A successful prototype would allow 10,000 players in the same virtual environment at the same time.

- **Designing Dynamic Consistency Models for Modifiable Virtual Environments and the Metaverse**

  To allow users to be in a virtual space together with other people, online games and metaverse applications must exchanging state between system components at a high rate and with low latency. Doing so requires solving a large number of technical challenges that are caused by hardware and software properties, as well as user behavior. For example, unreliable networks, data ownership, and simultaneous writes all have to be considered to design a good solution that works in practice. Although many engineering solutions for this problem exist today, these solutions typically impose idiosyncratic limitations on the application such as the number of users it can support or the amount of bandwidth that is required for it to operate. Making matters worse, these different solutions are typically incompatible with each other, forcing developers to pick one solution and live with the limitations. In this project aim to design a dynamic consistency model applicable for the metaverse, which allows expressing the behaviors of popular existing architectures. Through the implementation of a prototype, we evaluate the efficacy of this approach. We aim to incorporate this prototype into Opencraft 2 and make it available as a stand-alone library via the Unity Marketplace.

- **Designing a Serverless Spectating System for the Metaverse**

  An important aspect of the online gaming and metaverse ecosystem is the ability for people to engage with players without participating in the games themselves. This process is currently facilitated through video and streaming platforms such as YouTube and Twitch. However, to make this possible, both of these companies operate large-scale infrastructure. In this project, we investigate how we can create an open and on-demand spectating and streaming architecture by leveraging *serverless computing*. Serverless is a successful cloud computing technology that allows for simple scalability, operational efficiency, and fine-grained billing. Importantly, it also allows systems to scale to zero when they are not needed. Although serverless has been successful is several application domains, we are just starting to understand how to leverage it for online gaming and metaverse applications. In this project, you design a serverless spectating and streaming architecture and implement a prototype to evaluate your approach. We aim to integrate our prototype with the successful Opencraft project, and make it available as a stand-alone library available via Github and/or the Unity marketplace.

- **Designing a Data Trace Archive for the Metaverse**

  Distributed user-facing computer systems used in education and research have been created and operated since the late 1950s. Although this long track record, one of the important challenges in designing such systems today is a limited understanding of how these systems operate. Specifically, data concerning the operation of these systems (such as the number of users, number of inputs per user, task scheduling decisions, amount of data generated, etc.) has not been archived or even recorded. Without this data, it is difficult for system designers to understand the requirements that their systems need to meet, and therefore increases the risk and cost of designing large-scale systems. In this project, we aim to address this challenge through the design and implementation of a data trace archive for the metaverse. The goal

of this archive is to allow the recording and archiving of a broad range of data that allows researchers to build a better understanding of how these systems operate. Importantly, we design this system for long-term operation and storage, preserving important knowledge about these systems for future generations. We approach this problem through the implementation of a prototype. We intend for both this system and the data it archives to be made available according to the FAIR research processes. (references: Game Trace Archive, LOCKSS)

BSc Theses

- **Benchmark Gaming Networking Libraries**
  Creating a multiplayer game today is relatively straightforward because there exist a large number of libraries and frameworks that offer gaming network functionality out of the box. However, once selected, changing to a different framework is typically not easy and requires significant implementation changes, making selecting the right library an important choice. Unfortunately, the differences in behavior and performance between these libraries are not well understood. In this project, the goal is to design and implement a benchmark to evaluate the performance of the most popular gaming network libraries. Through this benchmark, we aim to better understand how provide students and academics with a better understanding of how these systems work and what trade-offs they make, and provide practitioners with a helpful tool for selecting a networking library that works for them.
- **Dynamic Consistency for Publish Subscribe Systems**
  Modern society depends crucially on distributed systems. Without the ability to plan your commute, go shopping, or watch lectures and other videos, our society would look very different. In turn, these distributed systems depend on *consistency models* to ensure that the data they display to their users makes sense and follows the rules of the application. Examples of a world without consistency models include seeing responses on social media without being able to see the original post, and receiving conflicting information about your commute from the screens at the station and your mobile app. Although we live in a world where we do have consistency models, it is challenging to meet the requirements of modern distributed systems with many of these models. Recently, novel consistency models have shown to effectively address application-specific requirements. Dynamic Consistency Units, or *Dyconits*, is one such model which has shown to have beneficial properties when applied to gaming applications. However, we believe this model to have beneficial properties for many other types of distributed systems. In this project, we aim to make dyconits available to a large number of distributed systems by incorporating the model into *publish subscribe* platforms. These platforms are a common component in many distributed systems to facilitate communication between system components, and are used in a wide range of application domains. We approach this problem by building on our novel consistency taxonomy and implementing a prototype in Kafka, one of the most popular publish-subscribe systems. We aim to make this system available according to the FAIR research processes.
- **Designing Behavioral Models for Minecraft-Like Online Games**
  TODO

Honours Projects

- **Honours Project: Evaluating the Energy Impact of Online Gaming**
  TODO --- Tenatively Taken
- **Building an Online Game without a Server**
  TODO

- **A Benchmark for Virtual World Persistence Formats**
  TODO

MJ - projects supervised by Matthijs Jansen

I have a general interest in cloud and edge computing, resource management, cellular networks, energy efficiency, virtual reality, serverless computing, and performance analysis.

For my research, I have developed an automated cloud-edge infrastructure deployment and benchmarking framework called Continuum: https://github.com/atlarge-research/continuum

Most projects I offer are related to this framework. These projects seek to extend the functionalities of this framework. Key technologies and definitions related to the Continuum framework include:

- Virtualization VMs, Containers
- DevOps Ansible, Terraform
- Resource management Kubernetes, Nomad
- Applications ML, **Virtual Reality, Graph Processing**
- Networks **6G**, MQTT
- Serverless computing AWS Lambda, OpenFaaS
- **Energy** Hardware and software power meters

Keywords marked as bold are priority topics.
The project proposals below show some example project ideas, but there are many more.
If you are interested in working in this area, please contact me.

About cloud and edge computing
Cloud computing represents the de facto standard for computing today, where a user can summon a large fleet of servers, and deploy a variety of user-customized infrastructure services (storage, resource management, scaling, monitoring) on them in a few clicks. In contrast to cloud computing, edge computing is an emerging computing paradigm where the majority of data is generated and processed in the field using decentralized, heterogeneous, and mobile computing devices and servers, often with limited resources.

MJ-01 [Industrial Internship]: Energy management of industry cloud workloads at Ortec Finance

What: Lit. survey (non-internship) and MSc thesis (internship)
Requirement: Open to MSc students who did well in Distributed Systems and/or Storage Systems courses.

With new EU rules calling for better reporting on company sustainability, there's a growing need to measure and understand the environmental impact of cloud computing. The desire for carbon emissions reporting of cloud operations will soon be the norm for private companies and research institutions alike. The rise of new tools for managing the deployment of such instances and measurement of electricity consumption has brought forward the possibility of cloud users to quantify the carbon emissions of their workloads. As a student working together with Ortec Finance and the AtLarge research group, you can work on highly relevant issues such as measuring the energy use, monetary cost and the carbon footprint of workloads in the cloud.
You will be able to build on the work done by Ortec Finance with tools like Sailfish and use new tools such as Kepler to measure energy use. The project will also cover the practical side of setting up and managing Kubernetes workloads in the cloud. This work aims to help companies and research institutions manage their digital infrastructure responsibly under the new EU guidelines.

As part of the thesis, the student will work as a student assistant at Ortec Finance, which will give them a chance to apply their research done in the R&D department in a real-world setting. We firmly believe that this combination provides the candidate with the best opportunity to experience what working for Ortec Finance is like. You can learn more about it here. The goal is to help the student develop practical solutions for energy and cost-saving in the cloud, while also keeping an eye on reducing environmental impact.

MJ-02: 5G/6G Network Emulation in the Compute Continuum

What: BSc, MSc + Lit. survey
Requirement: Having followed one or multiple computer network courses with sufficient grades, such as computer networks (BSc), advanced network programming (BSc).

Devices in the compute continuum are connected through a variety of networks (e.g., WiFi, ethernet, bluetooth, 4G, 5G, etc.), each with its own characteristics (e.g., latency, throughput, reliability, energy usage, etc.). Currently, the Continuum framework emulated network staticically, that is, network properties between emulated cloud, edge, and endpoint devices are set when the framework in invoked, and will not change over time. This allows for limited flexibility in network emulation, with static throughput, latency, packet drop rate, etc.

In this thesis project, you will extend Continuum's network emulation to emulate real-world 5G traces. You will use existing, real-world traces of 5G network that include real-world latency, throughput, and packet drop rate metrics from physical 5G infrastructure, and use those traces in the Continuum framework to emulate 5G networks. You will use the Mahimahi tool for this, a state-of-the-art network emulation tool that can replay network traces. After implementing Mahimahi in the Continuum framework, you will verify the functionality of the tool by measuring the emulated network's performance with tools such as Netperf, and comparing the network's performance and characteristics to the input trace, to answer the question if Mahimahi's network emulation is accurate and reliable. This project can be extended further in multiple ways, including traces from other network technologies (e.g., WiFi, Bluetooth), more in-depth network verification, more complex network emulation, etc.

Links

- Mahimahi paper: Mahimahi: Accurate Record-and-Replay for HTTP
- Mahimahi Github: mahimahi-traces
- Mahimahi Ubuntu Manpage: Ubuntu Manpage: mahimahi - lightweight, composable network-emulation tools
- Mahimahi website: Mahimahi
- Further related work on 5G networks and traces: Understanding Operational 5G: A First Measurement Study on Its Coverage, Performance and Energy Consumption

MJ-03: 6G from a Computer Systems Perspective

What: BSc, MSc + Lit. survey
Requirement: Having followed one or multiple computer network courses with sufficient grades, such as computer networks (BSc), advanced network programming (BSc).

6G seeks to bring great changes in wireless communication with much lower latency, higher throughput, and new network programmability allowing multiple users and applications sharing the same network with different priorities and configurations. While 5G has been standardized and is being rolled out across the world, 6G is still in the standardizing process.

In this project, you will analyze requirements for 6G from the networking community and telecom companies (https://www.nowpublishers.com/article/DownloadEBook/9781638282389?format=pdf), and compare them to requirements from the computer systems community and services. What do the computer systems and cloud communities need to do to properly support 6G? What new systems and system requirements do we need? How do we even define “proper support”?

MJ-04: WebAssembly in the Compute Conitnuum

What: Bsc, MSc + Lit.Survey
Requirement: Experience wiith virtualization technology (e.g., virtual machines, containers). Experience with resource management software (preferably Kubernetes) would help.
Status: This project is reserved!

Deploy and benchmark WebAssembly in the Continuum framework. WebAssembly is a new virtualization technology that promises to be much more performant compared to VMs and containers, but trades in isolation. For this thesis, you will survey WebAssembly runtimes, and integrate one or multiple into the Continuum framework. This will be an alternative to the current Kubernetes with containerized applications setup, or OpenFaaS with serverless applications.

MJ-05: Trace Generation and Replaying at the Edge

What: BSc, MSc + Lit. survey
Requirement: Experience with virtualization technology, resource managers, and cloud technology would help but are not strict requirement.

Traces allow users to capture and replay the behaviour of an application or system, without needing to run the actual setup. This allows users to perform and measure a specific deployment, and share a trace of this deployment with the community for developing resource management and scheduling algorithms. For this project, you will generate application and/or system level traces of edge application deployments using the Continuum framework, transform the generated traces into a standardized trace format, and replay the trace either inside the Continuum framework or in one of many popular simulators for edge infrastructure. This project is available for multiple BSc and MSc students, and offers flexibility.

MJ-06: Serverless Kubernetes

What: MSc + Lit. survey
Requirement: This is a more complex thesis. It requires the student to have a good grade in the MSc distributed systems course and/or familiarity with Kubernetes, serverless, and AWS services.
Status: This project is reserved!

Most workloads in the IT sector use containerized applications, with the resource manager Kubernetes being the primary choice for deploying these containers on a cluster of machines for the user [1]. Kubernetes can dynamically scale applications up and down to serve a variable amount of workload requests over time, spinning up many applications on request bursts, and scaling to zero with no user requests. However, Kubernetes' controlplane, responsible for managing the applications and cluster, is always running, which takes up a significant amount of resources, and is inflexible for scaling the Kubernetes cluster itself up and down as this currently requires control plane components to restart.
In this thesis, you will deploy Kubernetes' control plane on a serverless platform, AWS Lambda [2], to make the control plane scale to zero when there are no user requests coming in, and dynamically scale up when more requests (either from users or the Kubernetes cluster itself) are coming in.

[1] https://www.cncf.io/reports/cncf-annual-survey-2022/
[2] https://aws.amazon.com/lambda/

MJ-07: Graph Processing in the Continuum

What: BSc, MSc + Lit. survey
Requirement: Experience with virtualization technology, resource managers, and graph processing would help, but are not strict requirements.

Graphs processing powers many well-known distributed services such as social media, big data, and gaming. Graph processing requires significant resources (CPU, memory) to operate and therefore has been exclusively been used within large-scale cloud datacenters. However, with users demanding lower latency and more privacy for graph processing applications, processing has been moving more towards the users, out of the datacenter, to the edge.

In this project, you will implement a graph processing benchmark in the Continuum framework, and not only focus on measuring performance of various graph processing applications and setups across the continuum, but also on measuring the energy usage of these different setups to research the performance / energy trade-off for graph processing applications.

MJ-08: Configuration Management for Cloud Systems

What: MSc + Lit. survey
Requirement: Experience with virtualization technology and resource managers would help, but are not strict requirements.

Computing systems such as Kubernetes, Spark, and Mesos power large parts of the modern cloud ecosystem. To support the wide variety of cloud infrastructures and use cases, these systems have become extremely modular, distributed, and consequently, complex. Users manage these complex systems via configuration parameters. However, with up to 1600 configuration parameters for Kubernetes alone, knowing what parameters to update and how to update them is a complex task of its own.

In this thesis, you will explore how one of these cloud computing systems manages the application life-cycle (from start to finish) and how its configuration space affects this life-cycle. Based on this study, you will design, implement, and evaluate a framework that automatically optimizes the configuration for a given deployment and workload.

MJ-09: Characterizing Energy and Carbon use of Kubernetes

What: MSc + Lit. survey
Requirement: Experience with virtualization technology and resource managers would help, but are not strict requirements.

In this thesis, you will measure, quantify, and characterize the energy use of common workloads for the popular resource manager Kubernetes. You will measure energy using the Kepler tool, but this can be extended to multiple parameters for an in-depth comparison.

MJ-10: Workload scheduling in the compute continuum

What: MSc + Literature Study
Low priority

There are many different computing models that leverage cloud, edge, and IoT resources, and often serve many users simultaneously, sharing infrastructure between different applications from different users, which introduces a need for efficient workload scheduling. Together, these computing models form a computing continuum. In this literature study you will investigate how workload scheduling is done at the cloud, edge and IoT-level, and how these insights are combined in multi-tier schedulers which consider the entire compute continuum.

Following the literature study, in the master thesis you will implement scheduling algorithms for the compute continuum by extending an already existing benchmarking platform. This platform offers automatic resource deployment across the continuum, handles the installation and usage of resource managers such as Kubernetes and KubeEdge, and deploys and benchmarks applications on these resource managers. You will extend this platform by adding scheduling algorithms and benchmarking their performance impact.

MJ-11: Resource Management in the Continuum

What: BSc, MSc + Literature Study
Low priority

Managing large-scale distributed infrastructure between cloud and edge is a very complex task, which includes key concerns such as scalability, consistency, availability, etc. Especially for the edge, with limited resources per device, lightweight, low-overhead resource management is key to use the available compute, storage, and network resources as efficiently as possible. In this project, you will extend Continuum's support of cloud-edge resource managers with Kubernetes and KubeEdge, to other lightweight resource managers and perform an in-depth performance comparison between these resource managers. Resource managers you may consider are Nomad (https://www.nomadproject.io/), K3s (https://k3s.io/), and/or MicroK8s (https://microk8s.io/). This project can be split into either 2 BSc thesis (use one resource manager), or a MSc thesis + Literature study (use two or more, and do a survey on resource management in the edge / compute continuum).

MJ-12: Analysis of concurrent and distributed workloads

What: MSc + Literature Study
Low priority
Cloud, edge, and endpoint devices are part of a continuum, extending cloud services to the edge, providing lower latency, higher bandwidth, and more privacy to users on endpoint devices (i.e., smartphones, self-driving vehicles, smart devices, etc.).
Application deployments in the continuum are therefore inherrently concurrent and distributed: Services are located all over the continuum, across various devices, and need to cooperate to provide functionalities to the users (see https://arxiv.org/pdf/2201.07312.pdf). Moreover, cloud and edge systems often run many different applications, possibly from many different users, at the same time.
In this project, you will work with Kubernetes and KubeEdge (https://kubeedge.io/), a Kubernetes based resource manager for the edge, deploy several edge continuum workloads, and analyze the characteristics of these workloads in terms of concurrency and distribution. These characteristics can be extracted to create a model which describes the scaling behaviour and limitations of various edge applications. This MSc thesis is paired with a literature study on the inner workings of cloud / edge resource managers and workloads.

MJ-13: Serverless at the Edge

What: MSc + Lit. survey
Low priority
The serverless computing paradigm promises many desirable properties for cloud applications - low-cost, fine-grained deployment, and management-free operation. Traditionally deployed in cloud, serverless for edge is even more promising due to the resource constrained nature of the edge, and the fine-grained deployment options of serverless. The Continuum framework currently supports serverless deployment in cloud and edge using the serverless platform OpenFaaS that runs on top of Kubernetes. In this project, you will perform an in-depth performance analysis of OpenFaaS on cloud and edge, and perform various complex benchmark to analyse OpenFaaS performance under different loads. This project supports multiple MSc students, because it can be extended to cover multiple serverless providers and various different benchmarks.

MJ-14: Extended Reality in the Continuum

What: BSc, MSc + Lit. survey
Status: Reserved
Extended reality (augmented / virtual) is a new up and coming use case that seeks to move human-computer interactions from 2D (computer screens) to 3D (XR headsets). Research on how the systems behind XR operate is still in its infancy - how are XR framework designed, how can they be deployed, and what are their performance and energy characteristics.

In this project, you will use state-of-the-art VR headsets from Meta to benchmark VR applications and their underlying systems, with a focus on performance and energy measurements. You will deploy these across the Continuum, on cloud and edge, to find how viable these type of setups are for VR applications with very low latency demands but high resource requirements. For more information, read the following article written with Jesse Donkervliet, also from the AtLarge group:
https://atlarge-research.com/pdfs/2023-jansen-measuringthemetaverse.pdf

ST - projects supervised by Sacheendra Talluri

~~ST-01 - Design and Implement a Serverless version of Kubernetes [allocated]~~

~~Level: MSc~~

~~Kubernetes[1] is the most popular cloud resource management and scheduling system. It is being usede to manage large (1000s) cluster for workloads such as machine learning (For example, at OpenAI), data processing, and web services.~~
~~Kubernetes itself is a distributed system which runs on a cluster of nodes. The kubernetes cluster is usually sized to support the maximum possible workload. Provisioning for maximum usage is inefficiency and wastes resources.~~
~~Serverless computing[2] (AWS Lambda for example) is a new set of technologies supporting users to use only as much resources as required and even scale to zero resources if necessary.~~
~~In this project, you will implement a serverless version of Kubernetes which can scale to 0 resources when no cluster management activities take place. But the system should also scale up to handle thousands of cluster management events a second.~~

~~[1] https://kubernetes.io/~~
~~[2] Van Eyk, Erwin, et al. "Serverless is more: From paas to present cloud computing." IEEE Internet Computing 22.5 (2018): 8-17.~~

ST-02 - Bug Detection and Error Localization in Kubernetes [1 allocated. Maximum 2]

Level: MSc

Datacenter resource managers, such as Kubernetes, are complex software. Kubernetes offers over 256 features [1] and 1000s of configurations. It is composed of tens of components and millions of lines of code.
The complexity makes it difficult for users to debug their erroneous configurations. The complexity also makes it difficult to identify problems caused by software bugs.
In this project, you will modify the Go compiler and use dynamic dataflow analysis[2] to (i) uncover bugs in Kubernetes or (ii) provide better error messages to Kubernetes's users.

[1] Truyen, Eddy, et al. "Managing feature compatibility in Kubernetes: Vendor comparison and analysis." Ieee Access 8 (2020): 228420-228439.
[2] https://cmu-program-analysis.github.io/2023/resources/program-analysis.pdf Chapter 16

ST-03 - Dynamic Instrumentation and Tracing of Serverless Programs [1 allocated. Maximum 2]

Level: MSc, BSc

Serverless application propose to ease deployment, but instrumenting them for observability ius still a manual process. Observability is important to understand their behavior, debug problems, and improve their functionality. However, these system software often process thousands of requests a second, making full observability/tracing expensive.
The community has proposed several techniques to reduce the instrumentation overhead for full system tracing, such as tail sampling [1], and optimal log statement placement [2].
There are two version of this project:

1. In one version, you will implement a mechanism in the go compiler to automatically instrument programs to produce a fully replayable/reproducible execution trace while incurring minimum overhead.
2. In the second version, you will implement a version which will collect partial traces from multiple identical requests and assemble them into a full trace.

[1] Zhang, Lei, et al. "The Benefit of Hindsight: Tracing {Edge-Cases} in Distributed Systems." 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023.
[2] Zhao, Xu, et al. "Log20: Fully automated optimal placement of log printing statements under specified overhead threshold." Proceedings of the 26th Symposium on Operating Systems Principles. 2017.

ST-04 - Task-in-Pod Scheduling Support for Kubernetes, Focusing on Ray [2 allocated. Maximum 2]

Level: MSc, BSc

Kubernetes is a popular framework for scheduling applications in the cloud. The basic unit of scheduling in Kubernetes is the Pod. Complex applications, such as Ray, launch Pods, and further schedule their workload in the Pods as Tasks. Kubernetes is not aware of these Tasks.

Each application framework such as Spark, Dask, OpenWhisk, Knative, Ray comes with its own scheduler for these tasks. Resource management for complex applications becomes complicated and many new techniques such as energy-aware scheduling do not make it to complex applications as the schedule needs to be modified for each framework.

In this project, you will implement a Task-aware scheduler for Kubernetes as an operator [1]. Application frameworks communicate with this Task-aware scheduler instead of managing their tasks directly. This enables new scheduling techniques to be implemented once in one scheduler, and the benefits accrue to all application frameworks which use the scheduler.

[1] https://kubernetes.io/docs/concepts/extend-kubernetes/operator/

~~ST-05 - Task-in-Pod Scheduling Support for Kubernetes, Focusing on Apache Spark [2 allocated. Maximum 2]~~

~~Level: MSc, BSc~~

~~Kubernetes is a popular framework for scheduling applications in the cloud. The basic unit of scheduling in Kubernetes is the Pod. Complex applications, such as Apache Spark, launch Pods, and further schedule their workload in the Pods as Tasks. Kubernetes is not aware of these Tasks.~~

~~Each application framework such as Spark, Dask, OpenWhisk, Knative comes with its own scheduler for these tasks. Resource management for complex applications becomes complicated and many new techniques such as energy-aware scheduling do not make it to complex applications as the schedule needs to be modified for each framework.~~

~~In this project, you will implement a Task-aware scheduler for Kubernetes as an operator [1]. Application frameworks communicate with this Task-aware scheduler instead of managing their tasks directly. This enables new scheduling techniques to be implemented once in one scheduler, and the benefits accrue to all application frameworks which use the scheduler.~~

~~[1] https://kubernetes.io/docs/concepts/extend-kubernetes/operator/~~

ST-06 - Characterize the fruit fly connectome analysis workload

Level: MSc, BSc

The full brain connectome of an adult fruit fly has recently been discovered [1]. Scientists are now analyzing the connectome [2] and even simulating parts of it [3].
This project aims to analyze the computational characteristics of these workloads such as CPU utilization, memory utilization, etc. We aim to answer questions regarding the ability to parallelize these workloads and efficient in-memory representation.

[1] Dorkenwald, Sven, et al. "Neuronal wiring diagram of an adult brain." bioRxiv (2023).
[2] Lin, Albert, et al. "Network Statistics of the Whole-Brain Connectome of Drosophila." bioRxiv (2023).
[3] Wang-Chen, Sibo, et al. "NeuroMechFly 2.0, a framework for simulating embodied sensorimotor control in adult Drosophila." bioRxiv (2023): 2023-09.

ST-07 - [Industrial Internship] Workload Characterization and Simulation of the ASML iACT system.

Level: MSc [Open to students who did well in Distributed Systems or Storage Systems]

One of the focuses of iACT is performance. To this end, we have developed a multi-user test (MUT) tool that simulates multiple users using iACT in parallel and measures the performance of the platform. Currently, we are reaching flexibility, maintainability, and scalability limits of the MUT.

In this project, you will design and implement our next generation MUT. You will align with a variety of stakeholders from different teams on requirements. Using these requirements, you will then identify which framework is the

best fit for the requirements. As a proof-of-concept, you'll then implement the current MUT scenarios in this new setup which you can then demo to the stakeholders and/or department.

ST-08 - [Industrial Internship] Code Quality and Performance Assessment Tool

Level: MSc [Open to students who did well in Distributed Systems or Storage Systems]

One of the focuses of iACT is performance and flexibility. Code is a common denominator for both: it dictates how performant and flexible (and extensible!) a product is. In this project, you will perform a comprehensive literature study of available tooling that can identify improvements in our code bases and apply the found recommendations (or issues!) to showcase their usefulness and ideally improve the performance and/or code quality. Besides proven, robust tooling, we're also interested in state-of-the-art, bleeding edge tools such as https://ieeexplore.ieee.org/document/9355303 which may have never been applied to a real-world system before! Finally, after proving the value of the tools you selected, as an optional cherry on the cake, you can work with our teams to add these tools to our continuous integration pipelines and have your work integrated in our production environment!

ST-09 - [Industrial Internship] Measure and Improve Plugin Infrastructure Performance

Level: MSc [Open to students who did well in Distributed Systems or Storage Systems]

iACT's flexibility also translates in its operations: plugins are independently developed by various departments in ASML which bring additional data, functions, and algorithms to iACT.

To showcase the capabilities of the platform, the platform itself offers example reference plugins: one for transforming data and one for generating data. It is important that these plugins feature not only clean and understandable code, but also extract the most out of iACT performance-wise.

During your internship, you will investigate the code of these reference plugins on both code quality and structure, but also performance-related aspects. Are the right Spark functionalities used? Can we use more memory efficient or more performant data structures? Can we improve the SQL statements or perhaps cache intermediate results? Etc.

ST-10 - [Industrial Internship] Performance Testing CLI for Spark

Level: BSc [Open to students who did well in Computer Organization, Computer Networks or ANP], MSc [Open to students who did well in Distributed Systems or Storage Systems]

At the heart of iACT's data processing is Apache Spark. Spark is a big-data framework that offers various knobs to tune performance and tools to inspect its performance.

For this project, you'll develop a CLI tool which enables our developers to:
1. Easily view the Spark-UI by connecting to Spark running on a specified cluster.
2. Benchmark combinations of Spark configuration options and values by leveraging our performance test framework and combining the performance- related metrics in a clear overview.

In addition, you'll leverage the tool to
1. Identify useful metrics not captured by the performance test framework and add these.
2. Benchmark various Spark configurations using the scenarios offered by the performance test framework.
3. Analyze the performance results and make a recommendation which configuration values to apply.
4. Together with the relevant team(s), apply the configuration to production.

XC - projects supervised by Xiaoyu Chu

Topic: System Monitoring and Observability, Operational Data Analytics, Workloads Characterization, Reliability and Energy Analysis, AI for Time Series, Data Science, Time Series Analysis and Prediction

~~XC-01 [N/A] - A survey of machine learning for workload analysis and modeling (non-thesis)~~

~~Level: BSc~~
~~Keywords: survey, bibliometrics, citation network, knowledge graph, information retirval~~
~~This is a survey project with focus on machine learning methods used in workload analysis and modeling. Workload analysis and modeling included topics such as failures prediction, anomaly detection, workload generation, workflow scheduling etc. With more and more ML techniques in these areas, the project aims to make a systematic review and analysis of existing works, and make exploration for future trends. A best example of such work in our group is [1].~~

~~The tool to do this project is AIP(Article Information Parser), a toolbox collected and parsed publication data from DBLP, Semantic Scholar, and AMiner. AIP is developed by atLarge group and has rich information for each publication, such as venue, year, author, title, abstract. With such datasets, we can do:~~

1. ~~Selection process. Query design, filter via AIP, manually selection, and get a publication dataset.~~
2. ~~Overall analysis in workload modeling:~~
    a. ~~Basic information anlaysis. Analyze time, venues, and research areas etc.~~
    b. ~~Keyword extraction and analysis. TopN keywords, TopN keyworks over time using algorithms like TF, TFIDF etc.~~
    c. ~~Topic clustering and anlayis. Automatically topic clustering using algorithms like LDA, Kmeans etc.~~
    d. ~~Bibliometrics via CiteSpace. CiteSpace is a common software to do bibliometrics based on publication data, it has nice features such as citation and authorship network analysis, emerging trends.~~
3. ~~Analisis for each research area (failures prediction, anomaly detection, workload generation, workflow scheduling):~~
    a. ~~Information extraction. Extract dataset, approach, model input and output, use case for each article.~~
    b. ~~Analysis and discussion with focus on approach and use case.~~
    c. ~~Future direction.~~

~~[1] Versluis, L., & Iosup, A. (2021). A survey of domains in workflow scheduling in computing infrastructures: Community and keyword analysis, emerging trends, and taxonomies. Future Generation Computer Systems, 123, 156-177.~~

XC-02 [assigned] - A statistical analysis of datacenters energy

~~Level: BSc~~
~~Datacenters are imperative for the digital society but consuming a lot of power. The current power consumption of data centers in the Netherlands amounts to 2.7 billion kilowatt hours, which is 2.3% of the Dutch electricity~~

~~consumption. In this context, understanding datacenter energy usage is important to help reducing energy consumption and costs.~~

~~In this project, you will do statistical analysis and modeling of energy usage, including: (1) Collect and process node-level operational data from SURFLisa cluster. (2) Conduct temporal anlaysis on metrics we have and find what kind of factors affect the power consumption. (3) Model and predict energy usage.~~
~~[1] Mehmet Berk Cetin. Operational Characterization of a Public Scientific Datacenter During and Beyond the COVID-19 Period~~
~~[2] Shane Minnema. A Statistical Analysis of Cloud Service Failures by Severity~~

XC-03 [assigned] - Failures traces analysis and prediction

~~Level: BSc~~
~~A large-scale datacenter needs to provide high service reliability and availability because failures can not only affect the quality of service but also lead to a significant waste of time and resources. However, the large number and complexity of workload failures data make it difficult to analyze and use.~~

~~In this project, you will (1) Collect and clean job data from the scheduler of SURFLisa cluster. (2) conduct statistical analysis and comparison of failed jobs and completed jobs, find the difference between these jobs and compute how much time and resources they waste. (3) develop classification models to predict failures in advance.~~

~~[1] Shane Minnema. A Statistical Analysis of Cloud Service Failures by Severity~~
~~[2] Kondo, Derrick, Bahman Javadi, Alexandru Iosup, and Dick Epema. "The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems.~~

XC-04 [assigned] - OpenAI Operational Analysis

Level: MSc
Keywords: web data crawler; outage analysis; incidents analysis; NLP;
OpenAI has their self-reported operational status webpage: https://status.openai.com/. It includes information about past incidents, uptime status, and so on. To access and with use of these data, we can do:

(1) Write web crawler to collect and extract past incidents(text), and uptime status(time-series) data.
(2) Statistical and temporal analysis on openai uptime status data.
(3) Comparison of different openai service status: API, chatGPT, labs, playground, etc.
(4) Text analysis on incidents data: keywords, severity, cluster etc.
(5) (Plus) The impact of incident on uptime status.
(6) (Plus) The impact of public opinions and news (e.g. reddit, hacknews) on uptime status and incidents. (Also can be found in Downdetector)
(7) (Plus) Comparison the operational data from user-reported platform (e.g. Downdetector)

Similiar thesis: [pdf]

ZR - projects supervised by Zebin Ren

ZR-01 Emulate NVME Device Using FPGA

Level: MSc

In storage search, it usually needs multiple fast NVME devices(such as Intel Optane) to drain the performance of the software stack. However, this is not always practical due to the limitation of the hardware device. Since the data stored in devices is not needed in storage research(such as user-space I/O library), we can use a mock device that will return 'OK' when a command is issued to the device and do nothing else. This will give the ultra-fast NVME device. Previous studies have used no-op drivers to emulate such a device. However, it's not possible to test the performance of the whole storage stack, which includes the driver, with this method. To solve this problem, we can use FPGA to make the emulation. Which will work like real hardware, and no change is needed on the software level.

Your goal is to provide an emulation of a no-op NVME device on FPGA, which should return 'OK' with read/write requests as soon as they are issued to the device.

ZR-02 Storage virtualization with fair-sharing and QoS on latency

Level: MsC?

Most current storage virtualization solutions focus on how to deliver the raw performance of NVMe SSDs to the VMs with low CPU overhead. To achieve high-throughput with low latency and CPU overhead, many solutions directly map the hardware submission-compeletion queue pair to the VM so the I/O requests issued by the applications in VMs can be directly write to the hardware queue in the storage device. This 'direct-mapping' method reudces CPU overheads and avoids copies between host and guests.
However, this 'direct-mapping' bypass the Linux I/O stack thus impede the host from scheduling the I/O requests from guests to achieve QoS or fair-sharing.

Your goal is to design a new storage virtualization framework that can achieve:

- Low CPU and latency overhead and can have similar performance as pervious study (such as MDev-NVMe).
- Achieve low latency for latency sensitive applications.
- Fair-sharing between guests

KD - projects supervised by Krijn Doekemeijer

KD-01 - Evaluate performance characteristics of ZNS NVMe when used with NVMe-oF

Level: BSc (idea is NOT finalized, also who will supervise)

The amount of data each year keeps increasing and this requires software and hardware solutions to scale along. We require storage to be fast and to be reliably fast. This has led to a large increase in storage research and the onset of faster storage devices such as NVMe *Solid State Drives* (SSDs). NVMe can also be used in a networked setting with the *NVMe over Fabrics* (NVMe-oF) interface to reap the benefits of NVMe in a network as well. This has lead to impressive performance gains. Recently, *Zoned Namespaces* (ZNS) has been proposed as an extension to NVMe, which is said to be able to lead to more performance stability and longer device lifetime among others. Unfortunately, ZNS has not been properly tested/evaluated with NVMe-oF yet. Therefore, we need to evaluate it. This will allow us to improve our storage stack and will also aid various later research avenues as they can leverage this stack without further ado.

Your goal is to evaluate ZNS when used with NVMe. You will use both TCP and RDMA as a network protocol. You will need to measure the throughput/latency of various operations (write, read, ZNS appends, ZNS resets) across the network. It is also expected that there will be a latency breakdown (e.g. what is the overhead of TCP?) of all operations.

DN - projects supervised by Dante Niewenhuis

My main topic of research is Digital Twinning and data center simulation. I am particularly looking into methods of simulating energy and carbon emission.

I am lead developer of OpenDC (https://opendc.org/), a datacenter simulator that can be used as a tool to evaluate the performance of different topologies, schedulers or energy models. The projects I offer are all related to the development of this program. I will propose several project I think will be interesting, but am open to other ideas. If you are interested in working on OpenDC and have another idea, please contact me.

DN-01 - Simulating cooling cost in OpenDC

Level: MSc, Bsc
[Taken] OpenDC is a data center simulator that can be used as a tool to evaluate the performance of different topologies, schedulers or energy models. OpenDC does currently not take the cost of cooling machines into account. However, when designing real datacenters, both are important factors. In this project, we will create a model that simulated the energy cost of cooling.

DN-02 - Simulating Hardware placement in OpenDC

DN-03 - Graph Processing in OpenDC

Level: MSc, BSc
[Taken] OpenDC is a data center simulator that can be used as a tool to evaluate the performance of different topologies, schedulers or energy models. At the moment, OpenDC is primarily used to simulated simple VM servers running. However, modern computing consists of many more types of tasks. One such type is working with graphs. Graph processing is a very important part of modern computing and is thus something that OpenDC should support. In this project we will first measure the performance of different graph operations. After, we will use the results to create a model that could simulate graph processing tasks.

DN-04 - GPUs in OpenDC

OpenDC is a data center simulator that can be used as a tool to evaluate the performance of different topologies, schedulers or energy models. At the moment, OpenDC only supports simulating CPUs. However, GPUs play a big role in modern computing. In this project we will first measure the performance of GPUs. After, we will use the results to create a model that could simulate GPUs.

DN-05 - Exploring scheduling algorithms for Data Centers

A large number of tasks are submitted to data centers every day. A scheduler decides which tasks are run on which machines and at what time. Many different schedulers exist that all have their advantages and disadvantages. In this project, we are exploring the large number of different schedulers used by data centers. The project will start

with a literature study determining what types of algorithms are used and what their characteristics are. Next, we implement a number of these schedulers into OpenDC (a data center simulator). Using OpenDC, we compare the schedulers on different workloads.

DB - projects supervised by Daniele BONETTA

Topics for BSc and MSc thesis topics (2023/2024):

My research interests are in **Language runtimes** (e.g., the Java Virtual Machine, JavaScript engines, WebAssembly runtimes etc), **Data analytics** systems (e.g., Apache Spark, query compilation, data serialization and de-serialization), **Parallel, Distributed** and **Concurrent** programming (e.g., GPU programming, SIMD parallelization, shared-memory concurrency), and broadly speaking **Cloud** applications (e.g., emerging computing paradigms, FaaS

frameworks, etc). In general, I like to *build* things, and I am interested in all performance-related aspects of the topics above.

Note: All offered projects are challenging, require good systems programming and knowledge. I assume that at the Bachelor's level you have taken (or you are familiar with some of the topics of) **Computer Organization**, **Advanced Network Programming**, **Compiler Construction**, and **Concurrency and Multithreading**. At the Master's level, you have taken/or you are familiar with at least some of **Programming Large-Scale Parallel Systems**, **Systems Seminar**, **Programming Multi-core and Many-core Systems**. Good knowledge of **Java**, **C**, **C++**, **Rust**, or comparable languages is strongly recommended. WebAssembly (or x86 assembly!) is a plus.

For each of the projects below, I will provide detailed literature and related work to interested students.

Projects can be either [free] or [taken]. Projects marked as [taken, but…] are projects that are currently taken (or that have been completed) for which the problem space is so big that multiple projects can be offered.

MSc projects.
Typically with a literature study. Most of the projects can be adapted to BSc-level (for ambitious BSc students!)

- **[db-msc-01, taken] Compressed strings in a language VM.** Modern language VMs such as Google V8 or the Java Virtual Machine (JVM) represent in-memory strings using advanced runtime techniques such as Ropes [ROP]. In this project we want to investigate if/how compression algorithms can be used in the context of language VMs to reduce memory consumption for large in-memory strings without negative performance impacts.
- **[db-msc-02, free – This project will be co-supervised with Dr Juan Fumero, Univ.Manchester (UK)] GPU-based garbage collection acceleration.** Google V8 or the Java Virtual Machine implement very advanced Garbage Collection techniques [GC]. Most of such GCs have a parallel, multithreaded implementation. In this project, we want to study how GPUs can be used to speed-up Garbage Collection instead of using multiple threads.
- **[db-msc-03, taken, but…] ML-based compiler optimizations for the Graal compiler.** Advanced dynamic JIT compilers such as Graal [GR] rely on hundreds of heuristics-based algorithms to perform compiler optimizations. Such heuristics are often hand-tuned and sub-optimal. In this project we want to explore using Machine Learning to replace (some of) the heuristics used in a modern compiler with a ML model, with the goal of improving performance.
- **[db-msc-04, taken, but…] Binary layout performance impact.** The way an executable file is stored on disk has a potential impact on the application's startup performance. For example, large binary files might result in hundreds of page faults depending on the order in which functions are declared and executed. In this project we want to develop automatic compiler-driven techniques to optimize the layout of binary files with the goal of improving application startup.
- **[db-msc-05, free] Human readable dump of GraalVM native image binary files.** The GraalVM native image [NI] framework can be used to (ahead-of-time) compile Java applications to highly-optimized binary files, with great startup performance. In this project we want to develop a tool to disassemble GraalVM native image binary files, making it possible to reverse-engineer their content producing (where possible) a human-readable output.
- **[db-msc-06, free] Compilation service for WebAssembly.** Modern WebAssembly runtimes such as wasmer [WSR] provide in-process Just-in-time compilation. JIT compilers are normally running as a concurrent thread in the same process of the application. In this project we want to study an alternative runtime design, where the JIT compiler runs as an external "service", i.e., in a different process, potentially running on a different machine. Such client-server design would allow to reduce even more the memory

consumption of the WebAssembly runtime, as well as enable distributed code caching and other optimizations.

- **[db-msc-07, taken] Profile-guided inlining for WebAssembly** Profile-guided optimization [PGO] is a compiler optimization technique where profiling information and runtime traces from previous execution of a given application are used to optimize future executions of the same application. The intuition is that the same application will most likely behave in the same way across multiple runs, with the consequence that knowledge from past runs can be used to optimize future executions. In this project, we want to explore the usage of PGO-style optimizations for WebAssembly, starting from function inlining.
- **[db-msc-08, free] Profile-guided Garbage collection.** The Garbage Collection runtime of modern language VMs such as V8 and the Java VM is triggered by runtime heuristics based on the current and predicted memory utilization. Such heuristics are often not optimal. In this project, we want to explore the usage of PGO [PGO] style optimizations to improve the performance of Garbage Collection. The intuition is that runtime traces and profiles from previous garbage collections can be used to fine-tune and improve existing language VMs.
- **[db-msc-09, free] Accelerated JSON parsing (with Dr. Tiziano De Matteis).** JSON parsing is typically a very CPU-heavy operation, often resulting in major performance bottlenecks in web services and Data analytics applications. In this project we want to explore using accelerators (such as FPGAs or others) to speed up JSON parsing performance.
- **[db-msc-10, free] Many-tiers compilation.** Language VMs such as Google V8 or the JVM have multiple JIT compilers (typically from 2 to 5/6). Each of such compilers can internally be configured to perform a very different number of optimizations. Typically, each compiler is configured once and then used in the same configuration for all methods it has to compile. In principle, however, a JIT compiler could be “tailored” to use a different configuration for each method to be compiled. In this project, we want to extend an existing JIT compiler in order to allow fine-grained per-method configuration.
- **[db-msc-11, free] Snapshot-repeat compilation caching (Graal graph caching)** Advanced compiler optimizations are essentially Graph traversal and Graph rewriting algorithms, converting a Graph data structure (IR) from after applying each optimization. Given the often complex nature of such graphs, a typical compiler would always perform all optimizations on all graphs, without caching. In this project we want to explore the usage of alternative data structures that would allow caching of a compiler IRs. In this way, the overall runtime cost of JIT compilation would be significantly reduced, as the JIT compiler could re-use graphs from previous compilations.
- **[db-msc-12, free] Combining CRIU and GraalVM native-image.** GraalVM native image [NI] is an open-source language technology that can be used to create optimized, cloud-ready binary executables for Java. By leveraging Ahead-of-time compilation of Java code, GraalVM native images can significantly reduce applications’ startup time, leading to reduced cold starts in Cloud deployments such as AWS Lambda. CRIU [CRIU] is an emerging technology provided by the Linux kernel aimed at the same goal: minimize applications startup time. Unlike GraalVM native image, CRIU leverages “user-space” snapshotting. In this project we want to explore how the two technologies can be combined in order to minimize even further the startup latency of Cloud applications.
- **[db-msc-13, free] Compressed objects in a language VM.** The language VM for programming languages such as Java or JavaScript stores objects in the VM heap memory space. Some objects are used very often, while other objects are accessed only at a specific time (e.g., during startup). In this project we want to explore how compression algorithms could be used in the context of the Java VM in order to reduce memory consumption. The key intuition for the project is that certain objects could be stored in compressed forms, with the goal of minimizing memory consumption.
- **[db-msc-14, free] Adaptive SQL compilation in Apache Spark** Data processing systems such as Apache Spark [SPK] perform runtime JIT compilation of SQL to machine code. Such compilation is typically static:

once compiled, the code is never modified. In this project we want to instead explore "dynamic" compilation techniques in the context of Apache Spark SQL.

- **[db-msc-15, free] Learned optimizations in a language VM** Learned data structures [LEA] have been used successfully in the context of DataBase processing systems. In this project we want to explore using such data structures to implement some of the internal components of a modern language VM such as Google V8 or the Java Virtual Machine.
- **[db-msc-16, free] Predictive JSON parsing: ML-based JSON data access.** JSON parsing libraries such as jackson [JK] do not make any assumptions on the actual structure of the JSON data being parsed. However, very often multiple JSON objects have a somehow "similar" structure. In this project we want to explore using Machine Learning models to speed up JSON data analytics. Specifically, we want to develop a JSON parsing library that "learns" the most-likely structure of the JSON data being parsed, and generates a JSON parser that is optimized for the predicted JSON data. The expectation is that such "learned" JSON parser should be significantly faster than a general-purpose one.
- **[db-msc-17, free] GPU-based parsing of a general-purpose programming language such as Java, Python, or JavaScript.** Parsing languages such as Java, Python or JavaScript is typically implemented as a single-threaded sequential operation. Depending on the language, it is actually possible to implement "parallel", much faster parsing. In this project, we want to explore using GPUs to speed up the parsing step of one of such very popular programming languages.
- **[db-msc-18, free – This project will be co-supervised with Dr Juan Fumero, Univ.Manchester (UK)] Accelerating Parallel Byte-code interpreters on GPUs.** Managed programming languages like Java offer robust environments with automatic memory management, cross-platform capabilities, and a large ecosystem of libraries and tools. However, the performance of bytecode execution in these languages can be a bottleneck, especially for compute-intensive applications. What optimising runtimes and Language Virtual Machines (VMs) do is that the runtime itself collects profiling information about the program. This profiling information is then used by a Just In Time compiler to produce optimized code for the hot-spots of the input program at a later stage of the execution. However, while collecting the profiling, the application still runs on a single core byte-code interpreter. This project will explore how to accelerate the byte-code execution by making use of Graphics Processing Units (GPUs). Some research in this area: https://jjfumero.github.io/files/JuanFumero-MoreVMs2020-Preprint.pdf, https://github.com/beehive-lab/ProtonVM Design, Implementation, and Application of GPU-Based Java Bytecode Interpreters https://users.ece.utexas.edu/~gligoric/papers/CelikETAL19GVM.pdf

BSc projects.

Depending on the topic, most of them can be extended to MSc-level projects.

- **[db-bsc-01 taken] Performance and usability analysis of SIMD support in WebAssembly.** SIMD instructions are at the core of modern data processing. For example, several data ingestion operations (e.g., importing data from a CSV file) requires UTF-8 validation [UTF]: the input data needs to be analyzed (one character at a time) to ensure that the input text is valid. Parallel execution techniques can be employed to speed-up the validation of large text files. In this project, we want to explore the performance of existing WebAssembly runtimes in the context of SIMD execution, looking at common operations such as e.g. UTF-8 validation. The goal of the project is to implement multiple SIMD-based data processing techniques targeting WebAssembly (either in WebAssembly itself, or by leveraging emscripten or alternative WASM compilers), to assess the current performance of WebAssembly in such Data-parallel operations.
- **[db-bsc-02, free] Performance and usability of the new Java Vector API for JSON data processing applications.** Upcoming versions of the Java language will feature built-in support for SIMD programming

[VEC]. Simdjson [SJ] is one of the most popular high-performance JSON processing libraries; it is implemented in highly-optimized C code, and leverages advanced SIMD instructions to perform data de-serialization in parallel. Achieving similar performance using a managed programming language such as Java would be simply impossible without access to SIMD parallelism. The recently-introduced Java Vector API [VEC] promises to bring high-performance SIMD parallel programming to the Java language. In this project we want to explore the usage of such a new API in the context of JSON data processing. In particular, the goal of the project is to implement the simdjson library using the new Java Vector API, and assess its performance profile compared against the native, highly-optimized simdjson implementation.

- **[db-bsc-03, free] Power consumption profile and analysis of modern server-side Java workloads** Java and the Java Virtual Machine are among the most popular technologies on the planet, with billions of daily users. Despite such great popularity, we still know very little about the carbon footprint of a language such as Java compared to statically-compiled languages such as C. In this project, we want to explore and understand the impact of language runtimes such as the JVM on energy consumption, ideally identifying guidelines and recommendations for the most energy-friendly configuration of a Java Virtual Machine [POW]
- **[db-bsc-04, free] Performance analysis of Database connector serverless interfaces** Database connectors [DB] are software components responsible for receiving query result data from a remote Database system and exposing them to a given programming language. As such, they have to serialize and de-serialize raw binary data in order to make it available to the language VM. In this project, we want to explore the performance impact of such serialization/de-serialization step, and we want to explore how language VMs can optimize or reduce associated performance overheads.
- **[db-bsc-05, free] Stream-loading of large WebAssembly binary files** WebAssembly modules can often be relatively large (in the order of 100s of MBs). Loading such big files in a WebAssembly runtime can take up to a few seconds, which is often not acceptable. In this project we want to explore the usage of streaming techniques in WebAssembly to minimize WASM application startup.
- **[db-bsc-06, free] Understanding programs using Graphs** Language Virtual Machines such as the Java Virtual Machine (JVM) or the V8 JavaScript execution engine achieve high performance by means of very complex runtime optimizations. One key core component of modern language VMs is a dynamic Just-in-time optimizing compiler (JIT). Most modern JIT compilers rely on large graph-like intermediate representations (IR) of the application being executed. In this context, optimizations performed by the JIT Compiler correspond to traversals and rewriting operations of the graph IR. Several tools exist that can be used to visualize and analyze the IR graphs used by modern JIT compilers [IGV]. While such tools allow developers and performance engineers to inspect a single IR graph, they lack many advanced features of modern DataBase Systems (e.g., a query language). In this project we want to investigate using Graph DatataBase technology to inspect and analyze the graph IRs used by a modern JIT compiler. The goal of the project is to use a Graph DB to store IR graphs, and to define and evaluate complex graph queries on the IRs. Such queries could then be used to gain knowledge about the application being compiled by the JIT compiler, as well as to identify missed optimization opportunities. The focus of this project will be the Graal compiler [GR].
- **[db-bsc-07, free] Performance evaluation of popular binary and textual encoding formats.** Data is often encoded using binary formats such as Parquet, Cap'n Proto, Protocol Buffers, Arrow, FlatBuffers etc. In this project, we want to build a comprehensive performance evaluation suite (i.e., a benchmark) to compare the performance of popular encoding formats on popular programming languages such as Java, JavaScript, Python, C/C++ and Rust.
- **[db-bsc-08, free] Energy-aware JIT compilation** Just-in-time (JIT) compilation is often one of the most energy-intensive operations performed by modern language runtimes such as the Java VM and Google's V8 JavaScript engine. Despite the importance of JIT compilation in modern programming languages, very little is known about the energy consumption of JIT compilers. In this project we want to analyze the

performance and energy consumption of modern JIT compilers [POW]. The goal of this project is to enable fine-grained power consumption measurements in a language VM, allowing one to understand the energy costs of individual compiler operations and optimizations (e.g., “how expensive is inlining?”, “does energy consumption grow with the number of methods being compiled?”, etc).

- **[db-bsc-09, free] Data access pattern performance impact on Apache Arrow.** In this project we want to answer and understand a simple question: does the order in which we access large data files stored in the Apache Arrow [ARW] format have an impact on performance? And, if so, can the application and/or the language VM optimize the way data is accessed to improve performance?
- **[db-bsc-10, free] Performance evaluation of the new CPython JIT compiler**. The CPython engine is the most popular language VM for Python. In recent weeks, the engine was finally extended with a JIT compiler [PYJIT]. In this project, we want to assess the performance impact of the new JIT, and want to understand what its properties and limitations are.

References

[ROP] Ropes: https://www.cs.tufts.edu/comp/150FP/archive/hans-boehm/ropes.pdf
[GC] https://developers.redhat.com/articles/2021/11/02/how-choose-best-java-garbage-collector
[GR] The Graal compiler: https://github.com/oracle/graal/
[NI] Native image: https://www.graalvm.org/22.0/reference-manual/native-image/
[WSR] Wasmer: https://github.com/wasmerio/wasmer
[PGO] PGO example in Go: https://go.dev/doc/pgo
[SPK] SQL compilation in Apache Spark:
[LEA] Learned data structures: https://arxiv.org/abs/1712.01208
[JK] Jackson: https://github.com/FasterXML/jackson
[UTF] UTF-8 validation using SIMD https://arxiv.org/abs/2010.03090
[VEC] Java vector API https://openjdk.org/jeps/460
[SJ] Simdjson https://github.com/simdjson/simdjson https://arxiv.org/abs/1902.08318
[POW] Examples of Java power consumption analyses: https://inria.hal.science/hal-03275286/document and https://ionutbalosin.com/2024/03/analyzing-jvm-energy-consumption-for-jdk-21-an-empirical-study/?utm_source=pocket_saves
[DB] Example of an optimized Database connector https://dl.acm.org/doi/10.14778/3551793.3551847
[IGV] IGV Graph visualizer https://www.graalvm.org/latest/tools/igv/
[W-SIMD] WebAssembly SIMD https://github.com/WebAssembly/simd/blob/main/proposals/simd/SIMD.md
[CRIU] https://criu.org/Main_Page
[SPK] Spark SQL code generation overview:
https://www.databricks.com/blog/2016/05/23/apache-spark-as-a-compiler-joining-a-billion-rows-per-second-on-a-laptop.html
[ARW] Apache Arrow: https://arrow.apache.org/
[PYJIT] https://github.com/python/cpython/pull/113465