DAGSTUHL
REPORTS

Volume 8, Issue 3, March 2018

ISSN 2192-5283

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at
http://www.dagstuhl.de/dagpub/2192-5283

Publication date
August, 2018

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at `http://dnb.d-nb.de`.

Digital Object Identifier: 10.4230/DagRep.8.3.i

Aims and Scope
The periodical *Dagstuhl Reports* documents the program and the results of Dagstuhl Seminars and Dagstuhl Perspectives Workshops.
In principal, for each Dagstuhl Seminar or Dagstuhl Perspectives Workshop a report is published that contains the following:
- an executive summary of the seminar program and the fundamental results,
- an overview of the talks given during the seminar (summarized as talk abstracts), and
- summaries from working groups (if applicable).

This basic framework can be extended by suitable contributions that are related to the program of the seminar, e. g. summaries from panel discussions or open problem sessions.

Report from Dagstuhl Seminar 18101

Scheduling

Edited by

Magnús M. Halldórsson[1], Nicole Megow[2], and Clifford Stein[3]

1 **Reykjavik University, IS,** mmh@ru.is
2 **Universität Bremen, DE,** nicole.megow@uni-bremen.de
3 **Columbia University, US,** cliff@ieor.columbia.edu

Abstract

This report documents the program and outcomes of the Dagstuhl Seminar 18101 "Scheduling" in March 2018. The seminar brought together algorithmically oriented researchers from two communities with interests in resource management: (i) the scheduling community and (ii) the networking and distributed computing community. The primary objective of the seminar was to expose each community to the important problems and techniques from the other community, and to facilitate dialog and collaboration between researchers.

Seminar March 4–9, 2018 – https://www.dagstuhl.de/18101
2012 ACM Subject Classification Mathematics of computing → Combinatorial optimization, Networks → Network algorithms, Theory of computation → Approximation algorithms analysis, Theory of computation → Scheduling algorithms
Keywords and phrases scheduling, optimization, approximation algorithms
Digital Object Identifier 10.4230/DagRep.8.3.1
Edited in cooperation with Lukas Nölke

1 Executive Summary

Magnús M. Halldórsson
Nicole Megow
Clifford Stein

This fifth meeting in a series of Dagstuhl "Scheduling" seminars brought together part of the community of algorithmic researchers who focus on scheduling, and part of the community of algorithmic researchers who focus on networking in general, and resource management within networks in particular. These communities are far from unknown to each other as they attend the same general academic conferences. But as each community has its own specialized conferences, there is less interaction between these communities than there should be. Further there are differences in the types of algorithmic problems these communities are naturally drawn towards.

The primary objective of the seminar was to expose each community to the important models, problems and techniques from the other community, and to facilitate dialog and collaboration between researchers. The program included 22 invited main talks including an inspiring talk on practical applications at ABB Corporate Research, 8 short spot-light talks, two open problem sessions in the beginning of the week, and ample unstructured time for research and interaction. The overall atmosphere among the 44 participants was very interactive.

A highlight of the seminar was a joint Wednesday-session with the Dagstuhl Seminar 18102 "Dynamic Traffic Models in Transportation Science". It was a fortunate coincidence that both seminars were scheduled in parallel. Indeed, questions related to networks, scheduling and resource sharing arise naturally in traffic control and transportation science. It was an inspiring secondary outcome of the workshop to realize this strong overlap in interests which led to interesting discussions between researchers of the different communities.

2 Table of Contents

3 Overview of Talks

3.1 Improved Online Algorithm for Weighted Flow Time

Yossi Azar (Tel Aviv University, IL)

License Creative Commons BY 3.0 Unported license
© Yossi Azar
Joint work of Yossi Azar, Noam Touitou
Main reference Yossi Azar, Noam Touitou: "Improved Online Algorithm for Weighted Flow Time", CoRR, Vol. abs/1712.10273, 2017.
URL http://arxiv.org/abs/1712.10273

We discuss one of the most fundamental scheduling problem of processing jobs on a single machine to minimize the weighted flow time (weighted response time). Our main result is a $O(\log P)$-competitive algorithm, where P is the maximum-to-minimum processing time ratio, improving upon the $O(\log^2 P)$-competitive algorithm of Chekuri, Khanna and Zhu (STOC 2001). We also design a $O(\log D)$-competitive algorithm, where D is the maximum-to-minimum density ratio of jobs. Finally, we show how to combine these results with the result of Bansal and Dhamdhere (SODA 2003) to achieve a $O(\log(\min(P, D, W)))$-competitive algorithm (where W is the maximum-to-minimum weight ratio), without knowing P, D, W in advance. As shown by Bansal and Chan (SODA 2009), no constant-competitive algorithm is achievable for this problem.

3.2 Scheduling Under Uncertainty In Safety-critical Systems

Sanjoy K. Baruah (Washington University, US)

License Creative Commons BY 3.0 Unported license
© Sanjoy K. Baruah

Many safety-critical system designs are subject to validation (in some cases, certification) prior to deployment. Consequently, routing and resource-allocation decisions for such systems may need to be made prior to run-time, with incomplete knowledge of the actual conditions that will be encountered by the system during run-time. I will discuss some open scheduling and routing problems that arise in the analysis of such safety-critical real-time systems as a consequence of needing to make decisions in the presence of this uncertainty.

3.3 On Minimizing the Makespan with Bag Constraints

Syamantak Das (IIIT - New Dehli, IN)

License Creative Commons BY 3.0 Unported license
© Syamantak Das
Joint work of Syamantak Das, Andreas Wiese
Main reference Syamantak Das, Andreas Wiese: "On Minimizing the Makespan When Some Jobs Cannot Be Assigned on the Same Machine", in Proc. of the 25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria, LIPIcs, Vol. 87, pp. 31:1–31:14, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
URL http://dx.doi.org/10.4230/LIPIcs.ESA.2017.31

We study the classical scheduling problem of assigning jobs to machines in order to minimize the makespan. It is well-studied and admits an EPTAS on identical machines and a $(2-1/m)$-approximation algorithm on unrelated machines. In this work we study a variation in which the input jobs are partitioned into bags and no two jobs from the same bag are allowed to be assigned on the same machine. Such a constraint can easily arise, e.g., due to system stability and redundancy considerations. Unfortunately, as we demonstrate in this work, the techniques of the above results break down in the presence of these additional constraints. Our first result is a PTAS for the case of identical machines. It enhances the methods from the known (E)PTASs by a finer classification of the input jobs and careful argumentations why a good schedule exists after enumerating over the large jobs. For unrelated machines, we prove that there can be no $(\log n)^{1/4-\epsilon}$-approximation algorithm for the problem for any $\epsilon > 0$, assuming that NP $\subseteq$ ZPTIME$\cdot(2^{(logn)^{O(1)}})$. This holds even in the restricted assignment setting. However, we identify a special case of the latter in which we can do better: If the same set of machines we give an 8-approximation algorithm. It is based on rounding the LP-relaxation of the problem in phases and adjusting the residual fractional solution after each phase to order to respect the bag constraints.

3.4 Fairness, Congestion Control, and Related Open Problems

Jelena Diakonikolas (Boston University, US)

Fairness is a central topic in a variety of areas, ranging from political philosophy, economic theory, and operations research to network congestion control, and more recently, machine learning. The purpose of this talk is three-fold: (i) to give a historical overview of different philosophical approaches to fairness, (ii) to formally introduce fair resource allocation problems and discuss different algorithmic approaches to solving them in the context of (offline) network congestion control, and (iii) to discuss related open problems in online optimization and scheduling.

3.5 Internet Transport Service using Dissemination Graphs, and the Shallow-Light Steiner Network Problem

Michael Dinitz (Johns Hopkins University - Baltimore, US)

Joint work of Amy Babay, Emily Wagner, Yair Amir, Zeyu Zhang
Main reference Amy Babay, Emily Wagner, Michael Dinitz, Yair Amir: "Timely, Reliable, and Cost-Effective Internet Transport Service Using Dissemination Graphs", in Proc. of the 37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017, pp. 1–12, IEEE Computer Society, 2017.
URL http://dx.doi.org/10.1109/ICDCS.2017.63
Main reference Amy Babay, Michael Dinitz, Zeyu Zhang: "Characterizing Demand Graphs for (Fixed-Parameter) Shallow-Light Steiner Network", CoRR, Vol. abs/1802.10566, 2018.
URL http://arxiv.org/abs/1802.10566

Emerging applications such as remote manipulation and remote robotic surgery require communication that is both timely and reliable, but the Internet natively supports only communication that is either completely reliable with no timeliness guarantees (e.g. TCP) or timely with best-effort reliability (e.g. UDP). We present an overlay transport service that can provide highly reliable communication while meeting stringent timeliness guarantees over the Internet. To do this we introduce "dissemination graphs", providing a unified framework for specifying routing schemes ranging from a single path, to multiple disjoint paths, to arbitrary graphs. We develop a timely dissemination-graph-based routing method using these graphs that can add targeted redundancy in problematic areas of the network. This routing method is based on algorithms for the Shallow-Light Steiner Network problem, in which we are given a graph $G = (V, E)$, a collection of pairs of vertices (the demands), and a length bound L, and are asked to find the smallest subgraph in which all demands have distance at most L. Motivated by dissemination graphs, we exactly characterize the classes of demands for which the problem is fixed parameter tractable, and prove that for all other classes of demands the problem is $W[1]$-hard.

3.6 Proximity Results and Faster Algorithms for Integer Programming using the Steinitz Lemma

Fritz Eisenbrand (EPFL - Lausanne, CH)

Joint work of Friedrich Eisenbrand, Robert Weismantel
Main reference Friedrich Eisenbrand, Robert Weismantel: "Proximity results and faster algorithms for Integer Programming using the Steinitz Lemma", in Proc. of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018, pp. 808–816, SIAM, 2018.
URL http://dx.doi.org/10.1137/1.9781611975031.52

We consider integer programming problems in standard form $\max\{c^T x : Ax = b,\, x \geq 0,\, x \in Z^n\}$ where $A \in Z^{m\times n}$, $b \in Z^m$ and $c \in Z^n$. We show that such an integer program can be solved in time $(m\Delta)^{O(m)} \cdot \|b\|_\infty^2$, where Δ is an upper bound on each absolute value of an entry in A. This improves upon the longstanding best bound of Papadimitriou (1981) of $(m \cdot \Delta)^{O(m^2)}$, where in addition, the absolute values of the entries of b also need to be bounded by Δ. Our result relies on a lemma of Steinitz that states that a set of vectors in R^m that is contained in the unit ball of a norm and that sum up to zero can be ordered such that all partial sums are of norm bounded by m. We also use the Steinitz lemma to show that the ℓ_1-distance of an optimal integer and fractional solution, also under the presence of upper bounds on the variables, is bounded by $m \cdot (2\,m \cdot \Delta + 1)^m$. Here Δ is again an upper bound on the absolute values of the entries of A. The novel strength of our bound is that it is independent of n. We provide evidence for the significance of our bound by applying it to general knapsack problems where we obtain structural and algorithmic results that improve upon the recent literature.

3.7 Optimization and Scheduling with Explorable Uncertainty

Thomas Erlebach (University of Leicester, GB)

Explorable uncertainty refers to settings where parts of the input data are initially unknown, but can be obtained at a certain cost using queries. In a typical setting, initially only intervals that contain the exact input values are known, and queries can be made to obtain exact values. An algorithm must make queries one by one until it has obtained sufficient information to solve the given problem. We discuss two lines of work in this area: In the area of query-competitive algorithms, one compares the number of queries made by the algorithm with the best possible number of queries for the given input. In the area of scheduling with explorable uncertainty, queries may correspond to tests that can reduce the running-time of a job by an a priori unknown amount and are executed on the machine that also schedules the jobs, thus contributing directly to the objective value of the resulting schedule.

3.8 On Scheduling Consistent Software-Defined Network Updates

Klaus-Tycho Foerster (Universität Wien, AT)

While Software Defined Networks are controlled centrally by one (logical) controller, the dissemination of updates in the network itself is an inherently asynchronous distributed process. Even though eventual consistency (e.g., no forwarding loops) is easy to guarantee, many useful network properties might be violated during the update process. In this talk, we will concentrate on the problem of scheduling such updates in a way that the consistency properties are not violated. In particular, we focus on the consistency properties of loop freedom and congestion freedom, providing a general overview and pointing out open problems.

3.9 How To Plan Ahead

Seth Gilbert (National University of Singapore, SG)

Joint work of M. Bender, M. Farach-Colton, Sándor P. Fekete, Jeremy T. Fineman, Seth Gilbert, Shunhao Oh
Main reference Michael A. Bender, Martin Farach-Colton, Sándor P. Fekete, Jeremy T. Fineman, Seth Gilbert: "Reallocation Problems in Scheduling", Algorithmica, Vol. 73(2), pp. 389–409, 2015.
URL http://dx.doi.org/10.1007/s00453-014-9930-4
Main reference Shunhao Oh, Seth Gilbert: "A Reallocation Algorithm for Online Split Packing of Circles", CoRR, Vol. abs/1802.05873, 2018.
URL http://arxiv.org/abs/1802.05873

Planning ahead has many benefits: it often leads to better results and less last minute stress. That is not what this talk will be about.

Instead, let us think about scheduling. Imagine you are scheduling appointments at a doctor's office. Dr. Spaceman has a busy schedule, with patients all day. And then a VIP case arrives which has to be scheduled exactly at 10am. The result? All the other patients have to be rescheduled, which makes them exceedingly unhappy. Ideally, there would be some way to plan ahead and avoid this disruption.

The same type of rescheduling problem occurs in many different contexts, ranging from delivery schedules to airline routes to assembly lines on a factory floors. In general, we want to maintain a nearly optimal schedule, while allowing jobs to be inserted and deleted. As the set of jobs in the system changes, we will reschedule existing jobs to maintain an efficient schedule. However, rescheduling jobs has a cost, and our goal is to minimize that cost. Specifically, I will talk about three examples: scheduling with arrival times and deadlines, scheduling to minimize the makespan, and scheduling to minimize the sum-of-completion-times.

In general, we will see that by planning ahead, we can minimize the disruption caused by changes to the schedule.

3.10 Approximation Algorithms for Stochastic Scheduling and Routing

Anupam Gupta (Carnegie Mellon University - Pittsburgh, US)

License Creative Commons BY 3.0 Unported license
© Anupam Gupta
Joint work of Anupam Gupta, Archit Karandikar, Amit Kumar, Viswanath Nagarajan, Xiangkun Shen
Main reference Anupam Gupta, Archit Karandikar: "Stochastic Unsplittable Flows", in Proc. of the Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA, LIPIcs, Vol. 81, pp. 7:1–7:19, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
URL http://dx.doi.org/10.4230/LIPIcs.APPROX-RANDOM.2017.7
Main reference Anupam Gupta, Amit Kumar, Viswanath Nagarajan, Xiangkun Shen: "Stochastic Load Balancing on Unrelated Machines", in Proc. of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018, pp. 1274–1285, SIAM, 2018.
URL http://dx.doi.org/10.1137/1.9781611975031.83

In this talk we will talk about some recent results for scheduling and routing problems where the input parameters are are not deterministic but random variables with known distributions. The goal now is to optimize some expected measure of goodness. We will show some of the ideas needed to develop algorithms in this setting, and to prove their performance guarantees.

3.11 MapReduce Models and Algorithmics

Sungjin Im (University of California - Merced, US)

License Creative Commons BY 3.0 Unported license
© Sungjin Im

MapReduce and its follow-up runners such as Spark have become dominant massively parallel computing platforms. These platforms take care of underlying cumbersome distributed system issues under the hood while offering easy interface to the programmers. They are distinguished from traditional parallel computing platforms in the effective way they bridge local computation and communication across machines. Motivated by the tremendous success of the platforms, there have been attempts in the algorithms community to model their unique computing constraints and power, and consequently, many interesting algorithmic ideas have been discovered. This talk will give a quick overview of the theoretical models and key algorithmic ideas/results developed for the massively parallel computing platforms.

3.12 Bypassing Lower Bounds by Stochastic Input Models: The Temp Secretary Problem and Beyond

Thomas Kesselheim (TU Dortmund, DE)

License Creative Commons BY 3.0 Unported license
© Thomas Kesselheim
Joint work of Thomas Kesselheim, Andreas Tönnis
Main reference Thomas Kesselheim, Andreas Tönnis: "Think Eternally: Improved Algorithms for the Temp Secretary Problem and Extensions", in Proc. of the 24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark, LIPIcs, Vol. 57, pp. 54:1–54:17, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.
URL http://dx.doi.org/10.4230/LIPIcs.ESA.2016.54

Stochastic input models are a promising way to bypass lower bounds of worst-case analysis. Generally, the input is specified by an adversary only to some degree and the remainder is drawn from a probability distribution. Typical examples are random-order analyses of online algorithms and smoothed analysis.

The Temp Secretary Problem (originally proposed by Fiat et al., ESA 2015) is a variant of online deadline scheduling in such a model. An adversary defines jobs by weights and processing times. The release dates are drawn i.i.d. uniformly from $[0, 1]$. At any release date, the algorithm may choose to accept the job and then has to process it immediately without preemption, or it may reject it. Each machine can process only one job at a time. The goal is to maximize the weight of the accepted jobs. If the input is completely adversarial, one cannot achieve any reasonable guarantee. In the partly-stochastic model, it is possible to be constant competitive (see K. and Tönnis, ESA 2016).

In the domain of scheduling, there are certainly many more examples of problems that are interesting to study in such an input model. Besides bypassing lower bounds on the competitive ratio, probably one can also bypass hardness-of-approximation bounds. An interesting candidate would be Flow Time minimization. I am interested in (further) potential applications and feedback for model refinement as well as collaborations to solve the (still to be defined) open problems.

3.13 Coflow Scheduling and Beyond

Samir Khuller (University of Maryland - College Park, US)

Applications designed for data-parallel computation frame-works such as MapReduce usually alternate between computation and communication stages. Coflow scheduling is a popular networking abstraction introduced to capture such application-level communication patterns in datacenters. In this framework, a datacenter is modeled as a single non-blocking switch with m input ports and m output ports. A coflow is a collection of flow demands that is said to be complete once all of its requisite flows have been scheduled. We consider the offline coflow scheduling problem with and without release times to minimize the total weighted completion time. Coflow scheduling generalizes the well studied concurrent open shop scheduling problem and is thus NP-hard.

We give a survey of recent results on Coflow scheduling and also some recent directions. This will be a short survey talk.

3.14 Sublinear communication for Solving Network Problems

Valerie King (University of Victoria, CA)

I describe a simple algorithm by which each node in a network sends one message of $n^{1/2}$ bits to a central controller which can then determine the connected components of the network. The nodes know only the approximate size of the network and there is no shared randomness.

3.15 Constant Factor Approximation Algorithm for Weighted Flow Time on a Single Machine in Pseudo-polynomial time

Amit Kumar (Indian Inst. of Technology - New Dehli, IN)

License Ⓒ Creative Commons BY 3.0 Unported license
© Amit Kumar
Joint work of Jatin Batra, Naveen Garg, Amit Kumar
Main reference Jatin Batra, Naveen Garg, Amit Kumar: "Constant Factor Approximation Algorithm for Weighted Flow Time on a Single Machine in Pseudo-polynomial time", CoRR, Vol. abs/1802.07439, 2018.
URL http://arxiv.org/abs/1802.07439

In the weighted flow-time problem on a single machine, we are given a set of n jobs, where each job has a processing requirement, release date and weight. The goal is to find a preemptive schedule which minimizes the sum of weighted flow-time of jobs, where the flow-time of a job is the difference between its completion time and its released date. We give the first pseudo-polynomial time constant approximation algorithm for this problem. The running time of our algorithm is polynomial in n, the number of jobs, and P, which is the ratio of the largest to the smallest processing requirement of a job. Our algorithm relies on a novel reduction of this problem to a generalization of the multi-cut problem on trees, which we call the Demand Multi-Cut problem. Even though we do not give a constant factor approximation algorithm for the Demand Multi-Cut problem on trees, we show that the specific instances of Demand Multi-Cut obtained by reduction from weighted flow-time problem instances have more structure in them, and we are able to employ techniques based on dynamic programming. Our dynamic programming algorithm relies on showing that there are near optimal solutions which have nice smoothness properties, and we exploit these properties to reduce the size of DP table.

3.16 Distributed Shortest Paths, Exactly

Danupon Nanongkai (KTH Royal Institute of Technology, SE)

License Ⓒ Creative Commons BY 3.0 Unported license
© Danupon Nanongkai
Joint work of Chien-Chung Huang, Danupon Nanongkai, Thatchaphol Saranurak
Main reference Chien-Chung Huang, Danupon Nanongkai, Thatchaphol Saranurak: "Distributed Exact Weighted All-Pairs Shortest Paths in $\tilde{O}(n^{5/4})$ Rounds", in Proc. of the 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017, pp. 168–179, IEEE Computer Society, 2017.
URL http://dx.doi.org/10.1109/FOCS.2017.24

This talk concerns the problem of quickly computing distances and shortest paths on distributed networks (the CONGEST model). There have been many developments for this problem in the last few year, resulting in tight approximation schemes. This left widely open whether exact algorithms can perform equally well. In this talk, we will discuss some recent progress in answering this question.

3.17 Guest lecture (Seminar 18102): Equilibria in the Fluid Queueing Model

Neil Olver (VU University of Amsterdam, NL)

I will discuss the fluid queueing model, introduced by Vickrey in '69. It is probably the simplest model that plausibly captures the notion of time-varying flows. Although the model is quite simple, our current theoretical understanding of equilibrium behaviour in this model is rather limited, and many fundamental questions remain open. I'll survey a few aspects, such as a structural characterization by Koch and Skutella, and quite general existence and uniqueness results by Cominetti, Correa and Larré. In the second part of the talk I'll discuss a recent result (joint work with Roberto Cominetti and Jose Correa) where we resolve one simple-sounding question: do queue lengths remain bounded in the equilibria under natural necessary conditions?

3.18 Scheduling and Optimization Problems in the Wild

Yvonne-Anne Pignolet (ABB Corporate Research - Baden-Dättwil, CH)

With the ongoing trend of increased automation and digitalization, the scheduling and networking algorithms that cope with the growing amount of data produced by industrial plants and processes rise in importance. From factory automation, operating power systems to mining, many automation systems have a scheduling component and due to the many interdependencies graph-theoretic and networking aspects are abundant as well.

In this talk I will present some of the challenges we have worked on at ABB Corporate research in the past. They include real-time communication in substations, stator winding optimizations and workforce scheduling.

3.19 Deterministic Discrepancy Minimization via the Multiplicative Weight Update Method

Thomas Rothvoss (University of Washington - Seattle, US)

Joint work of Avi Levy, Harishchandra Ramadas, Thomas Rothvoss
Main reference Avi Levy, Harishchandra Ramadas, Thomas Rothvoss: "Deterministic Discrepancy Minimization via the Multiplicative Weight Update Method", in Proc. of the Integer Programming and Combinatorial Optimization - 19th International Conference, IPCO 2017, Waterloo, ON, Canada, June 26-28, 2017, Proceedings, Lecture Notes in Computer Science, Vol. 10328, pp. 380–391, Springer, 2017.
URL http://dx.doi.org/10.1007/978-3-319-59250-3_31

A well-known theorem of Spencer shows that any set system with n sets over n elements admits a coloring of discrepancy $O(\sqrt{n})$. While the original proof was non-constructive, recent progress brought polynomial time algorithms by Bansal, Lovett and Meka, and Rothvoss. All those algorithms are randomized, even though Bansal's algorithm admitted a complicated derandomization.

We propose an elegant deterministic polynomial time algorithm that is inspired by Lovett-Meka as well as the Multiplicative Weight Update method. The algorithm iteratively updates a fractional coloring while controlling the exponential weights that are assigned to the set constraints.

A conjecture by Meka suggests that Spencer's bound can be generalized to symmetric matrices. We prove that $n \times n$ matrices that are block diagonal with block size q admit a coloring of discrepancy $O(\sqrt{n} \cdot \sqrt{\log(q)})$. Bansal, Dadush and Garg recently gave a randomized algorithm to find a vector x with entries in $\{-1, 1\}$ with $\|Ax\|_\infty \leq O(\sqrt{\log n})$ in polynomial time, where A is any matrix whose columns have length at most 1. We show that our method can be used to deterministically obtain such a vector.

3.20 Clustering with an Oracle

Barna Saha (University of Massachusetts, US)

Joint work of Arya Mazumdar, Barna Saha
Main reference Arya Mazumdar, Barna Saha: "Clustering with an oracle", in Proc. of the 54th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2016, Monticello, IL, USA, September 27-30, 2016, pp. 738–739, IEEE, 2016.
URL http://dx.doi.org/10.1109/ALLERTON.2016.7852305

Suppose, we are given $V = \{1, 2, \ldots, n\} \equiv [n]$, a set of n points, that can be clustered into k parts $V_i, i = 1, \ldots, k; V_i \cap V_j = \emptyset, \forall i \neq j$; the subsets $V_i \subset [n]$ and k are unknown to us. There is an oracle that can answer any pair-wise queries $V \times V \rightarrow \{\pm 1\}$, where a query answer of $+1$ for $(u, v) \in V \times V$ indicates u and v belong to the same cluster, and -1 indicates they do not. How many such queries are necessary and/or sufficient to find the clusters exactly?

3.21 Interactive Communication with Multiple Parties, or Scheduling with Noise and Feedback

Jared Saia (University of New Mexico, US)

A group of n users want to run a distributed protocol Π over a network where communication occurs via private point-to-point channels. Unfortunately, an adversary, who knows Π, is able to maliciously flip bits on the channels. Can we efficiently simulate Π in the presence of such an adversary?

We show that this is possible, even when L, the number of bits sent in Π, and T, the number of bits flipped by the adversary are not known in advance. In particular, we show how to create a robust version of Π that 1) fails with probability at most ϵ, for any $\epsilon > 0$; and 2) sends soft-$O(L + T)$ bits, where the soft-O notation hides a $\log(n(L + T)/\epsilon)$ term multiplying L.

We believe that algorithms for this problem can be viewed as "scheduling with noise" in the following sense. In each time step it is possible for a "good" node to send a bit of communication over a channel, and it is also possible for the adversary to jam that channel. Given the worst case actions of the adversary, we would like to schedule all necessary communication as efficiently as possible. A major open problem is to obtain results with a "makespan" of $O(L + T)$ instead of soft-$O(L + T)$.

3.22 Queueing in the Mist: Buffering and Scheduling with Limited Knowledge

Gabriel Scalosub (Ben Gurion University - Beer Sheva, IL)

Joint work of Itamar Cohen, Gabriel Scalosub
Main reference Itamar Cohen, Gabriel Scalosub: "Queueing in the mist: Buffering and scheduling with limited knowledge", in Proc. of the 25th IEEE/ACM International Symposium on Quality of Service, IWQoS 2017, Vilanova i la Geltrú, Spain, June 14-16, 2017, pp. 1–6, IEEE, 2017.
URL http://dx.doi.org/10.1109/IWQoS.2017.7969126

Managing queues and scheduling with bounded buffers are among the most fundamental problems in computer networking. Although traditionally it is often assumed that all the properties of each packet are known immediately upon arrival, various real life scenarios render such assumptions invalid. We study some buffering and scheduling problems where traffic characteristics only become known after some preliminary processing. As opposed to having stochastic assumptions on the underlying traffic, we use an adversarial model, and present algorithms and lower bounds for such settings.

3.23 Routing and Scheduling in Hybrid Networks

Christian Scheideler (Universität Paderborn, DE)

Hybrid networks are networks in which the nodes have different communication modes. Cell phones, for example, can communicate via the cellular infrastructure or via their Wifi interfaces. Communication via Wifi interfaces has the advantage that there is no limit (other than the bandwidth and battery constraints) on the amount of data that can be exchanged while the amount of data that can be transferred at a reasonable rate via long-range links using the cellular infrastructure or satellite is limited (by some data plan) or costly. However, routing and scheduling in Wifi networks is challenging, particularly because the topology of the network might be messy and not under the control of the nodes, while the nodes would be able to set up arbitrary overlay networks on top of the cellular infrastructure, but this comes at a price. In abstract terms, we are given a network of cheap links of arbitrary topology that is not under the control of the nodes and potentially changing, but in addition to that the nodes have the ability to build arbitrary overlay networks of costly links that are under the control of the nodes. How can these overlays be used effectively in order to solve routing and scheduling problems for the cheap network? I will give an overview of recent results on that problem.

3.24 Recent Advances for Online Machine Minimization

Kevin Schewior (University of Chile, CL)

License Creative Commons BY 3.0 Unported license
© Kevin Schewior
Joint work of Lin Chen, Nicole Megow, Kevin Schewior
Main reference Lin Chen, Nicole Megow, Kevin Schewior: "The Power of Migration in Online Machine Minimization", in Proc. of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016, pp. 175–184, ACM, 2016.
URL http://dx.doi.org/10.1145/2935764.2935786
Main reference Lin Chen, Nicole Megow, and Kevin Schewior: "An O(log m)-competitive algorithm for online machine minimization". In Proc. of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms (SODA '16), Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 155-163, 2016.
URL https://dl.acm.org/citation.cfm?id=2884447

Online Machine Minimization (Phillips et al., STOC 1997) is a fundamental online scheduling problem: Jobs arrive over time at their release dates and need to be scheduled preemptively on parallel machines until their deadlines. Here, the performance of a schedule is measured in the number of machines it requires. It has been an open question since the introduction of the problem whether constant-competitive algorithms exist, until the results presented in this talk even if the number m of machines that the offline optimum uses is fixed. We review the recent progress that has been made on this problem. We introduce the new lower-bound technique by Chen, Megow, and Schewior (SODA 2016) and the algorithms that arose from it: We present the $O(\log m)$-competitive algorithm from the same paper, the modification that yields an $O(\log m/\log\log m)$-competitive algorithm (Azar and Cohen, OR Letters 2018), and the additional building block used to obtain an $O(\log\log m)$-competitive algorithm (Im et al., RTSS 2017). We complement these results with improved upper bounds for special cases and an impossibility result for non-migratory algorithms (Chen, Megow, and Schewior, SPAA 2016).

3.25 On Packet Scheduling with Adversarial Jamming and Speedup

Jirí Sgall (Charles University - Prague, CZ)

License Creative Commons BY 3.0 Unported license
© Jirí Sgall
Joint work of Martin Böhm, Lukasz Jez, Jirí Sgall, Pavel Veselý
Main reference Martin Böhm, Lukasz Jez, Jirí Sgall, Pavel Veselý: "On Packet Scheduling with Adversarial Jamming and Speedup", in Proc. of the Approximation and Online Algorithms - 15th International Workshop, WAOA 2017, Vienna, Austria, September 7-8, 2017, Revised Selected Papers, Lecture Notes in Computer Science, Vol. 10787, pp. 190–206, Springer, 2017.
URL http://dx.doi.org/10.1007/978-3-319-89441-6_15

In Packet Scheduling with Adversarial Jamming packets of arbitrary sizes arrive over time to be transmitted over a channel in which instantaneous jamming errors occur at times chosen by the adversary and not known to the algorithm. The transmission taking place at the time of jamming is corrupt, and the algorithm learns this fact immediately. An online algorithm maximizes the total size of packets it successfully transmits and the goal is to develop an algorithm with the lowest possible asymptotic competitive ratio, where the additive constant may depend on packet sizes.

Our main contribution is a universal algorithm that works for any speedup and packet sizes and, unlike previous algorithms for the problem, it does not need to know these properties in advance. We show that this algorithm guarantees 1-competitiveness with speedup 4, making it the first known algorithm to maintain 1-competitiveness with a moderate speedup in the general setting of arbitrary packet sizes. We also prove a lower bound of $\phi + 1 \approx 2.618$ on the speedup of any 1-competitive deterministic algorithm, showing that our algorithm is close to the optimum.

Additionally, we formulate a general framework for analyzing our algorithm locally and use it to show upper bounds on its competitive ratio for speedups in $[1, 4)$ and for several special cases, recovering some previously known results, each of which had a dedicated proof. In particular, our algorithm is 3-competitive without speedup, matching the algorithm and the lower bound of Jurdzinski et al.

3.26 Generalizing the Kawaguchi-Kyan Bound to Stochastic Parallel Machine Scheduling

Martin Skutella (TU Berlin, DE)

Joint work of Sven Jäger, Martin Skutella
Main reference Sven Jäger, Martin Skutella: "Generalizing the Kawaguchi-Kyan Bound to Stochastic Parallel Machine Scheduling", in Proc. of the 35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France, LIPIcs, Vol. 96, pp. 43:1–43:14, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.
URL http://dx.doi.org/10.4230/LIPIcs.STACS.2018.43

Minimizing the sum of weighted completion times on m identical parallel machines is one of the most important and classical scheduling problems. For the stochastic variant where processing times of jobs are random variables, Möhring, Schulz, and Uetz (1999) presented the first and still best known approximation result achieving, for arbitrarily many machines, performance ratio $1 + \frac{1}{2}(1 + \Delta)$, where Δ is an upper bound on the squared coefficient of variation of the processing times. We prove performance ratio $1 + \frac{1}{2}(\sqrt{2} - 1)(1 + \Delta)$ for the same underlying algorithm – the Weighted Shortest Expected Processing Time (WSEPT) rule. For the special case of deterministic scheduling (i.e., $\Delta = 0$), our bound matches the tight performance ratio $\frac{1}{2}(1 + \sqrt{2})$ of this algorithm (WSPT rule), derived by Kawaguchi and Kyan in a 1986 landmark paper.

3.27 Partitioning into Quadruples

Frits C. R. Spieksma (TU Eindhoven, NL)

Joint work of Thomas Erlebach, Annette Ficker, Matúš Mihalak, Frits C. R. Spieksma

We consider a clustering problem where $4k$ given vectors need to be partitioned into k clusters of four vectors each. A cluster of four vectors is called a quad, and the cost of a quad is the sum of the component-wise maxima of the four vectors in the quad. The problem is to partition the given $4k$ vectors into k quads with minimum total cost. We analyze the worst-case behavior of a straightforward matching-based algorithm, and prove that this algorithm is a 3/2-approximation algorithm for this clustering problem. We also analyze the performance of this algorithm on special cases of the problem.

3.28 Locality and Distributed Scheduling

Jukka Suomela (Aalto University, FI)

In this talk I will look at scheduling problems from the perspective of distributed computing. It turns out that there are two commonly used interpretations of the term distributed computing, and at first they seem to take us in the opposite directions in comparison with the classical centralised setting:

(1) "Big data perspective." The key resource is computation, and distributed computing helps us in comparison with centralised algorithms. We have access to the computing power of multiple computers, and we can use them to speed up computation and to solve large instances that do not fit in the memory of one computer. Relevant models of computing include the MapReduce model and the bulk synchronous parallel model (BSP), and there is a close connection to parallel computing (e.g. the PRAM model).

(2) "Network algorithms perspective." The key resource is communication, and the distributed setting is an additional challenge in comparison with centralised algorithms. We have a large number of nodes, each of them is initially aware of its own part of the input, and each node needs to compute its own part of the solution (e.g. when to switch on), ideally without any global coordination. Relevant models of computing include the LOCAL model and the CONGEST model, and there is a close connection to sublinear-time algorithms (e.g. property testing) and communication complexity.

In this talk I will mainly focus on the new challenges that we have when we look at scheduling problems from perspective (2). Here the overarching theme is locality: how far does a node need to see in order to find its own part of the solution? We will discuss ways of defining scheduling problems in this setting, and how they are connected to the main challenges of the field (e.g. symmetry breaking) and current research themes (e.g. classification of so-called locally checkable problems).

However, I will also look at ways of bridging the gap between perspectives (1) and (2); I will discuss unifying models such as the congested clique model, which is closely related to both the BSP model and the CONGEST model.

3.29 A Constant-factor Approximation Algorithm for the Asymmetric Traveling Salesman Problem

Ola Svensson (EPFL - Lausanne, CH)

Joint work of Ola Svensson, Jakub Tarnawski, László A. Végh
Main reference Ola Svensson, Jakub Tarnawski, László A. Végh: "A Constant-Factor Approximation Algorithm for the Asymmetric Traveling Salesman Problem", CoRR, Vol. abs/1708.04215, 2017.
URL http://arxiv.org/abs/1708.04215

We give a constant-factor approximation algorithm for the asymmetric traveling salesman problem. Our approximation guarantee is analyzed with respect to the standard LP relaxation, and thus our result confirms the conjectured constant integrality gap of that relaxation.

Our techniques build upon the constant-factor approximation algorithm for the special case of node-weighted metrics. Specifically, we give a generic reduction to structured instances that resemble but are more general than those arising from node-weighted metrics. For those instances, we then solve Local-Connectivity ATSP, a problem known to be equivalent (in terms of constant-factor approximation) to the asymmetric traveling salesman problem.

3.30 Greed is Good (for Scheduling under Uncertainty)

Marc Uetz (University of Twente, NL)

Joint work of Varun Gupta, Benjamin Moseley, Marc Uetz, Qiaomin Xie
Main reference Varun Gupta, Benjamin Moseley, Marc Uetz, Qiaomin Xie: "Stochastic Online Scheduling on Unrelated Machines", in Proc. of the Integer Programming and Combinatorial Optimization - 19th International Conference, IPCO 2017, Waterloo, ON, Canada, June 26-28, 2017, Proceedings, Lecture Notes in Computer Science, Vol. 10328, pp. 228–240, Springer, 2017.
URL http://dx.doi.org/10.1007/978-3-319-59250-3_19

The talk addresses a classical problem in the area of scheduling, namely minimizing the total weighted completion time of non-preemptive jobs on a set of unrelated machines. Uncertainty enters the model by assuming that job processing times are stochastic. In order to obtain constant factor approximation algorithms for that problem, prior work required sophisticated linear or convex programming relaxations for the assignment of jobs to machines. In contrast, we analyze a purely combinatorial, online algorithm. Maybe surprisingly, we show how to derive performance bounds for that algorithm that are of the same order of magnitude as those of earlier work, while our results are the first for an online setting. The analysis is based on dual fitting techniques.

4 Open problems

4.1 Algorithmic Problems Combining Network Design, Routing, and Scheduling

Michael Dinitz (Johns Hopkins University - Baltimore, US)

Some next-generation networks will have the ability to dynamically reconfigure their topology. Specific proposals for such networks have recently appeared in the datacenter architecture literature, and this is already happening in optical wide-area networks. Particularly in the WAN setting, scheduling large flows becomes an important problem. But in a graph where we control the topology dynamically, we get problems which combine three different types of algorithmic questions: Network design, routing, and scheduling. There is already systems work in this area, but the theoretical work typically ignores at least one of these aspects (usually routing or scheduling). Combining all three should lead to some interesting questions.

4.2 Distributed Computation on Speed Scalable Processors

Kirk Pruhs (University of Pittsburgh, US)

Consider a problem P that one wants to solve in some distributed model of computation. For example, computing the minimum spanning tree in the congest model. Assume that there is an algorithm A for this problem that runs in time T. If one assumes that each node in the n node network uses a unit of energy per unit time, then this algorithm would use nT units of energy. Now assume that the nodes are speed scalable, that is they can run at any speed s. Running at speed s means that a node can send s rounds of messages in unit time. But running as speed s costs s^2 unit of energy. Then algorithm A can be converted into an algorithm that runs in T/s time using energy nTs^2. If A is optimal then this is an optimal time/energy trade-off if all the processors run at the same speed all the time. So the question is, for your favorite problem P, like minimum spanning, can one ever beat this time/energy trade-off by allowing processors to run at different speeds at different times. Intuitively this is asking whether solving P inherently involves geographically and temporally local bottlenecks, that can be ameliorated with speed scaling, whether bottlenecks can be evenly spread out temporally and geographically.

4.3 Online Buffers on the Line

Rob van Stee (Universität Siegen, DE)

Main reference Matthias Englert: "The reordering buffer problem on the line revisited", SIGACT News, Vol. 49(1), pp. 67–72, 2018.
URL http://dx.doi.org/10.1145/3197406.3197418
Main reference Iftah Gamzu, Danny Segev: "Improved online algorithms for the sorting buffer problem on line metrics", ACM Trans. Algorithms, Vol. 6(1), pp. 15:1–15:14, 2009.
URL http://dx.doi.org/10.1145/1644015.1644030

A single server needs to serve requests (points) appearing on the line by visiting them. It can store k requests in a buffer, and the goal is to minimize the total distance traveled by the server. There is a relatively simple online algorithm which is also the best algorithm known, online or offline (!). It works by partitioning the line geometrically and is $O(\log n)$-competitive. Improve it. The best known lower bound is less than 3.

Participants

- Susanne Albers
TU München, DE
- Yossi Azar
Tel Aviv University, IL
- Nikhil Bansal
TU Eindhoven, NL
- Sanjoy K. Baruah
University of North Carolina at Chapel Hill, US
- Syamantak Das
IIIT – New Dehli, IN
- Jelena Diakonikolas
Boston University, US
- Michael Dinitz
Johns Hopkins University – Baltimore, US
- Fritz Eisenbrand
EPFL – Lausanne, CH
- Thomas Erlebach
University of Leicester, GB
- Klaus-Tycho Foerster
Universität Wien, AT
- Seth Gilbert
National University of Singapore, SG
- Anupam Gupta
Carnegie Mellon University, US
- Magnús M. Halldórsson
Reykjavik University, IS
- Sungjin Im
University of California – Merced, US
- Thomas Kesselheim
TU Dortmund, DE
- Samir Khuller
University of Maryland – College Park, US
- Valerie King
University of Victoria, CA
- Fabian Daniel Kuhn
Universität Freiburg, DE
- Amit Kumar
Indian Inst. of Technology – New Dehli, IN
- Alberto Marchetti-Spaccamela
Sapienza University of Rome, IT
- Nicole Megow
Universität Bremen, DE
- Danupon Nanongkai
KTH Royal Institute of Technology, SE
- Yvonne-Anne Pignolet
ABB Corporate Research – Baden-Dättwil, CH
- Kirk Pruhs
University of Pittsburgh, US
- Thomas Rothvoss
University of Washington – Seattle, US
- Barna Saha
University of Massachusetts, US
- Jared Saia
University of New Mexico, US
- Gabriel Scalosub
Ben Gurion University – Beer Sheva, IL
- Christian Scheideler
Universität Paderborn, DE
- Kevin Schewior
University of Chile - Santiago, CL
- Jiri Sgall
Charles University – Prague, CZ
- David Shmoys
Cornell University, US
- René Sitters
VU University of Amsterdam, NL
- Martin Skutella
TU Berlin, DE
- Frits C. R. Spieksma
TU Eindhoven, NL
- Clifford Stein
Columbia University, US
- Leen Stougie
VU University of Amsterdam, NL
- Jukka Suomela
Aalto University, FI
- Ola Svensson
EPFL – Lausanne, CH
- Marc Uetz
University of Twente, NL
- Rob van Stee
University of Leicester, GB
- Anke van Zuylen
College of William and Mary – Williamsburg, US
- Jose Verschae
Pontifical Catholic University of Chile - Santiago, CL
- Tjark Vredeveld
Maastricht University, NL

Report from Dagstuhl Seminar 18102

Dynamic Traffic Models in Transportation Science

Edited by

Roberto Cominetti[1], Tobias Harks[2], Carolina Osorio[3], and Britta Peis[4]

1 **Adolfo Ibáñez University, CL,** `roberto.cominetti@uai.cl`
2 **Universität Augsburg, DE,** `tobias.harks@math.uni-augsburg.de`
3 **MIT – Cambridge, US,** `osorioc@mit.edu`
4 **RWTH Aachen, DE,** `britta.peis@oms.rwth-aachen.de`

Abstract

Traffic assignment models are crucial for traffic planners to be able to predict traffic distributions, especially, in light of possible changes of the infrastructure, e.g., road constructions, traffic light controls, etc. The starting point of the seminar was the observation that there is a trend in the transportation community (science as well as industry) to base such predictions on complex computer-based simulations that are capable of resolving many elements of a real transportation system. On the other hand, within the past few years, the theory of dynamic traffic assignments in terms of equilibrium existence and equilibrium computation has not matured to the point matching the model complexity inherent in simulations. In view of the above, this interdisciplinary seminar brought together leading scientists in the areas *traffic simulations*, *algorithmic game theory* and *dynamic traffic assignment* as well as people from industry with strong scientific background who identified possible ways to bridge the described gap.

Seminar March 4–9, 2018 – https://www.dagstuhl.de/18102
2012 ACM Subject Classification Networks → Network algorithms, Networks → Network performance evaluation, Theory of computation → Design and analysis of algorithms
Keywords and phrases Algorithm and complexity of traffic equilibrium computation, dynamic traffic assignment models, Simulation and network optimization
Digital Object Identifier 10.4230/DagRep.8.3.21
Edited in cooperation with Daniel Schmand, Veerle Timmermans

1 Executive Summary

Tobias Harks (Universität Augsburg, DE)
Roberto Cominetti (Adolfo Ibáñez University, CL)
Carolina Osorio (MIT – Cambridge, US)
Britta Peis (RWTH Aachen, DE)

Traffic assignment models play an important role for traffic planners to predict traffic distributions, especially, in light of possible changes of the infrastructure, e.g., road constructions, traffic light controls, speed limits, tolls, etc. The prevailing *mathematical* approaches used in the transportation science literature to predict such distributions can be roughly classified into static traffic assignment models based on aggregated static multi-commodity flow formulations and dynamic traffic assignment (DTA) models based on the methodology of flows over time. While static models have seen several decades of development and practical

use, they abstract away too many important details and, thus, become less attractive. On the other hand, dynamic models are known to be notoriously hard to analyze in terms of existence, uniqueness and computability of dynamic equilibria.
In light of the prevailing computational difficulties for realistic-sized networks, the systematic optimization of such networks (e.g., by designing the network infrastructure, link tolls, or traffic light controls) becomes even more challenging as the resulting mathematical programs with equilibrium constraints contain already in the lower level presumably "hard" optimization-, complementarity- or variational inequality problems; not to speak of the resulting optimization problem for the first level.
On the other hand, there is a trend in the transportation science community to use *large-scale computer-based microsimulations* for predicting traffic distributions. The striking advantage of microscopic simulations over DTA models is that the latter usually ignores the feedback of changing network conditions on user behavior dimensions such as flexible departure time choice, mode choice, activity schedule choice, and such. Current simulation tools integrate all these dimensions and many more. The increasing model complexity, however, is by far not matched by the existing theory of dynamic traffic assignments.

The seminar brought together leading researchers from three different communities – Simulations (SIM), Dynamic Traffic Assignment (DTA) and Algorithmic Game Theory (AGT). This years seminar was centered around three topics:

- *Horizontal queueing models.* Most of the static traffic assignment models assume that queues can occur, but do not take up any physical space. In order to make the current models more realistic one should assume that queues might effect traffic on other nearby road segments, thus, include possible spill-back effects.
- *Oligopolistic competition.* With the rise of autonomous vehicles new routing decisions need to be made. As a novel aspect, individual vehicles might to be interested in selfishly optimizing their routes, but cooperate with other vehicles using the same software in order to decrease the average journey time.
- *Risk-averse travelers.* Current static traffic models often assume that each player is rational, and has the sole purpose of minimizing travel time or distance. However, the exact travel time of many routes might be uncertain at the moment of departure. Hence, travelers might stick to a more predictable route and might be unwilling to explore possibly better alternatives.

Again, the seminar was a big success both in terms of stimulating new and very fruitful collaborations. We got enthusiastic feedback from many participants which is also reflected in the survey conducted by Dagstuhl.

2 Table of Contents

3 Overview of Talks

3.1 Equilibrium Computation in Atomic Splittable Routing Games with Convex Cost Functions

Umang Bhaskar (TIFR Mumbai, IN)

We present polynomial-time algorithms as well as hardness results for equilibrium computation in atomic splittable routing games, for the case of general convex cost functions. These games model traffic in freight transportation, market oligopolies, data networks, and various other applications. An atomic splittable routing game is played on a network where the edges have traffic-dependent cost functions, and player strategies correspond to flows in the network. A player can thus split it's traffic arbitrarily among different paths. While many properties of equilibria in these games have been studied, efficient algorithms for equilibrium computation are known for only two cases: if cost functions are affine, or if players are symmetric. Neither of these conditions is met in most practical applications. We present two algorithms for routing games with general convex cost functions on parallel links. The first algorithm is exponential in the number of players, while the second is exponential in the number of edges; thus if either of these is small, we get a polynomial-time algorithm. These are the first algorithms for these games with convex cost functions. We further give evidence that in general networks equilibrium computation may be hard, showing that given input C it is NP-hard to decide if there exists an equilibrium where every player has cost at most C.

3.2 Travel behavior variability and congestion feedback in iterated transport simulations

Gunnar Flötteröd (KTH – Royal Institute of Technology – Stockholm, SE)

Most transport micro-simulations rely on stochastic travel behavior models. Their stochasticity may be a meaningful modeling feature given expected (i.e. converged) network conditions. During simulation transients, however, their variability may be amplified by network congestion feedback and lead to convergence problems. Means to control travel behavior variability throughout the simulation process are hence considered.

3.3 The Price of Stability of Weighted Congestion Games

Martin Gairing (University of Liverpool, GB)

Joint work of George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, Paul Spirakis
Main reference George Christodoulou, Martin Gairing, Yiannis Giannakopoulos, Paul G. Spirakis: "The Price of Stability of Weighted Congestion Games", CoRR, Vol. abs/1802.09952, 2018.
URL http://arxiv.org/abs/1802.09952

We give exponential lower bounds on the Price of Stability (PoS) of weighted congestion games with polynomial cost functions. Our lower bound asymptotically matches the the upper bound on the Price of Anarchy upper bound. We further show that the PoS remains exponential even for singleton games. More generally, we also provide a lower bound on the PoS of approximate Nash equilibria. All our lower bounds extend to network congestion and hold for mixed and correlated equilibria as well.

On the positive side, we give a general upper bound on the PoS of approximate Nash equilibria, which is sensitive to the range W of the player weights. We do this by explicitly constructing a novel approximate potential function, based on Faulhaber's formula, that generalizes Rosenthal's potential in a continuous, analytic way. From the general theorem, we deduce two interesting corollaries. First, we derive the existence of an approximate pure Nash equilibrium with PoS at most $(d + 3)/2$; the equilibrium's approximation parameter ranges from $O(1)$ to $d+1$ in a smooth way with respect to W. Secondly, we show that for unweighted congestion games, the PoS of α-approximate Nash equilibria is at most $(d + 1)//alpha$.

3.4 Great Tolls: How to Induce Optimal Flows under Strategic Link Operators

Cristóbal Guzmán (Pontifical Catholic University of Chile, CL)

Joint work of José Correa, Cristóbal Guzmán, Thanasis Lianeas, Evdokia Nikolova, Marc Schröder

Network pricing games provide a framework for modeling real-world settings with two types of strategic agents: owners (operators) of the network and users of the network. Owners of the network post a price for usage of the link they own so as to a attract users and maximize profit; users of the network select routes based on price and level of use by other users. We point out that an equilibrium in these games may not exist, may not be unique and may induce an arbitrarily inefficient network performance.

Our main result is to observe that a slight regulation on the network owners market solves all three issues above. Specifically, if the authority could set appropriate caps (upper bounds) on the tolls (prices) operators can charge then: the game among the link operators has a unique and strong Nash equilibrium and the users' game results in a Wardrop equilibrium that achieves the optimal total delay. We call any price vector with these properties a great set of tolls. As a secondary objective, we want to compute great tolls that minimize total users' payments and provide a linear program that does this. We obtain multiplicative approximation results compared to the optimal total users' payments for arbitrary networks with polynomial latencies of bounded degree, while in the single-commodity case we obtain a bound that only depends on the topology of the network. Lastly, we show how the same mechanism of setting appropriate caps on the allowable prices extends to the model of elastic demands.

3.5 Efficient Black-Box Reductions for Separable Cost Sharing

Anja Huber (Universität Augsburg, DE)

Joint work of Tobias Harks, Martin Hoefer, Anja Huber, Manuel Surek

In cost sharing games with delays, a set of agents jointly allocates a finite subset of resources. Each resource has a fixed cost that has to be shared by the players, and each agent has a non-shareable player-specific delay for each resource. A prominent example is uncapacitated facility location (UFL), where facilities need to be opened (at a shareable cost) and clients want to connect to opened facilities. Each client pays a cost share and his non-shareable physical connection cost. Given any profile of subsets allocated by the agents, a separable cost sharing protocol determines cost shares that satisfy budget balance on every resource and separability over the resources. Moreover, a separable protocol guarantees existence of pure Nash equilibria in the induced strategic game for the agents.

In this talk, we study separable cost sharing protocols in several general combinatorial domains. We provide black-box reductions to reduce the design of a separable cost-sharing protocol to the design of an approximation algorithm for the underlying cost minimization problem. In this way, we obtain new separable cost-sharing protocols in games based on arbitrary player-specific matroids, single-source connection games without delays, and connection games on n-series-parallel graphs with delays. All these reductions are efficiently computable – given an initial allocation profile, we obtain a cheaper profile and separable cost shares turning the profile into a pure Nash equilibrium. Hence, in these domains any approximation algorithm can be used to obtain a separable cost sharing protocol with a price of stability bounded by the approximation factor.

3.6 Computing all Wardrop Equilibria parametrized by the Flow Demand

Max Klimm (HU Berlin, DE)

Joint work of Max Klimm, Philipp Warode

We develop an algorithm that computes for a given undirected or directed network with flow-dependent piece-wise linear edge cost functions all Wardrop equilibria as a function of the flow demand. Our algorithm is based on Katzenelson's homotopy method for electrical networks. The algorithm uses a bijection between vertex potentials and flow excess vectors that is piecewise linear in the potential space and where each linear segment can be interpreted as an augmenting flow in a residual network. The algorithm iteratively increases the excess of one or more vertex pairs until the bijection reaches a point of non-differentiability. Then, the next linear region is chosen in a Simplex-like pivot step and the algorithm proceeds. We first show that this algorithm correctly computes all Wardrop equilibria in undirected single-commodity networks along the chosen path of excess vectors. We then adapt our algorithm to also work for discontinuous cost functions which allows to model directed edges and/or edge capacities. Our algorithm is output-polynomial in non-degenerate instances where the solution curve never hits a point where the cost function of more than one edge becomes non-differentiable. For degenerate instances we still obtain an output-polynomial algorithm computing the linear segments of the bijection by a convex program. The latter technique also allows to handle multiple commodities.

3.7 Microscopic simulation of taxicabs and autonomous vehicles with MATSim

Kai Nagel (TU Berlin, DE)

From a transportation planning perspective, autonomous vehicles (AVs) can be seen as a mode of transport: Rather than, say, walking to the car and then driving, or walking to the pt stop, waiting, and then boarding, one would request an AV, wait for it, then board, etc. This can be simulated microscopically, where microscopic means that there are as many synthetic avartars as there are persons and vehicles in reality. One interesting problem is real-time dispatch. I will show results from simulations that reach up to synthetically replacing one million privately owned cars in Berlin by a fleet of 100'000 AVs, and then derive some policy recommendations based on these simulations. These include results for making that fleet electric, and thus (locally) free of emissions.

The presentation uses results obtained with and by Michał Maciejewski, Joschka Bischoff and Gregor Leich.

3.8 Equilibria in the fluid queueing model

Neil Olver (VU University of Amsterdam, NL)

Joint work of Roberto Cominetti, Omar Larré, José Correa, Neil Olver
Main reference Roberto Cominetti, José R. Correa, Omar Larré: "Dynamic Equilibria in Fluid Queuing Networks", CoRR, Vol. abs/1401.6914v1, 2014
URL https://arxiv.org/abs/1401.6914

I'll discuss the fluid queueing model, introduced by Vickrey in '69. It is probably the simplest model that plausibly captures the notion of time-varying flows. Although the model is quite simple, our current theoretical understanding of equilibrium behaviour in this model is rather limited, and many fundamental questions remain open. I'll survey a few aspects, such as a structural characterization by Koch and Skutella, and quite general existence and uniqueness results by Cominetti, Correa and Larré. In the second part of the talk I'll discuss a recent result (joint work with Roberto Cominetti and Jose Correa) where we resolve one simple-sounding question: do queue lengths remain bounded in the equilibria under natural necessary conditions?

3.9 Optimization and Simulation

Carolina Osorio (MIT – Cambridge, US)

Simulation-based dynamic network models have the potential to provide a detailed (e.g., disaggregate) description of demand and of supply. Nonetheless, unlike analytical models, they are computationally inefficient to evaluate and their use to address transportation optimization problems is a challenge. In this talk we present simulation-based optimization algorithms that enable the direct and efficient use of simulation-based dynamic network models for optimization. The main idea is to embed within the algorithms information from the analytical network models. The latter provide analytical problem-specific structural information, which enables the design of computationally efficient algorithms. We present case studies for high-dimensional intricate (e.g., non-convex) optimization problems, such as OD calibration, congestion pricing and signal control. We present results for large-scale networks, including Berlin, Singapore and Manhattan.

3.10 (Approximate) Equilibrium Computation for Games

Rahul Savani (University of Liverpool, GB)

This talk will give an overview of equilibrium computation for games. We will cover different representations of games, exact and approximate solutions concepts, and the important algorithmic approaches and complexity-theoretic results. We will place the material in the context of traffic models where possible, and will mention some related open questions.

3.11 When is selfish routing bad? The price of anarchy in light and heavy traffic

Marco Scarsini (LUISS Guido Carli – Rome, IT)

Joint work of Riccardo Colini-Baldeschi, Roberto Cominetti, Panayotis Mertikopoulos, Marco Scarsini

This paper examines the behavior of the price of anarchy as a function of the traffic inflow in nonatomic congestion games with multiple origin-destination (O/D) pairs. Empirical studies in real-world networks show that the price of anarchy is close to 1 in both light and heavy traffic, thus raising the question: can these observations be justified theoretically? We first show that this is not always the case: the price of anarchy may remain a positive distance away from 1 for all values of the traffic inflow, even in simple three-link networks with a single O/D pair and smooth, convex costs. On the other hand, for a large class of cost functions (including all polynomials), the price of anarchy does converge to 1 in both heavy and light traffic, and irrespective of the network topology and the number of O/D pairs in the network. We also examine the rate of convergence of the price of anarchy, and we show that it follows a power law whose degree can be computed explicitly when the network's cost functions are polynomials.

3.12 Earliest Arrival Transshipments in Networks With Multiple Sinks

Miriam Schlöter (TU Berlin, DE)

Joint work of Miriam Schlöter, Martin Skutella

We study a classical flow over time problem that captures the essence of evacuation planning: Given a network with capacities and transit times on the arcs and sources/sinks with supplies/demands, an earliest arrival transshipment (EAT) sends the supplies from the sources to the sinks such that the amount of flow which has reached the sinks is maximized for every point in time simultaneously. So far, a lot of effort has been put into the development of algorithms for computing earliest arrival transshipments in dynamic networks with only a single sink, as in such networks earliest arrival transshipments do always exist. Regarding earliest arrival transshipments in networks with multiple sinks not much is known, aside from the fact that EATs do in general not exist in general networks. In particular, there is no PSPACE algorithm known for computing EATs in case of existence and also the complexity of deciding whether an earliest arrival transshipment solving a given transshipment problem in a multiple sink network does exist is still unknown.

This talk concentrates on our latest results regarding EATs in networks with multiple sinks: In particular, we formulate the first PSPACE algorithm that decides whether a given tight earliest arrival transshipment problem in a general network has solution and computes the EAT in case of existence. We show that in case of existence an earliest arrival transshipment can essentially be determined as a convex combination of special flows over time while minimizing a suitably defined submodular function. The solution we achieve has additionally the nice structural property of being a generalized temporally repeated flow. Additionally, we settle the complexity question by showing that in multiple sink networks it is NP-hard to decide whether an earliest arrival transshipment solving a given problem does exist.

3.13 Selfish Network Creation with Wardrop Followers

Daniel Schmand (RWTH Aachen, DE)

Joint work of Daniel Schmand, Marc Schröder, Alexander Skopalik

We study the following network design game. The game is set in two stages. In the first stage some players, called providers, aim to maximize their profit individually by investing in bandwidth on edges of a given graph. The investment yields a new graph on which Wardrop followers,abs/1401.6914 called users, travel from their source to their sink through the network. The cost for any user on an edge follows market principles and is dependent on the demand for that edge and the supplied bandwidth. The profit of the providers depends on the total utilization of their edges, the current price for their edges and the bandwidth. We analyze the existence, uniqueness and efficiency of Nash Equilibria in the described game.

3.14 Network Congestion Games are Robust to Variable Demand

Marc Schröder (RWTH Aachen, DE)

Joint work of José Correa, Ruben Hoeksma, Marc Schröder
URL https://www.dii.uchile.cl/ jcorrea/papers/Conferences/CHS2017.pdf

Network congestion games have provided a fertile ground for the algorithmic game theory community. Indeed, many of the pioneering works on bounding the efficiency of equilibria use this framework as their starting point. In recent years, there has been an increased interest in studying randomness in this context though the efforts have been mostly devoted to understanding what happens when link latencies are subject to random shocks. Although this is an important practical consideration, it is not the only source of randomness in network congestion games. Another important source is the inherent variability of the demand that most practical networks suffer from. Therefore in this paper we look at the basic non-atomic network congestion game with the additional feature that demand is random. Our main result in this paper is that under a very natural equilibrium notion, in which the basic behavioral assumption is that users evaluate their expected cost according to the demand they experience in the system, the price of anarchy of the game is actually the same as that in the deterministic demand game. Moreover, the result can be extended to the more general class of smooth games.

3.15 Computing Efficient Nash Equilibria in Congestion Games

Guido Schäfer (CWI – Amsterdam, NL)

Joint work of Pieter Kleer, Guido Schäfer

Congestion games constitute an important class of games which capture many applications in network routing, resource allocation and scheduling. Rosenthal (1973) established the existence of pure Nash equilibria in congestion games by exhibiting an exact potential function whose local minima coincide with the set of pure Nash equilibria of the game. We investigate structural properties which allow us to efficiently compute global minima of Rosenthal's potential function. To this aim, we use a polyhedral description to represent the strategy sets of the players and identify two general properties which are sufficient for our result to go through. In addition, we show that the resulting Nash equilibria provide attractive social cost approximation guarantees. Our contributions thus provide an efficient algorithm to find an approximately optimal Nash equilibrium for a large class of polytopal congestion games. Joint work with Pieter Kleer (CWI).

3.16 Simple, distributed, and powerful – improving local search for distributed resource allocation and equilibrium computation

Alexander Skopalik (Universität Paderborn, DE)

Joint work of Vipin Ravindran Vijayalakshmi, Alexander Skopalik

Congestion games constitute an important class of games to model resource allocation by different users such as in traffic networks. We study the approximation ratio of local optima in these games. However, we allow for that the cost functions used during the local search procedure may be different from the overall objective function. Our analysis exhibits an interesting method to choose these cost functions to obtain a number of different results:

1. As computing an exact or even approximate pure Nash equilibrium is in general PLS-complete, Caragiannis et al. [FOCS 2011] presented a polynomial-time algorithm that computes $(2 + \epsilon)$ – approximate pure Nash equilibria for games with linear cost functions and further results for polynomial cost functions. We show that this factor can be improved to $1.67 + \epsilon$ by a seemingly simple modification of their algorithm using our technique.
2. Bilo and Vinci [EC 2016] presented an algorithm to compute load depended taxes the improve the price of anarchy e.g. for linear game from 2.5 to 2. Our methods yield slightly weaker results, e.g., 2.1 for linear games. However, our tax functions are locally computable an independent of the actually instance of the game. There algorithm is a centralized algorithm and sensitive to changes of the instance such as e.g. the number of players.
3. Computing an optimal allocation in congestion games is NP-hard. The best known centralized approximation algorithm is due to Makarychev and Srividenko [FOCS14]. Again, our technique does not quite match there bounds but offers a modified local search procedure as a much simpler alternative which can easily be implemented in a distributed fashion.

3.17 Multiplicative Pacing Equilibria in Auction Markets

Nicolás Stier-Moses (Facebook – Menlo Park, US)

Joint work of Vincent Conitzer, Christian Kroer, Eric Sodomka, Nicolás Stier-Moses
Main reference Vincent Conitzer, Christian Kroer, Eric Sodomka, Nicolás E. Stier Moses: "Multiplicative Pacing Equilibria in Auction Markets", CoRR, Vol. abs/1706.07151, 2017.
URL http://arxiv.org/abs/1706.07151

Budgets play a significant role in real-world sequential auction markets such as those implemented by Internet companies. To maximize the value provided to auction participants, spending is smoothed across auctions so budgets are used for the best opportunities. Motivated by a mechanism used in practice by several companies, this paper considers a smoothing procedure that relies on *pacing multipliers*: on behalf of each bidder, the auction market applies a factor between 0 and 1 that uniformly scales the bids across all auctions. Reinterpreting this process as a game between bidders, we introduce the notion of *pacing equilibrium*, and prove that they are always guaranteed to exist. We demonstrate through examples that a market can have multiple pacing equilibria with large variations in several natural objectives. Although we show that computing either a social-welfare-maximizing or a revenue-maximizing pacing equilibrium is NP-hard, we present a mixed-integer program (MIP) that can be used to find equilibria optimizing several relevant objectives. We use the MIP to provide evidence that: (1) equilibrium multiplicity occurs very rarely across several families of random instances, (2) static MIP solutions can be used to improve the outcomes achieved by a dynamic pacing algorithm with instances based on a real-world auction market, and (3) for our instances, bidders do not have an incentive to misreport bids or budgets provided there are enough participants in the auction.

3.18 Queues in the cyclically time-expanded network model

Martin Strehler (TU Cottbus, DE)

We present extensions of the cyclically time-expanded network model for traffic signal optimization to cope with queues. The model has already shown its value for simultaneous traffic assignment and traffic signal optimization by computing competitive solutions. However, due to the lack of a proper first in-first out property and a limited storage capacity, several traffic situations were inaccurately modeled. In this talk, we present two constructions to handle overload, spill-back, and queues. We have a special focus on applications, namely the optimization of signals for exceptional mega events and (passive) transit signal priority, i.e., accelerating public transport by optimizing signal settings.

3.19 Non-separable costs and their impact on (a class of) user equilibrium algorithms

Chris Tampère (KU Leuven, BE)

This presentation discusses convergence properties of a class of algorithms for computing user equilibrium (UE) flows on large networks with multiple origins and destinations. The class derives from the formulation of the UE problem as a fixed point between a demand and a supply submodel. A general structure of this class of algorithms is presented. Typically, the more efficient algorithms decompose the UE problem into several sub-problems, usually defined in terms of origins and/or destinations that use common link subsets. In time-dependent problems, another decomposition dimension is the departure time interval. UE is achieved by iteratively solving a (fully or partially converged) step of each sub-problem. This involves adding or removing links from the sub-problem, deciding direction and size of the step (=swap) towards sub-problem-UE, and updating the link costs as a consequence of this swap. It is shown how efficient convergence of the combined sub-problems depends strongly (i) on the sequence of solving the sub-problems, and (ii) on whether the link cost updates are computed simultaneously after deciding swaps for all sub-problems or sequentially after each sub-problem. Moreover, non-separability of link costs (e.g. due to intersection or spillback interactions, or in time-dependent flows by definition) affects strongly whether a chosen sequence for solving sub-problems and updating link costs convergences efficiently, if at all.

This is illustrated on an algorithmic solution for the Vickrey bottleneck model and a simple 2-route dynamic UE problem. These insights hint at improved solution sequences for these types of problems, which is ongoing work.

3.20 Effects of fixed-time vs. traffic-adaptive signal control on the total travel time in the user equilibrium in agent-based transport simulations

Theresa Thunig (TU Berlin, DE)

Joint work of Nico Kühnel, Kai Nagel, Theresa Thunig
URL https://svn.vsp.tu-berlin.de/repos/public-svn/publications/vspwp/2018/18-02/KuehnelThunigNagel2018LaemmerImpl.pdf

When travelers choose a route to their destination, they also take signal green times into account: When a signal delays their travel time significantly, they will try out a different route on the next day. With a system-wide optimization of fixed-time signal plans, one can benefit from this user reaction. Fixed-time signals can intentionally make certain routes unattractive to force the user equilibrium (UE) towards the system optimum (SO) route pattern, i.e. to improve total travel time. In contrast, traffic-adaptive signals that control traffic based on local sensor data are usually de-centralized and not able to control system-wide effects. They aim to locally minimize delay at intersections which supports the UE routes. This can reduce waiting times and, thus, total travel time, but does not necessarily force the SO.

In this talk we will compare the effects of fixed-time and traffic-adaptive signals in the transport simulation MATSim. At first, the small network of Braess' paradox is analyzed. It is shown that in contrast to fixed-time signals, local delay-minimizing signals are not able to force the SO. Still, they are able to react to unexpected demand changes and improve waiting times, which will be shown in the second part of the talk with a real-world application of the city of Cottbus, Germany.

3.21 Oligopolistic Competitive Packet Routing

Veerle Timmermans (Maastricht University, NL)

Joint work of Laura Vargas Koch, Björn Tauer, Britta Peis, Veerle Timmermanns

We study a game-theoretic variant of packet routing. Oligopolistic competitive packet routing games model situations in which traffic is routed in discrete units through a communication network over time. In contrast to classical packet routing, we are lacking a central authority to decide on an oblivious routing protocol. Instead, selfish acting decision makers ("players") control a certain amount of traffic each, which needs to be sent as fast as possible from a player-specific origin to a player-specific destination through a commonly used network. The network is represented by a directed graph, each edge of which being endowed with a transit time, as well as a capacity bounding the number of traffic units entering an edge simultaneously. Additionally, a priority policy on the set of players is publicly known with respect to which conflicts at intersections are resolved. We prove the existence of a pure

Nash equilibrium and show that it can be constructed by sequentially computing an integral earliest arrival flow for each player. Moreover, we derive several tight bounds on the price of anarchy and the price of stability in single source games.

3.22 Competitive Packet Routing With Edge Priorities

Laura Vargas Koch (RWTH Aachen, DE)

License Creative Commons BY 3.0 Unported license
© Laura Vargas Koch
Joint work of Robert Scheffler, Martin Stehler, Laura Vargas Koch
Main reference Robert Scheffler, Martin Strehler, Laura Vargas Koch: "Nash equilibria in routing games with edge priorities", CoRR, Vol. abs/1803.00865, 2018.
URL http://arxiv.org/abs/1803.00865

We present a game-theoretic variant of packet routing. In contrast to classical packet routing, we are lacking a central authority. Instead, selfish acting players route from a source to a sink and try to reach the sink as fast as possible. The network is represented by a directed graph, each edge of which being endowed with a transit time, as well as a capacity bounding the number of traffic units entering an edge simultaneously. A very natural way to motivate competitive packet routing arises from traffic. Here it makes sense to define priorities on the edges (main roads are preferred over smaller roads). This means if more player want to enter an edge than it is possible, the players coming from the main road are preferred. We present an algorithm computing a certain class of equilibria in these games, mistrustful equilibria and we analyze their properties. Further we examine the connection to earliest arrival flows.

4 Open problems

4.1 The Inefficiency of Wardrop Routing with Uncertain Demand

Daniel Schmand (RWTH Aachen, DE), Anja Huber (Universität Augsburg, DE), and Veerle Timmermans (Maastricht University, NL)

License Creative Commons BY 3.0 Unported license
© Daniel Schmand, Anja Huber, and Veerle Timmermans

We discussed open problems related to uncertainty of the demand in selfish routing games. We consider a model of a Wardrop-game in which the overall demand in the system is drawn from a probability distribution. We seek to understand the expected inefficiency of Wardrop equilibria in the system. An easy example shows that current lower bounds on the price of anarchy (i.e. 4/3 for linear functions) are not robust to changes in the overall demand. That is, we could already show that the price of anarchy decreases in the famous Pigou example. Further analysis of the inefficiency of Wardrop equilibria in games with uncertain demand is a very interesting research topic and remains open.

4.2 Stochastic Atomic Congestion Games

Marc Schröder (RWTH Aachen, DE), Roberto Cominetti (Adolfo Ibáñez University, CL), Marco Scarsini (LUISS Guido Carli – Rome, IT), and Nicolás Stier-Moses (Facebook – Menlo Park, US)

We consider an atomic network congestion game in which each player is drawn independently with a given probability to play the game. Roughgarden (2015) and Correa, Hoeksma and Schroder (2017) already showed that the upper bound on the price of anarchy of these games is the same as for the deterministic game in which the set of players is fixed and known. It is however open whether this upper bound is tight, and/or can be characterized in terms of the probability of playing. Another potential direction would be to see whether atomic congestion games with independent, but identically distributed random weights can be shown to be potential games.

4.3 Complexity of Mixed Equilibria in Potential Games

Alexander Skopalik (Universität Paderborn, DE), Martin Gairing (University of Liverpool, GB), and Rahul Savani (University of Liverpool, GB)

We discussed the complexity theoretic characterization of mixed Nash equilibria in Potential or Congestion Games. It is known that this problem is in CLS (and in the intersection of PPAD and PLS) but it is not known whether they are hard for this class. We also considered an alternative complexity class EOPL and discussed possible approaches for polynomial time algorithms for some special cases such as e.g. two player network congestion games. This problem is particularly interesting as it is not known whether finding a pure Equilibrium is PLS-hard. Only for so called restricted network games that has been shown. Although we were not able to solve the original problem, our discussion resulted in interesting new insights and observations.

4.4 Alternatives to Wardrop equilibrium and Convergence of iterated transport simulations

Dave Watling (University of Leeds, GB), Gunnar Flötteröd (KTH – Royal Institute of Technology – Stockholm, SE), and Chris Tampère (KU Leuven, BE)

Although Wardrop Equilibrium (WE) is a famous reference model for transportation, many researchers in transportation science focus instead on models that represent heterogeneity/unknown factors using probability distributions. This gives rise to a different model, Stochastic User Equilibrium. In a sense SUE generalises WE, since in its most general form SUE admits both continuous and discrete probability distributions for the random error terms in travel utilities, including the case of probability mass 1 at a zero error (i.e. WE). As SUE is used relatively widely in transportation science, and (it seems) hardly at all in algorithmic game theory, it seems an area of potential future investigation for linking the two communities, especially in terms of studying how results for Nash-type games might extend to these alternative models.

The SUE mechanism for path choice is also attractive for use in non-equilibrium situations, in particular iterated simulations, since conditional path choice is unique and dispersed, but nevertheless can lead to computational problems. In this session we considered some recent development in utilising bounded forms of the logit choice model in an SUE context, as opposed to the unbounded models typically used. This has important implications in that for unbounded models all routes are used, whereas for bounded models we obtain something analogous to WE in a division between used and unused routes. The idea we pursued was to utilise such bounded models in iterated simulations, where the bounding could be systematically varied over the course of the simulation – so in an early stage we allow greater variance in order to generate a working set of routes that are sufficient to serve the demand, but in later stages would systematically reduce the variance so as to reduce the exploration and instead focus on re-balancing between routes. Some initial calculations have suggested a potential functional form to describe how the bound might be systematically altered across iterations.

In our proposal for open problems we also discussed two other issues that we believed to merit further exploration. The first was the problem of non-separable cost/latency functions, as might occur (for example) at an uncontrolled priority intersection, when the delay to a minor turning movement is mainly controlled by the flow of a different (major/priority) flow. The second was the issue of what were referred to as *anti-congestion games*, where the cost/latency of an alternative may be decreasing in the flow on that alternative (as may occur due to social/imitation/mass effects, or due to economies of scale).

Participants

- Umang Bhaskar
TIFR Mumbai, IN
- Roberto Cominetti
Adolfo Ibáñez University, CL
- Gunnar Flötteröd
KTH – Royal Institute of Technology – Stockholm, SE
- Martin Gairing
University of Liverpool, GB
- Cristóbal Guzmán
Pontifical Catholic University of Chile, CL
- Tobias Harks
Universität Augsburg, DE
- Martin Hoefer
Goethe-Universität – Frankfurt am Main, DE
- Anja Huber
Universität Augsburg, DE
- Max Klimm
HU Berlin, DE
- Ekkehard Köhler
TU Cottbus, DE
- Kai Nagel
TU Berlin, DE
- Neil Olver
VU University of Amsterdam, NL
- Carolina Osorio
MIT – Cambridge, US
- Britta Peis
RWTH Aachen, DE
- Rahul Savani
University of Liverpool, GB
- Marco Scarsini
LUISS Guido Carli – Rome, IT
- Guido Schäfer
CWI – Amsterdam, NL
- Heiko Schilling
TomTom International – Amsterdam, NL
- Miriam Schlöter
TU Berlin, DE
- Daniel Schmand
RWTH Aachen, DE
- Marc Schröder
RWTH Aachen, DE
- Alexander Skopalik
Universität Paderborn, DE
- Nicolás Stier-Moses
Facebook – Menlo Park, US
- Sebastian Stiller
TU Braunschweig, DE
- Martin Strehler
TU Cottbus, DE
- Chris Tampère
KU Leuven, BE
- Theresa Thunig
TU Berlin, DE
- Veerle Timmermans
Maastricht University, NL
- Laura Vargas-Koch
RWTH Aachen, DE
- Bernhard von Stengel
London School of Economics, GB
- Dave Watling
University of Leeds, GB

Report from Dagstuhl Seminar 18111

Loop Optimization

Edited by
Sebastian Hack[1], Paul H. J. Kelly[2], and Christian Lengauer[3]

1 **Universität des Saarlandes, DE,** hack@cs.uni-saarland.de
2 **Imperial College London, UK,** p.kelly@imperial.ac.uk
3 **Universität Passau, DE,** christian.lengauer@uni-passau.de

Abstract

This report documents the programme of Dagstuhl Seminar 18111 "Loop Optimization". The seminar brought together experts from three areas: (1) model-based loop optimization, chiefly, in the polyhedron model, (2) rewriting and program transformation, and (3) metaprogramming and symbolic evaluation. Its aim was to review the 20+ years of progress since the Dagstuhl Seminar 9616 "Loop Parallelization" in 1996 and identify the challenges that remain.

Seminar March 11–16, 2018 – https://www.dagstuhl.de/18111
2012 ACM Subject Classification Theory of computation → Semantics and reasoning, Computing methodologies → Symbolic and algebraic manipulation, Computing methodologies → Parallel computing methodologies, Software and its engineering → Software notations and tools, Software and its engineering → Compilers
Keywords and phrases Autotuning, dependence analysis, just-in-time (JIT), loop parallelization, parallel programming, polyhedron model
Digital Object Identifier 10.4230/DagRep.8.3.39

1 Executive Summary

Sebastian Hack
Paul H. J. Kelly
Christian Lengauer

Motivation

Loop optimization is at the heart of effective program optimization – even if the source language is too abstract to contain loop constructs explicitly as, e.g., in a functional style or a domain-specific language. Loops provide a major opportunity to improve the performance of a program because they represent compactly a large volume of accessed data and executed instructions. Because the clock frequency of processors fails to continue to grow (end of Dennard scaling), the only way in which the execution of programs can be accelerated is by increasing their throughput with a compiler: by increasing parallelism and improving data locality. This puts loop optimization in the center of performance optimization.

Context

The quick and easy way to optimize a loop nest, still frequently used in practice, is by restructuring the source program, e.g., by permuting, tiling or skewing the loop nest. Beside being laborious and error-prone, this approach favors modifications that can be easily recognized and carried out, but which need not be the most suitable choice. A much better approach is to search automatically for optimization options in a mathematical model of the iteration space, in which all options are equally detectable and the quality of each option can be assessed precisely.

Recently, the polyhedral compilation community has produced a set of robust and powerful libraries that contain a variety of algorithms for the manipulation of Presburger sets, including all standard polyhedral compilation techniques. They can be incorporated in a program analysis to make other compiler optimizations more precise and powerful, like optimizers and code generators for domain-specific languages, or aggressive optimizers for high-performance computing.

Polyhedral loop optimization relies on strict constraints on the structure of the loop nest and may incur a computationally complex program analysis, based on integer linear programming. The optimization problems become much simpler when information at load or run time is available, i.e., the optimization is done just-in-time. Also, the search for the best optimization can be supported by other techniques, e.g., auto-tuning, machine learning or genetic algorithms. While these techniques are all fully automatic, engineering of software with robust performance characteristics requires programmers to have some level of explicit control over the data distribution and communication costs. However, manually optimized code is far too complicated to maintain. Thus, a major research area concerns the design of tools that allow developers to guide or direct analysis (e.g., via dependence summaries or domain-specific code generation) and optimization (e.g., via directives, sketches and abstractions for schedules and data partitioning).

Goal

The goal of this seminar was to generate a new synergy in loop optimization research by bringing together representatives of the major different schools of thought in this field. The key unifying idea is to formulate loop optimization as a mathematical problem, by characterizing the optimization space and objectives with respect to a suitable model.

One school is focused on reasoning about scheduling and parallelization using a geometric, "polyhedral", model of iteration spaces which supports powerful tools for measuring parallelism, locality and communication – but which is quite limited in its applicability.

Another major school treats program optimization as program synthesis, for example by equational rewriting, generating a potentially large space of variants which can be pruned with respect to properties like load balance and locality. This approach has flourished in certain application domains, but also suffers from problems with generalization.

A third family of loop optimization approaches tackles program optimization through program generation and symbolic evaluation. Generative approaches, such as explicit staging, support programmers in taking explicit control over implementation details at a high level of abstraction.

The seminar explored the interplay of these various loop optimization techniques and fostered the communication in the wide-ranging research community of model-based loop optimization. Participants represented the various loop optimization approaches but also application domains in high-performance computing.

Conclusions

The seminar succeeded in making the participants aware of common goals and relations between different approaches. Consensus emerged on the potential and importance of tensor contractions and tensor comprehensions as an intermediate representation. There was also some excitement in connecting the classical dependence-based optimization with newly emerging ideas in deriving parallel algorithms from sequentially-dependent code automatically. Guided automatic search and inference turned out to be a dominant theme. Another important insight was that the optimization criteria currently in use are often too coarse-grained and do not deliver satisfactory performance. More precise hardware models are needed to guide optimization. This will require a closer collaboration with the performance modeling and engineering community.

It was agreed that publications and collaborations fueled by the seminar will acknowledge Schloss Dagstuhl.

2 Table of Contents

3 Overview of Talks

30-min talk slots covered the programme until Thursday mid-afternoon; four keynote presentations took up two slots each. The latter part of the seminar was devoted to the planning of future collaborations. A list of talks follows in the order in which they were presented.

3.1 On the Design of Intermediate Representations for Loop Nest Optimization (Keynote)

Albert Cohen (ENS – Paris, FR)

Joint work of Ulysse Beaugnon, Albert Cohen, Tobias Grosser, Jacques Pienaar, Chandan Reddy, Vivek Sarkar, Jun Shirako, Nicolas Vasilache, Sven Verdoolaege, Oleksandr Zinenko, Jie Zhao

The associated slides focus on advanced affine scheduling heuristics and attempt to derive some lessons from the experience of customizing a rather generic framework (polyhedral compilation) to a specific purpose (harnessing the multi-level parallelism and memory hierarchy of modern multi-processors).

3.2 Beyond the Polyhedral Model (Keynote)

Paul Feautrier (ENS – Paris, FR)

The polyhedral model is a powerful tool for program analysis, verification, optimization and parallelization. However, its applicability is restricted to regular programs with only scalar and arrays as the only data structures, affine subscripts, and counted loops with affine bounds as the only control constructs. As it now stands, the model is especially applicable to linear algebra and signal processing algorithms. Many extensions to the model were proposed since its inception, as for instance using enabling transformations or detection of static control parts: SCoPs. I feel that the time has come for a more drastic approach: the creation and exploration of new models, either more powerful than the polyhedral model or directed at other families of algorithms.

I will first review early attempts at the creation of other models, as for instance the polynomial model, the flowchart model, or models based on the theory of formal languages. A promising research direction is the use of approximate methods, applying concepts borrowed from the abstract interpretation theory. The flowchart model is a first attempt at combining both types of tools.

Proof assistants like Coq may help in the construction of models. However, Coq is not a solver, and hence is not suited for the analysis of existing programs (the "dusty deck" problem). It may best be used for the construction of frameworks guaranteeing bug-free programming.

3.3 Static Instruction Scheduling for High Performance on Limited Hardware

Alexandra Jimborean (Uppsala University – Uppsala, SE)

Joint work of Kim-Anh Tran, Trevor E. Carlson, Konstantinos Koukos, Magnus Själander, Vasileios Spiliopoulos, Stefanos Kaxiras, Alexandra Jimborean

Complex out-of-order (OoO) processors have been designed to overcome the restrictions of outstanding long-latency misses at the cost of increased energy consumption. Simple, limited OoO processors are a compromise in terms of energy consumption and performance, as they have fewer hardware resources to tolerate the penalties of long-latency loads. In the worst case, these loads may stall the processor entirely.

We present Clairvoyance, a compiler-based technique that generates code able to hide memory latency and better utilize simple OoO processors. By clustering loads found across basic block boundaries, Clairvoyance overlaps the outstanding latencies to increase memory-level parallelism. We show that these simple OoO processors, equipped with the appropriate compiler support, can effectively hide long-latency loads and achieve performance improvements for memory-bound applications. To this end, Clairvoyance tackles (i) statically unknown dependencies, (ii) insufficient independent instructions, and (iii) register pressure.

Clairvoyance achieves a geomean execution time improvement of 14% for memory-bound applications, on top of standard O3 optimizations, while maintaining compute-bound applications' high performance.

3.4 FPGAs vs. GPUs: How to Beat the Beast

Frank Hannig (Friedrich-Alexander University Erlangen-Nürnberg – Erlangen, DE)

Joint work of Frank Hannig, Richard Membarth, M. Akif özkan, Oliver Reiche, Moritz Schmid, Jürgen Teich

Graphics processing units (GPUs) and field-programmable gate arrays (FPGAs) are often employed as accelerators for computationally intensive applications. Both architectures have an abundant number of computational resources in common, albeit of different type and granularity. However, when it comes to programming, FPGAs and GPUs are greatly different. To bridge this sort of programmability gap, domain-specific languages (DSLs) are a promising solution, since they separate algorithm development from parallelization and low-level implementation details on an actual target architecture. Thus, DSLs offer high flexibility among heterogeneous hardware targets, such as CPUs and GPUs. With the recent rise of high-level synthesis (HLS) tools, such as Xilinx Vivado HLS and Intel/Altera OpenCL, FPGAs become tame. Particularly in the domain of image processing, applications often come with stringent requirements regarding performance, energy efficiency, and power, for which FPGAs have been proven to be among the most suitable architectures.

We present the Hipacc framework, a DSL and source-to-source compiler for image processing. We show that domain knowledge can be captured to generate tailored implementations for C-based high-level software from a common high-level DSL description targeting FPGAs. Our approach includes FPGA-specific memory architectures for handling point and local operators, as well as several high-level transformations. We evaluate our approach by comparing the resulting hardware accelerators to GPU implementations, generated from the same DSL source code.

3.5 Structured Parallel Programming: Code Generation by Rewriting Algorithmic Skeletons

Michel Steuwer (University of Glasgow – Glasgow, GB)

Joint work of Lift team

I will argue that we should structure our parallel programs using higher-level primitives which encode higher-level semantics explicitly. I will draw on similarities to "structured programming" which introduced concepts such as 'while' and 'for' loops as an answer to the software crisis in the late 1960s. The arguments made by Dahl, Dijkstra, and Hoare are still valid today but we need to revisit them in the context of parallelism.

I will discuss our work on the Lift project (http://www.lift-project.org) which introduces a set of data-parallel high-level primitives, called algorithmic skeletons, which are used to express programs in an abstract purely functional way. A rich exploration process optimizes these programs by rewriting the high-level program into low-level programs which encode implementations and optimization decisions explicitly. I will present encouraging performance result and sketch our ongoing research.

3.6 Rewriting with an Index-Based Intermediate Representation

Charisee Chiw (University of Chicago – Chicago, US)

Joint work of Charisee Chiw, John Reppy

The EIN representation is a hybrid design that embeds expression trees into a normalized SSA representation. The representation can concisely and clearly express the indexing of different arguments involved in a large computation as a single term. Invariant terms can be moved in and out of loops with simple pattern matching, rewriting, and analysis. In the future, we want to examine the entire computation at an earlier stage of compilation. Then we can develop a more advanced approach to loop optimization.

We designed an intermediate representation (IR), EIN, for tensor math. This design preserves the useful properties of the SSA representation, while providing flexibility in the specification of tensor and tensor-field operations. The key property of this design is that it allows reference to indices in the body of EIN expressions, while also providing a compact representation of the nested iteration that is implicit in the definition of tensors and tensor fields. A single EIN term consists of a tensor or field variable binding and indices. The index binding and ordering describes the shape and sampling of each argument. EIN supports standard linear algebra operations on tensors, such as addition, subtraction, the dot product, as well as other tensor operations such as the double-dot (colon) product, the outer product, and trace. One can think of the index space as defining an n-deep loop nest over the index variables, where the EIN expression is the loop body that defines scalar components of the result.

By examining the indices in EIN terms, we can do some modest loop invariant code motion. Each summation operator represents a loop nest that will be unrolled later in the compilation. Thus, moving operations outside the summation can avoid subsequent code duplication. This transformation is essentially loop-invariant code hoisting for the special case of summations. When there are nested summations, our method applies additional analysis to see whether the summation can be converted into the product of independent summations. We identify loop invariants by looking at the indices. We are currently wondering whether we can expand on this simple idea to improve code generation. If we take a step back and look at the mathematical structure of the larger computation, we can leverage this knowledge into efficient code.

3.7 Synthesis of Modular Parallelism for Nested Loops

Victor Nicolet (University of Toronto – Toronto, CA)

Joint work of Victor Nicolet, Azadeh Farzan

Parallelizing loops is notoriously difficult when there are true dataflow dependencies that forbid parallelization using the wealth of sound code transformation techniques studied over the years. In previous work, we tackled the problem of finding divide-and-conquer parallel implementations of sequential loops. The idea was to first lift the loop by discovering and adding new computation of information that is redundant in the sequential program, but necessary for an efficient divide-and-conquer parallelization. This approach was specific to a class of loops that traverse a linear iteration space and compute a function of a sequence, and does not generalize to nested loops over multidimensional data. We propose a modular approach to analyze these by treating each loop nest separately. First, we encapsulate the inner loop to abstract its effect in the outer loop. Then, we explain how parallelizing the outer loop, that uses only this encapsulation, can give us a parallel implementation of the initial loop nest. I will talk about how this lets us leverage our existing automatic parallelization solution for single loops to one for more sophisticated loops.

3.8 Multidimensional Scheduling in the Polyhedral Model

Louis-Noël Pouchet (Colorado State University – Fort Collins, US)

Compositions of loop transformations are represented in the polyhedral compilation framework using scheduling functions, represented as integer matrices. To find a good schedule, two approaches can be employed: (a) computing each row of the scheduling matrix one at a time, typically solving one integer linear program (ILP) for each row, as done for example by the Pluto algorithm; and (b) computing all rows at once, using a single but usually more complex ILP, as done in the Ponos tool.

We will focus on one-shot multidimensional scheduling, where a single ILP is formulated to find the entire scheduling matrix. We will present the generic space of legal schedules implemented in Ponos, and show how to quickly design scheduling objectives, e.g., for specific parallelization or data locality patterns. We will present an interactive interface to the Ponos tool, to facilitate the design of new ILP-based multidimensional scheduling techniques.

3.9 Iterative Schedule Optimization for Parallelization in the Polyhedron Model

Stefan Ganser (University of Passau – Passau, DE)

Joint work of Stefan Ganser, Armin Größlinger, Norbert Siegmund, Sven Apel, Christian Lengauer

The polyhedron model is a powerful model to identify and apply systematically loop transformations that improve data locality (e.g., via tiling) and enable parallelization. In the polyhedron model, a loop transformation is, essentially, represented as an affine function. Well-established algorithms for the discovery of promising transformations are based on performance models. These algorithms have the drawback of not being easily adaptable to the characteristics of a specific program or target hardware. An iterative search for promising loop transformations is more easily adaptable and can help to learn better models. We present an iterative optimization method in the polyhedron model that targets tiling and parallelization. The method enables either a sampling of the search space of legal loop transformations at random or a more directed search via a genetic algorithm. For the latter, we propose a set of novel, tailored reproduction operators. We evaluate our approach against existing iterative and model-driven optimization strategies. We compare the convergence rate of our genetic algorithm to that of random exploration. Our approach of iterative optimization outperforms existing optimization techniques in that it finds loop transformations that yield significantly higher performance. If well configured, then random exploration turns out to be very effective and reduces the need for a genetic algorithm.

3.10 The Polyhedral Model Beyond Static Compilation, Affine Functions and Loops

Philippe Clauss (University of Strasbourg – Strasbourg, FR)

The polyhedral model has been proven to be a powerful framework for automatic analysis & transformation of loops. However, it suffers from strong limitations since it is mostly limited to “Fortran like” loops and linear transformations. We show that its scope and efficiency can be extended either by extending its mathematical objects to polynomials and algebraic expressions, or thanks to its use at run time. Run-time (speculative) polyhedral compilation opens new challenging opportunities for handling more general (non-linear) loops or handling non-loop programs that have a looping behavior, while algebraic expressions provide greater effectiveness and a larger scope of loop transformations. We illustrate the presentation with the speculative polyhedral optimization framework Apollo, and with the Trahrhe expressions (“Ehrhart” read backwards), which are the inverse of ranking Ehrhart polynomials. Some uses of Trahrhe expressions are presented: collapsing of non-rectangular loops and algebraic tiling.

3.11 Efficient Online Tuning of Accelerator Mapping Decisions

Philip Pfaffe (KIT – Karlsruhe, DE)

Automatic parallelization is a key component in state-of-the-art industry-grade compilers as well as research. With polyhedral optimization, even parallelization for heterogeneous platforms is realizable within a well-structured framework. Nevertheless, achieving optimal or even near-optimal performance with automatic transformations is hard, courtesy of a multitude of degrees of freedom inherent to platform specific optimization and parallelization. Fortunately, autotuning is an already established technique to deal with optimizing performance in the presence of high-dimensional search spaces. In this positional talk, we will examine the benefits that online-autotuning can offer to heterogeneous parallelization, and discuss promising future directions in tightly coupling autotuners with parallelizing compilers.

3.12 Loop Execution Time Modeling

Julian Hammer (Friedrich-Alexander University Erlangen-Nürnberg – Erlangen, DE)

The modeling of loop execution time allows us to evaluate optimization potentials prior to run time. If found in closed form, they may even directly yield optimization parameters, derived from hardware and code properties. We will present the layer condition cache model and ongoing efforts in instruction-level out-of-order execution time prediction. We believe that the integration of such models in compilers and auto-tuning tools will largely reduce – or even eradicate – test-runs, decrease guessing and support informed choices during build time.

3.13 Compiling Tensor Algebra for Finite-Element Computations

Lawrence Mitchell (Imperial College – London, GB)

Joint work of Lawrence Mitchell, Thomas Gibson, David A. Ham, Miklós Homolya, Paul H. J. Kelly, Fabio Luporini, Tianjiao Sun

At the core of a PDE, any library that uses finite elements is a large tensor contraction. Providing a low flop count, highly efficient implementation of this contraction is either devolved to the computational scientist (and then a general-purpose compiler), or else to a domain-specific compiler (and thence to a general-purpose one).

I will talk about the domain-specific compiler, and the optimisation passes, that we use in the Firedrake project (www.firedrakeproject.org), that deliver low algorithmic complexity algorithms on a class of finite elements that exhibit kronecker product structure.

I will then cover some open questions and future research directions, in particular how to extend the code transformation pipeline to incorporate operations on tensors that are not easily expressible as scalar indexed expressions: of particular interest is to widen the applicability to include tensor inverse and determinant calculations.

3.14 Automated Cross-Element Vectorization in Firedrake

Tianjiao Sun (Imperial College – London, GB)

Joint work of Tianjiao Sun, Lawrence Mitchell, David A. Ham, Paul H. J. Kelly

Firedrake is a domain-specific language embedded in Python for numerical solution of partial differential equations (PDEs) using the finite-element method. Firedrake provides the users with a high-level interface to express the problems in a high-level mathematical language while generating efficient low-level code. The internal intermediate representations in this code generation pipeline offer performance optimization opportunities at different levels of abstraction. We present one of the latest developments in Firedrake which enables automated vectorization across elements on unstructured meshes for typical finite-element assembly kernels, so as to address the problem of better performance and hardware utilization on SIMD architectures.

Modern CPUs increasingly rely on SIMD instructions to achieve higher throughput and better energy efficiency. It is therefore important to vectorize sequences of computations in order to sufficiently utilize the hardware today and in the future. This requires the instructions to operate on groups of data that are multiples of the width of the vector lane (e.g., 4 doubles, 8 floats on AVX2 instructions). Finite-element computations usually require the assembly of vectors and matrices which represents differential forms on the domain. This process consists of applying a local assembly kernel to each element, and increment the global data structure with the local contribution. Typical local assembly kernels suffer from issues that often preclude efficient vectorization. These include complicated loop structure, poor data access patterns, and loop trip counts that are not multiples of the vector width. General-purpose compilers often perform poorly in generating efficient, vectorized code for such kernels.

We present a generic and portable solution in Firedrake based on cross-element vectorization. Although vector-expanding the assembly kernel is conceptually clear, it is only enabled by applying a chain of complicated loop transformations. Loo.py is a Python package that defines array-style computations in the integer-polyhedral model and supports a rich family of transformations that operate on this model. In Firedrake, we adapt the form compiler, TSFC, to generate Loo.py kernels for local assembly operations, and systematically generate data gathering and scattering operations across the mesh in PyOP2. Firedrake drives loop transformations using Loo.py from this high-level interface to generate efficient code vectorized across a group of elements which fully utilizes the vector lane. This tool chain automates the tedious and error-prone process of data layout transformation, loop unrolling and loop interchange, while being transparent to the users.

We will present experimental results performed on multiple kernels and meshes. We achieve speedups consistent with the vector architecture available compared to baseline which vectorizes inside the local assembly kernels. The global assembly computations reach tens of percent of hardware peak arithmetic performance.

3.15 Automated Loop Generation for High-Performance Finite Differences (and Beyond)

Fabio Luporini (Imperial College – London, GB)

Joint work of Fabio Luporini, Charles Yount, Mathias Louboutin, Navjot Kukreja, Philipp Witte, Tim Burger, Michael Lange, Felix Herrmann, Gerard Gorman

We present the architecture and performance of Devito, a system to express numerical kernels in high-level mathematical notation. We focus, in particular, on the generation of highly optimized operators for seismic inversion. These involve solving partial differential equations (via finite differences) as well as other non-trivial mathematical operations (e.g., sparse points interpolation). The codes that need to be generated by the Devito compiler are therefore quite complex, including arbitrary, non-perfect nests of regular or irregular loops. We discuss the design of the compiler and the performance of the generated code for production-level seismic operators, showing roofline models for two Intel architectures (Skylake, KNL).

3.16 Implementations of Loop Constructs

Shigeru Chiba (University of Tokyo – Tokyo, JP)

This presentation compares and discusses several implementation techniques of programming language constructs for loops. Programmers need appropriate abstraction for describing their loops. Compiler developers also need it to retrieve optimization hints related to data dependency and other kinds of memory access patterns seen in the loop. Such abstraction will be provided as (built-in) domain-specific data types and operators for programmers. A question is how to implement these data types and operators to deliver good performance. This presentation shows an overview of several techniques including simple object-oriented libraries (also known as shallow embedding), C++ template meta programming, external DSLs developed from scratch, pragmas, deep embedding, and deep reification. Benefits and drawbacks of those techniques are mentioned. An important metric here is the implementation costs of abstraction (the data types and operators) since providing abstraction designed for a smaller application domain will be feasible if its implementation cost is not expensive.

3.17 Loop Iterations – Aligned and/or Pipelined?

Ayal Zaks (Intel and Technion – Haifa, IL)

There are two distinct and complementary transformations that can be applied to the iterations of a loop: aligning them to achieve data-level parallelism, also known as vectorization, and pipelining them according to an iteration initiation interval to achieve instruction-level parallelism at fine grain, and double-buffering to achieve memory-level parallelism at coarse grain. Vectorization, also related to loop coarsening, can handle uniform branches and certain dependencies. Pipelining can handle dependencies but not branches. Both are applied to countable loops, or loops whose trip count is known a few iterations ahead of time.

Aligning iterations is supported by data-parallel heterogeneous programming models such as OpenCL's ND-range, facilitating both SIMD execution and dynamic load balancing across massively parallel GPUs. Pipelining and double-buffering, however, stitches all iterations together and leads to the static allocation of all iterations on one device. As a result, there is a mismatch between data-parallel models and deeply pipelined devices, such as FPGAs, which we seek to resolve.

3.18 Parallelizing Dependent Computations

Madanlal Musuvathi (Microsoft Research – Redmond, US)

Joint work of Madanlal Musuvathi, Mike Barnett, Saeed Maleki, Todd Mytkowicz, Yufei Ding, Daniel Lupei, Charith Mendis, Mathias Peters, Veselin Raychev

Parallelization is often synonymous with identifying independent subcomputations. On the other hand, it is well known that certain dependent computations, such as summing all elements in an array, can be parallelized by using the associativity of the operations involved. I will present our recent work on generalizing this insight to mechanically parallelize computations that appear inherently sequential.

The basic idea is to treat dependences as symbolic unknowns and use techniques inspired by program analysis and symbolic execution to execute dependent computations in parallel. Applications include large-scale stream processing and machine learning.

3.19 Communication-Optimal Loop Tilings (Keynote)

James Demmel (University of California – Berkeley, US)

Joint work of James Demmel, Michael Christ, Grace Dinh, Nicholas Knight, Alex Rusciano, Thomas Scanlon, Katherine Yelick

It is well-known that, given a two-level memory hierarchy with a fast memory of size M, the optimal loop tiling for classical $O(n^3)$ matrix multiplication, $C = A * B$, that minimizes data movement (communication) between fast and slow memory, tiles A, B and C into square tiles of size $\Theta(M^{1/2})$, and achieves a communication lower bound of $\Omega(n^3/M^{1/2})$. We extend this result as follows: Given any perfectly nested set of loops, with any number of arrays accessed in the innermost loop, each of which may have any number of subscripts, where each subscript may be an arbitrary affine function of the loop indices (e.g., $A(i, i-j, i+2*j-3*k+4, \dots))$, we present algorithms for computing (1) a constant e so that $\Omega(\#\text{loop\_iterations}/M^e)$ is a communication lower bound, and (2) an optimal loop tiling that achieves this lower bound. The lower bound assumes any execution order in which the loop bodies are not interleaved, and also that array entries cannot be allocated, used arbitrarily often, and then freed/discarded, without requiring any memory traffic. The proof depends on a recent discrete extension of the well-known Brascamp-Lieb inequality by Terry Tao, Michael Christ and others.

The optimal tiling makes two assumptions:

1. The loop bounds are large enough to fit the optimal tile. When this is not the case (e.g., think of matrix-vector multiply as a special case of matrix-matrix multiply) then it is possible to extend our results to get tighter lower bounds. We illustrate this with Convolutional Neural Nets (CNNs), expressible as seven nested loops, and provide a loop reordering that lowers the communication cost by a greater factor than possible for matrix multiply.
2. Data dependencies between loop iterations permit reordering. In the case of uniform-dependence algorithms, where data dependencies are represented by a finite set of constant distance vectors, it is possible to test whether data dependencies permit reordering. Generalizing the Brascamp-Lieb inequality to derive tighter lower bounds in the presence of dependencies is an open problem.

Time permitting, we can describe extensions of these results to memory hierarchies with multiple levels, and to distributed memory.

3.20 Effective Performance Modeling: A Grand Challenge for Loop Transformations in Compilers

P. Sadayappan (Ohio-State University – Columbus, US)

Joint work of P. Sadayappan, Changwan Hong, Aravind Sukumaran-Rajam, Sriram Krishnamoorthy, Louis-Noël Pouchet, Fabrice Rastello

A fundamental challenge for loop optimization is effective performance modeling. Existing loop optimizers in compilers generally use highly simplified performance models that do not necessarily correlate very well with actual realized performance on the target platform. This is particularly true of polyhedral loop optimization, where linear objective functions enable elegant ILP-based solutions. While transformations based on simplified models may improve performance over a naive baseline version, achieving performance comparable to hand-optimized code or code generated by domain-specific compilers is extremely challenging.

We describe an approach to performance modeling for GPU kernels that used abstract emulation of a small number of thread-blocks of the kernel. Key hardware resources like global memory, shared memory, functional units, etc. are modeled using two parameters: latency and gap (the inverse of throughput). Sensitivity analysis with respect to resource latency/gap parameters is used to predict the bottleneck resource for a given kernel's execution. Bottleneck analysis is in turn used for performance optimization. The approach hold promise in assisting manual code optimization, as well as automated model-driven auto tuning for performance enhancement.

3.21 Polyhedral Expression Propagation

Johannes Doerfert (Saarland University – Saarbrücken, DE)

Polyhedral techniques have proven to be powerful for various optimizations, from automatic parallelization to accelerator programming. At their core, these techniques compute accurate dependences among statement instances in order to apply complex program transformations. Such transformations comprise memory layout or program order modifications by optimizing memory access functions or scheduling functions. However, these approaches treat statements as opaque entities and do not consider changing the structure of the contained expressions or the memory accesses involved.

We present a technique that statically propagates expressions in order to avoid communicating their result via memory. While orthogonal to other polyhedral optimizations, this transformation can be used to enable them. Applied separately, expression propagation can increase parallelism, eliminate temporary arrays, create independent computations and improve cache utilization. It is especially useful for streaming codes that involve temporary arrays and scalar variables.

For multiple image processing pipelines, we achieve portable speedups of up to 21.3x as well as a significant memory reduction compared to a naive parallel implementation. In 6 out of 7 cases, expression propagation outperforms a state-of-the-art polyhedral optimization especially designed for this kind of programs by a factor of up to 2.03x.

3.22 The isl Scheduler

Sven Verdoolaege (Facebook – Paris, FR)

isl is a library for manipulating integer sets and relations bounded by affine constraints, such as those that occur in polyhedral compilation. Next to several generic operations on such objects, isl also supports some operations tailored to polyhedral compilation, including a row-by-row scheduler.

After a general overview of isl focusing on the representation of fundamental concepts, some details are presented about the isl scheduler, including the representation and meaning of the input and the output, the available algorithms and their use of ILP solvers, as well as some issues that have been encountered in practice.

3.23 PolyJIT: Polyhedral Optimization Just in Time

Andreas Simbürger (University of Passau – Passau, DE)

Joint work of Andreas Simbürger, Sven Apel, Armin Größlinger, Christian Lengauer

While polyhedral optimization appeared in mainstream compilers during the past decade, its profitability in scenarios outside its classic domain of linear-algebra programs has remained in question. Recent implementations, such as the LLVM plugin Polly, produce promising speedups, but the re- striction to affine loop programs with control flow known at compile time continues to be a limiting factor. PolyJIT combines polyhedral optimization with multi-versioning at run time, at which one has access to knowledge enabling polyhedral optimization, which is not available at compile time. By means of a fully-fledged implementation of a light-weight just-in-time (JIT) compiler and a series of experiments on a selection of real-world and bench- mark programs, we demonstrate that the consideration of run-time knowledge helps in tackling compile-time violations of affinity and, consequently, offers new opportunities of optimization at run time.

3.24 Organizing Computation for High Performance Graphics & Visual Computing (Keynote)

Jonathan Ragan-Kelley (University of California – Berkeley, US)

Future visual computing applications – from photorealistic real-time rendering, to 4D light field cameras, to pervasive sensing and computer vision – demand orders of magnitude more computation than we currently have. From data centers to mobile devices, performance and energy scaling is limited by locality (the distance over which data has to move, e.g., from nearby caches, far away main memory, or across networks) and parallelism. Because of this, I argue that we should think of the performance and efficiency of an application as determined not just by the algorithm and the hardware on which it runs, but critically also by the organization of computations and data. For algorithms with the same complexity – even the exact same set of arithmetic operations and data – executing on the same hardware, the order and granularity of execution and placement of data can easily change performance by an order of magnitude because of locality and parallelism. To extract the full potential of our machines, we must treat the organization of computation as a first class concern while working across all levels from algorithms and data structures, to compilers, to hardware.

I will present facets of this philosophy in systems I have built for visual computing applications from image processing and vision, to 3D rendering, simulation, optimization, and 3D printing. I will show that, for data-parallel pipelines common in graphics, imaging, and other data-intensive applications, the organization of computations and data for a given algorithm is constrained by a fundamental tension between parallelism, locality, and redundant computation of shared values. I will focus particularly on the Halide language and compiler for image processing, which explicitly separates what computations define an algorithm from the choices of organization which determine parallelism, locality, memory footprint, and synchronization. I will show how this approach can enable much simpler programs to deliver performance often many times faster than the best prior hand-tuned C, assembly, and CUDA implementations, while scaling across radically different architectures, from ARM cores, to massively parallel GPUs, to FPGAs and custom ASICs.

3.25 AnyDSL: A Partial Evaluation System for Programming High-Performance Libraries

Roland Leißa (Saarland University – Saarbrücken, DE)

Joint work of Roland Leißa, Klaas Boesche, Sebastian Hack, Richard Membarth, Arsène Pérard-Gayot, Philipp Slusallek, André Müller, Bertil Schmidt

Nowadays, the computing landscape is becoming increasingly heterogeneous and this trend is currently showing no signs of turning around. In particular, hardware becomes more and more specialized and exhibits different forms of parallelism. For performance-critical codes it is indispensable to address hardware-specific peculiarities. Because of the halting problem, however, it is unrealistic to assume that a program implemented in a general-purpose programming language can be fully automatically compiled to such specialized hardware while still delivering peak performance.

We present AnyDSL. This framework allows to embed a domain-specific language (DSL) into a host language. On the one hand, a DSL offers the application developer a convenient interface; on the other hand, a DSL can serve to specify domain-specific optimizations and effectively map DSL constructs to various architectures. In order to implement a DSL, one usually has to write or modify a compiler. With AnyDSL, though, the DSL constructs are directly implemented in the host language while a partial evaluator removes any abstractions that are required in the implementation of the DSL.

3.26 Tensor Comprehensions: Framework-Agnostic High-Performance Machine Learning Abstractions

Nicolas Vasilache (Facebook – New York City, US)

Deep learning models with convolutional and recurrent networks are now ubiquitous and analyze massive amounts of audio, image, video, text and graph data, with applications in automatic translation, speech-to-text, scene understanding, ranking user preferences, ad placement, etc. Competing frameworks for building these networks such as TensorFlow, Chainer, CNTK, Torch/PyTorch, Caffe1/2, MXNet and Theano, explore different tradeoffs between usability and expressiveness, research or production orientation and supported hardware. They operate on a DAG of computational operators, wrapping high-performance libraries such as CUDNN (for NVIDIA GPUs) or NNPACK (for various CPUs), and automate memory allocation, synchronization, distribution. Custom operators are needed where the computation does not fit existing high-performance library calls, usually at a high engineering cost. This is frequently required when new operators are invented by researchers: such operators suffer a severe performance penalty, which limits the pace of innovation.

Furthermore, even if there is an existing runtime call these frameworks can use, it often does not offer optimal performance for a user's particular network architecture and dataset, missing optimizations between operators as well as optimizations that can be done knowing the size and shape of data. Our contributions include (1) a language close to the mathematics of deep learning called Tensor Comprehensions, (2) a polyhedral just-in-time compiler to convert a mathematical description of a deep learning DAG into a CUDA kernel with delegated memory management and synchronization, also providing optimizations such as operator fusion and specialization for specific sizes, (3) a compilation cache populated by an autotuner.

3.27 Loop Synthesis for Basic Linear Algebra Computations with Structured Matrices

Daniele G. Spampinato (Carnegie Mellon University – Pittsburgh, US)

Joint work of Markus Püschel

I describe the loop synthesis process in LGen, a research compiler designed for the generation of explicitly vectorized C code for small-scale basic linear algebra computations where input and output matrices may have a structure, such as lower triangular or symmetric. The input computation is expressed mathematically and the structures of the matrices on which it computes are described using a polyhedral notation. These structures, and the semantics of the operations contained in the computation, are used by LGen to produce a SCoP representation of the computation's iteration space. The resulting SCoPs are finally processed by an adapted version of the CLooG generator capable of synthesizing a mathematical formulation of the input computation where loops are expressed in terms of summations.

3.28 A Systematic Approach to High-Performance Generalized Matrix Multiplication Kernels

Richard Veras (Louisiana State University – Baton Rouge, US)

Joint work of Richard Veras, Tyler M. Smith, Tze Meng Low, Franz Franchetti, Robert van de Geijn

A large body of problems arising from linear algebra and big data can be represented by a generalization of the matrix-matrix multiplication operation. These domains are performance-critical and obtaining the level of performance seen in traditional dense matrix-matrix multiplication is a sought-after goal by the community. This is difficult because current high-performance matrix multiplication kernels are tuned for their target architecture and are written by hand in assembly code by expert programmers. Unfortunately, this approach is not sustainable for producing the large span of kernels that we are interest in. Therefore, our solution involves capturing the expert's knowledge and automating the application of this knowledge in the form of kernel code generator. The result is generalized matrix-matrix multiplication kernels that perform as well as an expert implementation.

3.29 Reasoning about Program Properties using Polyhedral Analysis

Sriram Krishnamoorthy (Pacific Northwest National Laboratory – Richland, US)

Joint work of Sriram Krishnamoorthy, Wenlei Bao, Sanket Tavarageri, Louis-Noël Pouchet, Fabrice Rastello, P. Sadayappan

Similarly to other program analysis techniques, polyhedral analysis can be used to reason about and check program properties. I will present a few use cases: detecting soft memory errors, checking program transformations, and modeling caching behavior. In addition, I will argue the need for robust and optimized tool chains to enable broader use of loop optimization techniques.

3.30 Using #pragmas to Direct Polly Transformations

Michael Kruse (ENS – Paris, FR)

Polly today decides automatically which loop transformations it applies, using isl's rescheduling algorithm. This might not always be the optimal transformation, so users may want to decide themselves which transformations lead to the most performant code. We are proposing to use #pragmas in the source code which enforce a specific transformation. These pragmas can either be inserted directly by the programmer, or, inserted by an autotuner framework which explores the search space of promising optimizations.

3.31 Polyhedral Optimizations toward Performance Portability

Jun Shirako (Georgia Institute of Technology – Atlanta, US)

Performance portability, the ability to enable sufficient performance across multiple hardware platforms, is getting more important in the era of extreme-scale heterogeneous computing. Compiler optimizations can play a key role in achieving this goal – i.e., enabling users to write simple and platform-independent programs and compilers to handle performance-oriented optimizations and customizations for the target system. We introduce a series of attempts to address this challenge based on the polyhedral model: (1) integration of polyhedral and AST-based transformations; (2) optimizations of explicitly parallel programs; (3) two-level parallelization for GPU accelerators; and (4) integration of data-layout and loop transformations.

Participants

- Cédric Bastoul
University of Strasbourg, FR
- Barbara M. Chapman
Stony Brook University, US
- Shigeru Chiba
University of Tokyo, JP
- Charisee Chiw
University of Chicago, US
- Philippe Clauss
University of Strasbourg, FR
- Albert Cohen
ENS – Paris, FR
- James W. Demmel
University of California – Berkeley, US
- Johannes Doerfert
Universität des Saarlandes, DE
- Andi Drebes
University of Manchester, GB
- Paul Feautrier
ENS – Paris, FR
- Stefan Ganser
Universität Passau, DE
- Armin Größlinger
Universität Passau, DE
- Tobias Grosser
ETH Zürich, CH
- Sebastian Hack
Universität des Saarlandes, DE
- Julian Hammer
Universität Erlangen-Nürnberg, DE
- Frank Hannig
Universität Erlangen-Nürnberg, DE
- Alexandra Jimborean
Uppsala University, SE
- Paul H. J. Kelly
Imperial College London, GB
- Sriram Krishnamoorthy
Pacific Northwest National Lab. – Richland, US
- Michael Kruse
ENS – Paris, FR
- Roland Leißa
Universität des Saarlandes, DE
- Christian Lengauer
Universität Passau, DE
- Fabio Luporini
Imperial College London, GB
- Benoit Meister
Reservoir Labs, Inc. – New York, US
- Lawrence Mitchell
Imperial College London, GB
- Madan Musuvathi
Microsoft Research – Redmond, US
- Victor Nicolet
University of Toronto, CA
- Philip Pfaffe
KIT – Karlsruher Institut für Technologie, DE
- Antoniu Pop
University of Manchester, GB
- Louis-Noël Pouchet
Colorado State University – Fort Collins, US
- Jonathan Ragan-Kelley
University of California – Berkeley, US
- P. (Saday) Sadayappan
Ohio State University – Columbus, US
- Jun Shirako
Georgia Institute of Technology – Atlanta, US
- Andreas Simbürger
Universität Passau, DE
- Daniele G. Spampinato
Carnegie Mellon University – Pittsburgh, US
- Michel Steuwer
University of Glasgow, GB
- Tianjiao Sun
Imperial College London, GB
- Nicolas Vasilache
Facebook – New York, US
- Richard M. Veras
Louisiana State Univ. – Baton Rouge, US
- Sven Verdoolaege
Facebook – Paris, FR
- Ayal Zaks
Technion – Haifa, IL

Report from Dagstuhl Seminar 18112

Coding Theory for Inference, Learning and Optimization

Edited by

Po-Ling Loh[1], Arya Mazumdar[2], Dimitris Papailiopoulos[3], and Rüdiger Urbanke[4]

1 **University of Wisconsin – Madison, US,** loh@ece.wisc.edu
2 **University of Massachusetts, US,** arya@cs.umass.edu
3 **University of Wisconsin – Madison, US,** dimitris@papail.io
4 **EPFL – Lausanne, CH,** rudiger.urbanke@epfl.ch

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 18112, "Coding Theory for Inference, Learning and Optimization."

Coding theory has recently found new applications in areas such as distributed machine learning, dimension reduction, and variety of statistical problems involving estimation and inference. In machine learning applications that use large-scale data, it is desirable to communicate the results of distributed computations in an efficient and robust manner. In dimension reduction applications, the pseudorandom properties of algebraic codes may be used to construct projection matrices that are deterministic and facilitate algorithmic efficiency. Finally, relationships that have been forged between coding theory and problems in theoretical computer science, such as k-SAT or the planted clique problem, lead to a new interpretation of the sharp thresholds encountered in these settings in terms of thresholds in channel coding theory.

The aim of this Dagstuhl Seminar was to draw together researchers from industry and academia that are working in coding theory, particularly in these different (and somewhat disparate) application areas of machine learning and inference. The discussions and collaborations facilitated by this seminar were intended to spark new ideas about how coding theory may be used to improve and inform modern techniques for data analytics.

Seminar March 11–16, 2018 – https://www.dagstuhl.de/18112
2012 ACM Subject Classification Mathematics of computing → Coding theory,Mathematics of computing → Probability and statistics, Theory of computation → Mathematical optimization
Keywords and phrases Coding theory, Distributed optimization, Machine learning, Threshold phenomena
Digital Object Identifier 10.4230/DagRep.8.3.60
Edited in cooperation with Po-Ling Loh

1 Executive Summary

Po-Ling Loh (University of Wisconsin – Madison, US)
Arya Mazumdar (University of Massachusetts, US)
Dimitris Papailiopoulos (University of Wisconsin – Madison, US)
Rüdiger Urbanke (EPFL – Lausanne, CH)

Codes are widely used in engineering applications to offer reliability and fault tolerance. The high-level idea of coding is to exploit redundancy in order to create robustness against system noise. The theoretical properties of codes have been studied for decades both from a purely mathematical point of view, as well as in various engineering contexts. The latter have resulted in constructions that have been incorporated into our daily lives: No storage device, cell phone transmission, or Wi-Fi connection would be possible without well-constructed codes.

Recent research has connected concepts in coding theory to non-traditional applications in learning, computation and inference, where codes have been used to design more efficient inference algorithms and build robust, large-scale, distributed computational pipelines. Moreover, ideas derived from Shannon theory and the algebraic properties of random codes have resulted in novel research that sheds light on fundamental phase transition phenomena in several long-standing combinatorial and graph-theoretic problems.

The main goal of our seminar was to accelerate research in the growing field of coding theory for computation and learning, and maximize the transformative role of codes in non-traditional application areas. The seminar brought together 22 researchers from across the world specializing in information theory, machine learning, theoretical computer science, optimization, and statistics. The schedule for each day included a tutorial talk by a senior researcher, followed by shorter talks by participants on recent or ongoing work. The afternoons were devoted to informal breakout sessions for groups to discuss open questions. Two of the larger breakout sessions focused on distributed optimization and group testing.

Seminar participants reported that they enjoyed hearing about new ideas, as well as delving into deeper technical discussions about open problems in coding theory. Some topics deserving special mention include the use of techniques in statistical mechanics; locally decodable and recoverable codes; submodular function optimization; hypergraph clustering; private information retrieval; and contagion on graphs. All participants valued the ample time for discussions between and after talks, as it provided a fruitful atmosphere for collaborating on new topics.

2 Table of Contents

3 Overview of Talks

3.1 Error-correcting codes as samplers

Dimitris Achlioptas (University of California – Santa Cruz, US)

We consider the task of summing a non-negative function f over a discrete set Ω; e.g., to compute the partition function of a graphical model. Ermon et al. have shown that in a probabilistic, approximate sense, summation can be reduced to maximizing f over random subsets of Ω defined by parity (XOR) constraints. Unfortunately, XORs with many variables are computationally intractable, while XORs with few variables have poor statistical performance. We introduce two ideas to address this tradeoff, both motivated by the theory of error-correcting codes. The first is to maximize f over explicitly generated random affine subspaces of Ω, which is equivalent to unconstrained maximization of f over an exponentially smaller domain. The second idea, closer in spirit to the original approach, is to use systems of linear equations defining Low Density Parity Check (LDPC) codes. Even though the equations in such systems only contain $O(1)$ variables each, their sets of solutions (codewords) have excellent statistical properties. By combining these ideas, we achieve dramatic speedup over the original approach and levels of accuracy that were previously unattainable.

3.2 Different facets of the repair problem

Alexander Barg (University of Maryland – College Park, US)

Joint work of Min Ye, Itzhak Tamo and Alexander Barg

The repair problem refers to correcting one or several erasures with a given error-correcting code using as little inter-nodal communication as possible. An information-theoretic lower bound on the repair bandwidth is known from the literature, and codes that meet it with equality are said to support optimal repair. In this talk, we consider several versions of the repair problem, illustrating different techniques behind the construction of optimal codes. We begin by describing an approach to optimal-repair codes using the notion of interference alignment, and presenting several families of codes with a number of additional properties (universality, error tolerance, optimal updates). We then discuss optimal repair of Reed-Solomon codes, presenting codes from this family that support optimal repair of a single failed node or of several failed nodes. In the final part of the talk, we discuss one more setting of the problem, where the task is to perform repair of several nodes in a distributed way, and again construct a family of codes with optimal repair bandwidth.

3.3 Twisted Reed-Solomon codes: Novel class of MDS codes

Martin Bossert (Universität Ulm, DE)

Joint work of Sven Puchinger, Johan Roseskilde, Peter Beelen, and Martin Bossert

We introduce the class of twisted RS codes and prove the MDS property. We derive bounds for the length of the codes, and show that many twisted RS codes exist which are not equivalent to RS codes (or Roth-Lempel codes). We study the application of twisted RS codes in the McEliece cryptosystem and show that the attacks for RS codes do not work in case of twisted RS codes. Furthermore, we show examples of the number of existing twisted RS codes that are not equivalent to RS codes for several parameters.

3.4 Can we access a database both locally and privately?

Elette Boyle (The Interdisciplinary Center – Herzliya, IL)

A private information retrieval (PIR) protocol allows a client to retrieve an item from a remote database while hiding which item is retrieved even from the servers storing the database. The main focus of the large body of work on PIR has been on minimizing the communication complexity. We present an exploratory approach for achieving PIR with sublinear computational complexity, based on a new primitive of Oblivious Locally Decodable Codes.

3.5 Random linear equations

Amin Coja-Oghlan (Goethe-Universität – Frankfurt am Main, DE)

Joint work of Peter Ayre, Pu Gao, Noela Müller, and Amin Coja-Oghlan

Let A be a random $m \times n$ matrix over the finite field F_q with precisely k non-zero entries per row, and let $y \in F_q^m$ be a random vector chosen independently of A. We identify the threshold m/n up to which the linear system $Ax = y$ has a solution with high probability, and analyze the geometry of the set of solutions. In the special case $q = 2$, known as the random k-XORSAT problem, the threshold was determined by [Dubois and Mandler, 2002; Dietzfelbinger et al., 2010; Pittel and Sorkin, 2016], and the proof technique was subsequently extended to the cases $q = 3, 4$ [Falke and Goerdt, 2012]. But the argument depends on technically demanding second moment calculations that do not generalize to $q > 3$. Here, we approach the problem from the viewpoint of a decoding task, which leads to a transparent combinatorial proof.

3.6 A lower bound for maximally recoverable codes with locality

Venkatesan Guruswami (Carnegie Mellon University – Pittsburgh, US)

Joint work of Sivakanth Gopi, Sergey Yekhanin, and Venkatesan Guruswami

MDS codes like Reed-Solomon codes enable erasure correction with optimal redundancy: with h redundant symbols, they allow recovery of any subset of h erased positions. In matrix terms, they are defined by an $h \times n$ parity check matrix, all of whose $h \times h$ submatrices are nonsingular. Such matrices, e.g., Vandermonde, exist over a field of size $O(n)$.

The prevalent use of erasure coding in today's large distributed storage systems, where individual storage nodes often fail, brings to the fore a new requirement: the ability to quickly recover any single symbol of the codeword based on few other codeword symbols. Such "locality" can be built into the code via local parity checks—for example, the n codeword symbols can be partitioned into n/r groups, each with r symbols, obeying a local parity check. Here, r is the locality parameter of the code. In matrix terms, we have an $(n/r + h) \times n$ matrix with the first n/r rows, each with r 1's, corresponding to the local parity checks, and the last h rows unrestricted. The local checks compromise the MDS property, but we would still like the code to correct all erasure patterns that can possibly be recovered given the specified topology of parity checks. This property is called Maximal Recoverability, and for the above topology, amounts to the nonsingularity of every $(n/r + h) \times (n/r + h)$ submatrix that includes at least one column from each local group.

The known constructions (and even existence proofs) of such matrices require a very large field size of about n^h, and it has been an important question whether MR codes can exist over smaller, even $O(n)$-sized, fields. The talk will mention the prior construction with n^h field size, and then present a recent super-linear lower bound on the field size which relies on known vertex expansion properties of the hyperplane-point incidence graph in projective space.

3.7 Shifted weight distributions

Anna Gál (University of Texas – Austin, US)

We discuss an open problem of a coding-theoretic flavor. This question came up as part of an approach towards solving an open problem in computational complexity theory.

3.8 Submodular maximization: The decentralized setting

Hamed S. Hassani (University of Pennsylvania – Philadelphia, US)

In this talk, we showcase the interplay between discrete and continuous optimization in network-structured settings. We propose the first fully-decentralized optimization method for a wide class of non-convex objective functions that possess a diminishing returns property. More specifically, given an arbitrary connected network and a global continuous submodular function, formed by a sum of local functions, we develop Decentralized Continuous Greedy (DCG), a message-passing algorithm that converges to the tight $(1-1/e)$-approximation factor of the optimum global solution using only local computation and communication. We also provide strong convergence bounds as a function of network size and spectral characteristics of the underlying topology. Interestingly, DCG readily provides a simple recipe for decentralized discrete submodular maximization through the means of continuous relaxations. Formally, we demonstrate that by lifting the local discrete functions to continuous domains and using DCG as an interface, we can develop a consensus algorithm that also achieves the tight $(1-1/e)$-approximation guarantee of the global discrete solution, once a proper rounding scheme is applied.

3.9 Sufficiently myopic adversaries are weak

Sidharth Jaggi (The Chinese University of Hong Kong, HK)

In this work, we consider a communication problem in which a sender, Alice, wishes to communicate with a receiver, Bob, over a channel controlled by an adversarial jammer, James, who is *myopic*. Roughly speaking, for blocklength n, the codeword X^n transmitted by Alice is corrupted by James, who must base his adversarial decisions (of which locations of X^n to corrupt and how to corrupt them) not on the codeword X^n, but on Z^n, an image of X^n through a noisy memoryless channel. More specifically, our communication model may be described by two channels: A memoryless channel $p(z|x)$ from Alice to James, and an *Arbitrarily Varying Channel* $p(y|x,s)$ from Alice to Bob, governed by a state X^n determined by James. In standard adversarial channels, the states S^n may depend on the codeword X^n, but in our setting, S^n depends only on James's view Z^n.

The myopic channel captures a broad range of channels and bridges between the standard models of memoryless and adversarial (zero-error) channels. In this work, we present upper and lower bounds on the capacity of myopic channels. For a number of special cases of interest, we show that our bounds are tight. We extend our results to the setting of *secure* communication, in which we require that the transmitted message remain secret from James. For example, we show that if (i) James may flip at most a p fraction of the bits communicated between Alice and Bob, and (ii) James views X^n through a binary symmetric channel with parameter q, then once James is "sufficiently myopic" (in this case, when $q > p$), the optimal communication rate is that of an adversary who is "blind" (that is, an adversary that does not see X^n at all), which is $1 - H(p)$ for standard communication, and $H(q) - H(p)$ for secure communication. A similar phenomenon exists for our general model of communication.

3.10 Fundamental limits of symmetric low-rank matrix estimation

Marc Lelarge (ENS – Paris, FR)

Joint work of Leo Miolane and Marc Lelarge

We consider the high-dimensional inference problem, where the signal is a low-rank symmetric matrix which is corrupted by additive Gaussian noise. Given a probabilistic model for the low-rank matrix, we compute the limit in the large-dimension setting for the mutual information between the signal and the observations, as well as the matrix minimum mean square error, while the rank of the signal remains constant. We also show that our model extends beyond the particular case of additive Gaussian noise, and we prove an universality result connecting the community detection problem to our Gaussian framework. We unify and generalize a number of recent works on PCA, sparse PCA, submatrix localization, and community detection, by computing the information-theoretic limits for these problems in the high-noise regime. In addition, we show that the posterior distribution of the signal given the observations is characterized by a parameter of the same dimension as the square of the rank of the signal (i.e., scalar in the case of rank one). Finally, we connect our work with the hard but detectable conjecture in statistical physics.

3.11 Statistical inference for infectious disease modeling

Po-Ling Loh (University of Wisconsin – Madison, US)

Joint work of Justin Khim and Po-Ling Loh

We discuss two recent results concerning disease modeling on networks. The infection is assumed to spread via contagion (e.g., transmission over the edges of an underlying network). In the first scenario, we observe the infection status of individuals at a particular time instance and the goal is to identify a confidence set of nodes that contain the source of the infection with high probability. We show that when the underlying graph is a tree with certain regularity properties and the structure of the graph is known, confidence sets may be constructed with cardinality independent of the size of the infection set. In the second scenario, the goal is to infer the network structure of the underlying graph based on knowledge of the infected individuals. We develop a hypothesis test based on permutation testing, and describe a sufficient condition for the validity of the hypothesis test based on automorphism groups of the graphs involved in the hypothesis test.

3.12 The adaptive interpolation method for proving replica formulas

Nicolas Macris (EPFL – Lausanne, CH)

In this talk, we give an introduction to the newly-introduced adaptive interpolation method to prove in a simple and unified way replica formulas for Bayesian optimal inference problems.

We illustrate the method with a paradigmatic inference problem, namely rank-one matrix estimation.

The replica method from statistical mechanics has been applied to Bayesian inference problems (e.g., coding, estimation) already two decades ago. Rigorous proofs of the formulas for the mutual informations/entropies/free energies stemming from this method have for a long time only been partial, consisting generally of one-sided bounds. It is only quite recently that there has been a surge of progress using various methods (e.g., spatial coupling, the Aizenman-Sims-Starr principle, and the cavity method) to derive full proofs, but which are typically quite complicated. Recently with Jean Barbier, we introduced a powerful evolution of the Guerra-Toninelli interpolation method—called adaptive interpolation—that allows to fully prove the replica formulas in a quite simple and unified way for Bayesian inference problems (we note that in its original, more complicated incarnation, we called this method "stochastic interpolation").

We review the method for the rank-one matrix estimation or factorisation problem (one of the simplest non-linear estimation problems). The main new ingredient is the concentration of the "overlap" which, remarkably, can be proven for Bayesian inference problems when the prior and hyperparameters are all known. We will refer to this setting where the prior and hyperparameters are known as the Bayesian optimal inference.

The adaptive interpolation method has been fruitfully applied to a range of more difficult problems with a dense underlying graphical structure. So far, these include matrix and tensor factorisation, estimation of traditional and generalised linear models, and learning (e.g., compressed sensing and the single-layer perceptron network). For inference problems with an underlying sparse graphical structure, full proofs of replica formulas are scarce and much more involved. The adaptive interpolation method is still in its infancy for sparse systems, and it would be desirable to develop it further.

3.13 Interactive learning for clustering and community detection

Arya Mazumdar (University of Massachusetts, US)

Clustering has always been a central problem of multiple disciplines within computer science, optimization, and statistics. In the model of interactive clustering, a clustering algorithm can adaptively query a (possibly noisy) oracle, with a small number of elements from the set that is to be clustered. For example, the oracle may answer pairwise queries of the form, "do two elements u and v belong to the same cluster?" This model fits exactly to the recently popular experimental setups where crowdsourcing is used to improve clustering accuracy for various data mining tasks. The goal is to minimize the number of such queries while obtaining the optimum clustering. One of our main contributions is to show the power of side information in the form of a similarity matrix: a matrix that represents noisy pairwise relationships, such as one computed by some automated function on attributes of the elements. The reduction in query complexity under general models of the similarity matrix is stunning, and this remains true even when the answer of each query can be erroneous with a certain probability and "resampling" is not allowed. Our results include a general framework for proving lower bounds for classification problems in the interactive setting. Our algorithms are computationally efficient and are parameter-free; i.e., it works without any knowledge of the number of clusters or the similarity matrix distribution. The query models we propose have interesting connections to popular community detection models such as the stochastic block model.

3.14 Query and higher-order clustering: Some open problems

Olgica Milenkovic (University of Illinois – Urbana Champaign, US)

We will describe a number of problems in clustering and hypergraph clustering that combine discrete optimization and combinatorial techniques to solve learning problems with side information. In particular, we will discuss the connections between the query k-means clustering problem and generalized constrained coupon collector problems, and submodular hypergraph partitioning.

3.15 Representation learning and signal recovery in nonlinear models

Ankit Singh Rawat (MIT – Cambridge, US)

Nonlinear generative models have become increasingly important in capturing the datasets that appear in various domains and realizing various inference tasks around these datasets. In this talk, we present results towards realizing two basic tasks of representation learning and robust signal recovery in the context of nonlinear generative models. In particular, we discuss the estimation of a low-rank matrix from its nonlinear transformation and recovery of a latent signal from its nonlinear measurements in the presence of outliers. The recovery algorithm is agnostic to underlying nonlinearity and comes with a tight performance analysis.

3.16 Coded gradient computation from cyclic MDS codes and expander graphs

Itzhak Tamo (Tel Aviv University, IL)

We discuss cyclic MDS codes and expander graph techniques for distributed gradient computation in the presence of stragglers. For exact gradient computation, the suggested techniques employ cyclic MDS codes to attain comparable parameters to previously known ones, but in a deterministic fashion. For estimated gradient computation, a novel scheme is suggested, stemming from adjacency matrices of expander graphs.

3.17 Inference, coding, and learning in quantum information processing

Pascal Vontobel (The Chinese University of Hong Kong, HK)

Some of the most interesting quantities associated with a factor graph are its marginals and its partition sum. For factor graphs *without cycles* and moderate message-update complexities, the sum-product algorithm (SPA) can be used to efficiently compute these quantities exactly. Moreover, for various classes of factor graphs *with cycles*, the SPA has been successfully applied to efficiently compute good approximations to these quantities. Note that in the case of factor graphs with cycles, the local functions are usually non-negative real-valued functions.

In this talk, we introduce a class of factor graphs, called double-edge factor graphs (DE-FGs), which allow local functions to be complex-valued and only require them, in some suitable sense, to be positive semi-definite kernel functions. We discuss various properties of the SPA when running it on DE-FGs and we show promising numerical results for various example DE-FGs, some of which have connections to quantum information processing.

3.18 Algorithmic applications of list-recovery

Mary Wootters (Stanford University, US)

List-recoverable codes and algorithms for list-recovery have found many applications in algorithm design. In this talk, we define list-recovery, mention state-of-the-art methods for list-recoverable codes, and give two examples of algorithmic applications in group testing and streaming algorithms. The hope is that list-recovery may be helpful in this workshop, as we think about algorithmic applications in inference, optimization, and learning.

4 Working groups

4.1 Group testing

Arya Mazumdar (University of Massachusetts, US)

The Tuesday afternoon break-out session focused on group testing: the recovery of a sparse binary signal under Boolean measurements. The main objective in nonadaptive group testing is to identify a set of defective elements by 1) creating pools of elements; and 2) testing the pools simultaneously for the existence of any defective elements in the pool.

The discussion was led by Sidharth Jaggi, who presented a nice overview of existing signal recovery algorithms for group testing. Earlier in the day, Mary Wootters introduced the group testing problem in relation to list recovery of error-correcting codes. Sidharth continued this discussion by presenting the simplest possible group testing algorithm, which he called combinatorial orthogonal matching pursuit, or COMP (to acknowledge the well-known orthogonal matching pursuit algorithm from the compressed sensing literature). In the COMP algorithm, all the elements in the pools that produce negative test results are cleared, and remaining items are declared to be defective. Sidharth showed the performance of random test matrices under COMP, and proposed a new heuristic algorithm.

The new algorithm is a generalization of COMP, where the correlations of pairs of columns with the test results are computed (instead of being computed column-wise, as in COMP). This algorithm demonstrates remarkable improvements in empirical simulations; however, the theoretical justification is still far from complete. Sidharth presented several possible ideas to analyze the algorithm, including technical loopholes that would need to be closed.

4.2 Large-scale machine learning meets coding theory

Dimitris Papailiopoulos (University of Wisconsin – Madison, US)

During the Monday afternoon breakout session, we discussed problems related to distributed machine learning and coding theory, focusing on straggler nodes and communication bottlenecks in the context of distributed gradient-based algorithms.

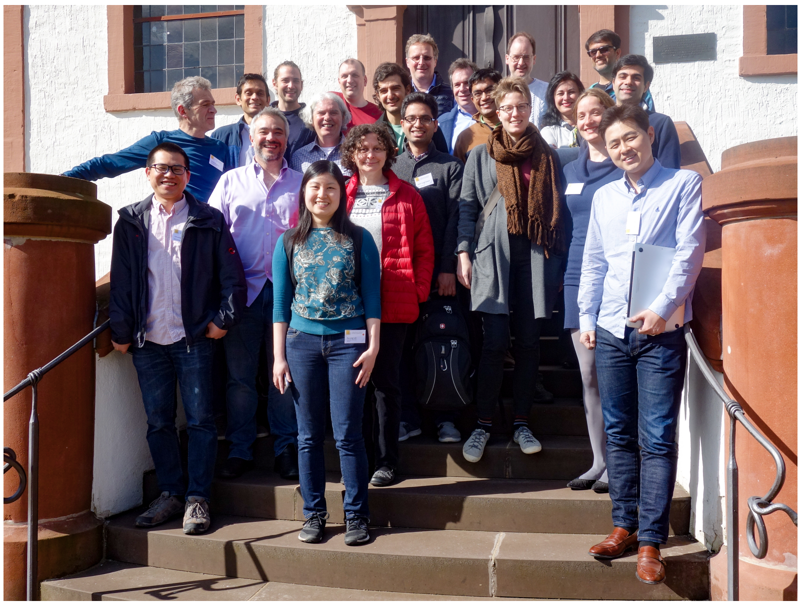
In synchronous distributed setups, the presence of straggler nodes (i.e., nodes slower than the average) can significantly impair the performance of training algorithms. A large body of recent work has focused on reducing the effect of stragglers. Current approaches include replicating jobs across nodes and dropping stragglers in settings where the system can tolerate errors. More recently, coding theory has gained traction as a way to speed up distributed computation. Codes have been used to reduce the runtime of the shuffling phase of MapReduce, improve the efficiency of distributed matrix multiplication, and reduce the effect of stragglers during gradient-based learning. Some interesting problems lying in the intersection between distributed learning and coding theory revolve around statistical accuracy, redundancy, and resilience to stragglers. We discussed several of these problems during our Monday break-out session.

We also discussed communication bottlenecks arising during distributed learning. In gradient-based algorithms, where a master node stores the global model and compute nodes evaluate gradients, frequent communication between nodes is needed to train an accurate model. However, such frequent communication may lead to bottlenecks in the system.

An open problem in the area of communication-efficient distributed training is that of gradient quantization. The challenge of optimally quantizing gradients may be posed as an optimization problem, where the objective is to minimize the variance of stochastic gradients, which controls the rate of convergence, subject to the constraint that gradients are sufficiently simple to represent. The constraints of this optimization problem usually take the following form: 1) the quantized gradient has to be an unbiased estimate of the true gradient on the data; and 2) such quantized gradients need to be represented using few atoms (e.g., bits, elements, or low-rank components). The problem of devising an optimal quantization scheme that minimizes gradient variance, while ensuring an unbiased and compressed representation gradient estimates, remains open. As we discussed in the break-out session, a potential approach to this problem is to establish a connection with traditional element and vector quantization schemes on Gaussian sources.

Participants

- Dimitris Achlioptas
University of California – Santa Cruz, US
- Alexander Barg
University of Maryland – College Park, US
- Martin Bossert
Universität Ulm, DE
- Elette Boyle
The Interdisciplinary Center – Herzliya, IL
- Amin Coja-Oghlan
Goethe-Universität – Frankfurt am Main, DE
- Anna Gál
University of Texas – Austin, US
- Venkatesan Guruswami
Carnegie Mellon University – Pittsburgh, US
- Hamed S. Hassani
University of Pennsylvania – Philadelphia, US
- Sihuang Hu
RWTH Aachen, DE
- Sidharth Jaggi
The Chinese University of Hong Kong, HK
- Marc Lelarge
ENS – Paris, FR
- Po-Ling Loh
University of Wisconsin – Madison, US
- Nicolas Macris
EPFL – Lausanne, CH
- Arya Mazumdar
University of Massachusetts – Amherst, US
- Olgica Milenkovic
University of Illinois – Urbana Champaign, US
- Dimitris Papailiopoulos
University of Wisconsin – Madison, US
- Ankit Singh Rawat
MIT – Cambridge, US
- Changho Suh
KAIST – Daejeon, KR
- Itzhak Tamo
Tel Aviv University, IL
- Rüdiger Urbanke
EPFL – Lausanne, CH
- Pascal Vontobel
The Chinese University of Hong Kong, HK
- Mary Wootters
Stanford University, US



Report from Dagstuhl Seminar 18121

Machine Learning and Model Checking Join Forces

Edited by

Nils Jansen[1], Joost-Pieter Katoen[2], Pushmeet Kohli[3], and Jan Kretinsky[4]

1 **Radboud University Nijmegen, NL,** `n.jansen@science.ru.nl`
2 **RWTH Aachen University, DE,** `katoen@cs.rwth-aachen.de`
3 **Google DeepMind – London, GB,** `pushmeet@google.com`
4 **TU München, DE,** `jan.kretinsky@tum.de`

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 18121 "Machine Learning and Model Checking Join Forces". This Dagstuhl Seminar brought together researchers working in the fields of machine learning and model checking. It helped to identify new research topics on the one hand and to help with current problems on the other hand.

Seminar March 18–23, 2018 – https://www.dagstuhl.de/18121
2012 ACM Subject Classification Theory of computation → Machine learning theory, Hardware → Model checking
Keywords and phrases artificial intelligence, cyber-physical systems, formal methods, formal verification, logics, machine learning, model checking, quantitative verification, safety-critical systems
Digital Object Identifier 10.4230/DagRep.8.3.74
Edited in cooperation with Alexis Linard

1 Executive Summary

Nils Jansen
Joost-Pieter Katoen
Pushmeet Kohli
Jan Kretinsky

This Dagstuhl Seminar aimed at bringing together researchers working in the fields of machine learning and model checking. Growing application areas for machine learning, such as autonomous driving, require the exclusion or likely avoidance of unsafe behaviors. An important question is then, how confidence in system behaviors obtained from machine learning can be transferred to formal verification. Vice versa, industrial usage of model checking still suffers from scalability issues for large applications. Leveraging the capabilities of machine learning to assess large data sets will help to enable the verification for more realistic systems.

Based on the concrete discussions and inputs from all the participants, we identified the following topics as great challenges to the combination of the fields of machine learning and model checking.

- Safety Verification of Deep Neural Networks
- Formal Program Synthesis and Analysis using Machine Learning
- Representation of Strategies and Controllers
- Explainable Artificial Intelligence
- Challenges for Machine Learning in Motion Planning
- Guarantees on Reinforcement Learning in Verification
- Social and Legal Issues in Artificial Intelligence
- Exploiting Weaknesses in Reinforcement Learning

2 Table of Contents

3 Overview of Talks

3.1 Formal verification of complex systems: model-based and data-driven methods

Alessandro Abate (University of Oxford, GB)

Two known shortcomings of standard techniques in formal verification are the limited capability to provide system-level assertions, and the scalability to large-scale, complex models, such as those needed in Cyber-Physical Systems (CPS) applications. Using data, which nowadays is becoming ever more accessible, has the potential to mitigate such limitations. However, this notoriously leads to a lack of formal proofs that are needed in safety-critical systems.

This talk covers research which addresses these shortcomings, by bringing model-based and data-driven methods together, which can help pushing the envelope of existing algorithms and tools in formal verification.

In the first part of the talk, I will discuss a new, formal, measurement-driven and model-based automated technique, for the quantitative verification of systems with partly unknown dynamics. I will formulate this setup as a data-driven Bayesian inference problem, formally embedded within a quantitative, model-based verification procedure. I argue that the approach can be applied to complex physical systems (e.g., with spatially continuous variables), driven by external inputs and accessed under noisy measurements.

In the later part of the talk, I will concentrate on systems represented by models that are probabilistic with heterogeneous dynamics (continuous/discrete, i.e. hybrid, as well as nonlinear). Such stochastic hybrid models (SHS) are a natural mathematical framework for CPS. With focus on model-based verification procedures, I will provide algorithms for quantitative model checking of temporal specifications on SHS with formal guarantees. This is attained via the development of formal abstraction techniques based on quantitative approximations.

Theory is complemented by algorithms, all packaged in a software tool that is available to users, and applied in the domain of Smart Energy.

References

1 E. Polgreen, V. B. Wijesuriya, S. Haesaert and A. Abate, "Automated Experiment Design for Efficient Verification of Parametric Markov Decision Processes," QEST17, LNCS 10503, pp. 259–274, 2017.
2 E. Polgreen, V. B. Wijesuriya, S. Haesert and A. Abate, "Data-efficient Bayesian verification of parametric Markov chains," QEST16, LNCS 9826, B. Van Houdt and G. Agha (Eds.), pp. 35–51, 2016.
3 S. Haesaert, P. M. J. V. d. Hof, and A. Abate, "Data-driven and Model-based Verification via Bayesian Identification and Reachability Analysis," Automatica, vol. 79, pp. 115–126, May 2017.

3.2 Shield Synthesis

Roderick Bloem (TU Graz, AT)

Shield synthesis is an approach to enforce safety properties at runtime. A shield monitors the system and corrects any erroneous output values instantaneously. The shield deviates from the given outputs as little as it can and recovers to hand back control to the system as soon as possible. In the first part of this paper, we consider shield synthesis for reactive hardware systems. First, we define a general framework for solving the shield synthesis problem. Second, we discuss two concrete shield synthesis methods that automatically construct shields from a set of safety properties: (1) k-stabilizing shields, which guarantee recovery in a finite time. (2) Admissible shields, which attempt to work with the system to recover as soon as possible. Next, we discuss an extension of k-stabilizing and admissible shields, where erroneous output values of the reactive system are corrected while liveness properties of the system are preserved. Finally, we give experimental results for both synthesis methods. In the second part of the paper, we consider shielding a human operator instead of shielding a reactive system: the outputs to be corrected are not initiated by a system but by a human operator who works with an autonomous system. The challenge here lies in giving simple and intuitive explanations to the human for any interferences of the shield. We present results involving mission planning for unmanned aerial vehicles.

3.3 Statistical Parameter Verification of Stochastic Models

Luca Bortolussi (University of Trieste, IT)

Joint work of Luca Bortolussi, Guido Sanguinetti et al.
Main reference Luca Bortolussi, Dimitrios Milios, Guido Sanguinetti: “Smoothed model checking for uncertain Continuous-Time Markov Chains”, Inf. Comput., Vol. 247, pp. 235–253, 2016.
URL http://dx.doi.org/10.1016/j.ic.2016.01.004

Parametric verification and parameter syntesis are fundamental tools to apply formal methods to the design of Cyber-Physical and complex systems. The biggest challenge in this area is scalability to realistic stochastic models of those systems. Recently, a parametric verification has been tackled by a statistical approach grounded in Bayesian Machine Learning techniques, namely Gaussian Processes. The method, called smoothed Model Checking [1], tackles parametric verification of linear time properties of black box statistical models, as a function of model or property parameters, under mild conditions on continuity on parameters of the satisfaction probability. It requires simulation data – substantially the truth value of the property of interest at a small number of parameters points of the parameter space, and only few simulations per point. Being Bayesian, it provides not only an estimate of the satisfaction probability, but also uncertainty estimates at each point. This approach has been leveraged to efficiently solve several tasks, like parameter synthesis [2], system design [4], counterexample generation [6], requirement synthesis [5], parameter estimation from Boolean observations [3], combining it with active learning ideas.

References

1 L. Bortolussi, D. Milios, and G. Sanguinetti, "Smoothed model checking for uncertain Continuous-Time Markov Chains", Information and Computation, vol. 247, pp. 235–253, Apr. 2016.

2 L. Bortolussi and S. Silvetti, "Bayesian Statistical Parameter Synthesis for Linear Temporal Properties of Stochastic Models", in Tools and Algorithms for the Construction and Analysis of Systems, vol. 10806, D. Beyer and M. Huisman, Eds. Cham: Springer International Publishing, 2018, pp. 396–413

3 L. Bortolussi and G. Sanguinetti, "Learning and Designing Stochastic Processes from Logical Constraints", Logical Methods in Computer Science, vol. 11, no. 2, 2015.

4 E. Bartocci, L. Bortolussi, L. Nenzi, and G. Sanguinetti, "System design of stochastic models using robustness of temporal properties", Theoretical Computer Science, vol. 587, pp. 3–25, Jul. 2015.

5 E. Bartocci, L. Bortolussi, and G. Sanguinetti, "Data-driven statistical learning of temporal logic properties", in Formal Modeling and Analysis of Timed Systems, Springer, 2014, pp. 23–37

6 S. Silvetti, A. Policriti, L. Bortolussi, "An Active Learning Approach to the Falsification of Black Box Cyber-Physical Systems". IFM 2017: 3–17

3.4 Learning to Represent Programs with Graphs

Marc Brockschmidt (Microsoft Research UK – Cambridge, GB)

Joint work of Marc Brockschmidt, Mahmoud Khademi, Miltos Allamanis
Main reference Miltiadis Allamanis, Marc Brockschmidt, Mahmoud Khademi: "Learning to Represent Programs with Graphs", CoRR, Vol. abs/1711.00740, 2017.
URL http://arxiv.org/abs/1711.00740

Learning tasks on source code (i.e. , formal languages) have been considered re- cently, but most work has tried to transfer natural language methods and does not capitalize on the unique opportunities offered by code's known sematics. For example, long-range dependencies induced by using the same variable or function in distant locations are often not considered. We propose to use graphs to represent both the syntactic and semantic structure of code and use graph-based deep learning methods to learn to reason over program structures. In this work, we present how to construct graphs from source code and how to scale Gated Graph Neural Networks training to such large graphs. We evaluate our method on two tasks: VarNaming , in which a network attempts to predict the name of a variable given its usage, and VarMisuse, in which the network learns to reason about selecting the correct variable that should be used at a given program location. Our comparison to methods that use less structured program representations shows the advantages of modeling known structure, and suggests that our models learn to infer meaningful names and to solve the VarMisuse task in many cases. Additionally, our testing showed that VarMisuse identifies a number of bugs in mature open-source projects.

3.5 A Unified View of Piecewise Linear Neural Network Verification

Rudy Bunel (University of Oxford, GB) and Pushmeet Kohli (Google DeepMind – London, GB)

Joint work of Rudy Bunel, Ilker Turkaslan, Philip H. S. Torr, Pushmeet Kohli, M. Pawan Kumar
Main reference Rudy Bunel, Ilker Turkaslan, Philip H. S. Torr, Pushmeet Kohli, M. Pawan Kumar: "Piecewise Linear Neural Network verification: A comparative study", CoRR, Vol. abs/1711.00455, 2017.
URL http://arxiv.org/abs/1711.00455

The success of Deep Learning and its potential use in many safety-critical applications has motivated research on formal verification of Neural Network (NN) models. Despite the reputation of learned NN models to behave as black boxes and the theoretical hardness of proving their properties, researchers have been successful in verifying some classes of models by exploiting their piecewise linear structure. To facilitate progress on this crucial area, we make two key contributions. First, we present a unified framework that encompasses previous methods. This analysis results in the identification of new methods that combine the strengths of multiple existing approaches. Second, we propose a new data set of benchmarks which includes a collection of previously released testcases. We use the benchmark to provide the first experimental comparison of the algorithms.

3.6 Managing and Exploiting Uncertainty for Fast Approximate Computations

Michael Carbin (MIT – Cambridge, US)

Joint work of Michael Carbin, Brett Boston, Zoe Gong

Many modern applications implement large-scale computations (e.g., machine learning, big data analytics, and financial analysis) in which there is a natural trade-off between the quality of the results that the computation produces and the performance and cost of executing the computation.

Exploiting this fact, researchers have recently developed a variety of new mechanisms that automatically change the structure and execution of an application to enable it to meet its performance requirements. Examples of these mechanisms include skipping portions of the application's computation and executing the application on fast and/or energy-efficient unreliable hardware systems whose operations may silently produce incorrect results.

In this talk, I survey a variety of these new mechanisms as well as present how program verification and analysis makes it possible to verify the safety, security, and accuracy of the approximate applications that these mechanisms produce.

References

1 Leto: Verifying Programs Under Custom Application-Specific Execution Models Brett Boston, Zoe Gong, and Michael Carbin https://arxiv.org/pdf/1805.06090.pdf
2 Verifying Quantitative Reliability for Programs that Execute on Unreliable Hardware Michael Carbin, Sasa Misailovic, and Martin C. Rinard. OOSPLA 2013 https://dl.acm.org/citation.cfm?id=2509546

3 Chisel: Reliability- and Accuracy-Aware Optimization of Approximate Computational Kernels Sasa Misailovic, Michael Carbin, Sara Achour, Zichao Qi, and Martin C. Rinard. OOPSLA 2014 https://dl.acm.org/citation.cfm?id=2660231
4 Proving Acceptability Properties of Relaxed Nondeterministic Approximate Programs Michael Carbin, Deokhwan Kim, Sasa Misailovic, and Martin C. Rinard. PLDI 2012 https://dl.acm.org/citation.cfm?id=2254086

3.7 Towards Correct-by-Construction Probabilistic Inference

Michael Carbin (MIT – Cambridge, US)

Researchers have recently proposed several systems that ease the process of performing Bayesian probabilistic inference. These include systems for automatic inference algorithm synthesis as well as stronger abstractions for manual algorithm development. However, existing systems whose performance relies on the developer manually constructing a part of the inference algorithm have limited support for reasoning about the correctness of the resulting algorithm.

In this talk, I'll present Shuffle, a programming language for manually developing inference procedures that 1) enforces the basic rules of probability theory, 2) enforces the statistical dependencies of the algorithm's corresponding probabilistic model, and 3) generates an optimized implementation. We have used Shuffle to develop inference algorithms for several standard probabilistic models. Our results demonstrate that Shuffle enables a developer to deliver correct and performant implementations of these algorithms.

References

1 Verifying Handcoded Probabilistic Inference Procedures Eric Atkinson, Cambridge Yang, and Michael Carbin https://arxiv.org/abs/1805.01863

3.8 A dual approach to scalable verification of neural networks

Krishnamurthy Dvijotham (Google UK, GB)

We present a novel approach to verifying input-output properties of neural networks. Our approach relies on dualizing an adversarial optimization problem that seeks to find the maximum violation of the property being verified. The dual problem provides an upper bound on the maximum violation, which, if smaller than zero, acts as a certificate of the property being true. We show that this approach can handle networks with arbitrary feedforward architectures and activation functions. Numerical experiments show that our approach can compute tight upper bounds on the maximum error rate of a neural network classifier under bounded adversarial perturbations in the infinity norm and also handle more complex specifications in a computationally tractable fashion.

3.9 Machine Learning and Formal Methods for Assessing Slope Stability

Rüdiger Ehlers (Universität Bremen, DE)

Joint work of Timo Hartmann, Cormac Reale, and other participants of the EU/H2020 project SAFE-10-T

This talk provided a summary of the *slope stability estimation problem* (dealt with in the EU/H2020 project *SAFE-10-T*) from the computer science perspective. As such estimations are safety-critical, solving the problem not only asks for utilizing the capabilities of modern machine learning approaches to infer models from data, but also for the correctness guarantees that are commonly given by techniques from the area of formal methods. The focus of the talk was on presenting the problem and what properties of the learned models need to be verified. The results of a naive application of neural network learning show that learned models do not automatically have the requested properties, and smarter approaches to combining machine learning and formal verification are likely to be useful for solving the problem.

3.10 Explainable RNNs: Modeling, Learning and Verification

Radu Grosu (TU Wien, AT)

Recurrent neural networks have recently achieved considerable success in matching and in many cases surpassing state-of-the-art robotic controllers. However, they have important deficiencies, that make them inappropriate for safety-critical applications: interpretability, size, and robustness to adversarial attacks. In this talk we present a biophysical alternative that does not suffer from such deficiencies.

3.11 Government & Industry Perspectives, Cultural Challenges, & Applications for Model Checking & Machine Learning

Laura Humphrey (AFRL – Wright Patterson, US)

Government & industry are heavily focused on the development of autonomous systems. However, verification & validation of autonomous systems remains a challenge because the space of behaviors autonomous systems can exhibit is orders of magnitude larger than current systems, and they are expected to be able to modify their behavior in response to new situations through approaches like machine learning. Current research in formal methods is focused on how to adapt approaches such as model checking to handle complex systems that incorporate machine learning. However, even if this can be done, many in government & industry do not have a background in formal methods or even discrete mathematics, leading to cultural challenges in the adoption of formal methods. This talk aims to provide an overview of government & industry perspectives and cultural challenges with respect to verification & validation of autonomous systems. It also presents some potential application problems involving cooperative control of unmanned aerial vehicles, successes in which would help provide concrete evidence to government & industry that model checking and machine learning can be used for design and verification of autonomous systems.

References

1 D. Ahner and C. Parson. Workshop report: Test and evaluation of autonomous systems. Technical report, STAT T&E Center of Excellence, 2016.

2 M. Clark. Autonomy Community of Interest (COI) Test and Evaluation, Verification and Validation (TEVV) working group: Technology investment strategy 2015 – 2018. Technical report, Office of the Assistant Secretary of Defense For Research & Engineering, 2015.

3.12 Motion Planning under Uncertainty and Partial Observability

Nils Jansen (Radboud University Nijmegen, NL)

Joint work of Nils Jansen, Joost-Pieter Katoen, Ufuk Topcu, Sebastian Junges, Ralf Wimmer, Bernd Becker, Leonore Winterer, Christian Dehnert, Steve Carr, Jie Fu

The subject of this talk are motion planning problems where agents move inside environments that are subject to uncertainties and potentially not fully observable. The goal is to compute a strategy or a set of strategies for an agent that is guaranteed to satisfy certain safety or performance specifications. Such problems are naturally modeled by Markov decision processes (MDPs) or partially observable MDPs (POMDPs). We discuss several technical approaches, ranging from the computation of permissive strategies that guarantee safe reinforcement learning in unknown environments, a game-based abstraction framework for POMDPs, as well as the utilization of parameter synthesis for Markov chains to compute randomized strategies for POMDPs. We also consider preliminary work on actively including humans into verification and synthesis processes, and what challenges arise.

References

1 Sebastian Junges, Nils Jansen, Ralf Wimmer, Tim Quatmann, Leonore Winterer, Joost-Pieter Katoen, and Bernd Becker. Finite-state Controllers of POMDPs via Parameter Synthesis. In *UAI*, 2018. to appear.

2 Leonore Winterer, Sebastian Junges, Ralf Wimmer, Nils Jansen, Ufuk Topcu, Joost-Pieter Katoen, and Bernd Becker. Motion planning under Partial observability using Game-based Abstraction. In *CDC*, pages 2201–2208. IEEE, 2017.

3 Steven Carr, Nils Jansen, Ralf Wimmer, Jie Fu, and Ufuk Topcu. Human-in-the-loop synthesis for partially observable markov decision processes. In *ACC*, 2018. to appear.

4 Sebastian Junges, Nils Jansen, Christian Dehnert, Ufuk Topcu, and Joost-Pieter Katoen. Safety-constrained reinforcement learning for MDPs. In *TACAS*, volume 9636 of *LNCS*, pages 130–146. Springer, 2016.

5 Shashank Pathak, Erika Ábrahám, Nils Jansen, Armando Tacchella, and Joost-Pieter Katoen. A greedy approach for the efficient repair of stochastic models. In *NFM*, volume 9058 of *LNCS*, pages 295–309. Springer, 2015.

6 Murat Cubuktepe, Nils Jansen, Sebastian Junges, Joost-Pieter Katoen, and Ufuk Topcu. Synthesis in pMDPs: A tale of 1001 parameters. *CoRR*, abs/1803.02884, 2018.

3.13 Bayes meets Dijkstra Exact Inference by Program Verification

Joost-Pieter Katoen (RWTH Aachen University, DE)

Joint work of Kevin Batz, Benjamin Kaminski and Christoph Matheaa
Main reference Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja: "How long, O Bayesian network, will I sample thee? – A program analysis perspective on expected sampling times", in Proc. of the Programming Languages and Systems – 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, Lecture Notes in Computer Science, Vol. 10801, pp. 186–213, Springer, 2018.
URL http://dx.doi.org/10.1007/978-3-319-89884-1_7

In this talk, I will give a perspective on inference in Bayes' networks (BNs) using program verification. I will argue how weakest precondition reasoning a la Dijkstra can be used for exact inference (and more). As exact inference is NP-complete, inference is typically done by means of simulation. I will show how by means of wp-reasoning exact expected sampling times of BNs can be obtained in a fully automated fashion. An experimental evaluation on BN benchmarks demonstrates that very large expected sampling times (in the magnitude of millions of years) can be inferred within less than a second. This provides a means to decide whether sampling-based methods are appropriate for a given BN. The key ingredients are to reason at program code in a compositional manner.

3.14 Towards Robust and Explainable Artificial Intelligence

Pushmeet Kohli (Google DeepMind – London, GB)

Deep learning has led to rapid progress being made in the field of machine learning and artificial intelligence, leading to dramatically improved solutions of many challenging problems such as image understanding, speech recognition, and automatic game playing. Despite these remarkable successes, researchers have observed some intriguing and troubling aspects of the behaviour of these models. A case in point is the presence of adversarial examples which make learning based systems fail in unexpected ways. Such behaviour and the difficultly of interpreting the behaviour of neural networks is a serious hindrance in the deployment of these models for safety-critical applications. In this talk, I will review the challenges in developing models that are robust and explainable and discuss the opportunities for collaboration between the formal methods and machine learning communities.

3.15 Guarantees in model checking and machine learning

Jan Kretinsky (TU München, DE)

We survey various kinds of combining model-checking and machine-learning algorithms, e.g. [1, 2, 3] and the guarantees on each of the two components as well as the result. We discuss the interest in guarantees from perspectives of both communities.

References

1 Tomas Brazdil, Krishnendu Chatterjee, Martin Chmelik, Vojtech Forejt, Jan Kretinsky, Marta Z. Kwiatkowska, David Parker, Mateusz Ujma: *Verification of Markov Decision Processes Using Learning Algorithms*. ATVA 2014.

2 Tomas Brazdil, Krishnendu Chatterjee, Martin Chmelik, Andreas Fellner, Jan Kretinsky: *Counterexample Explanation by Learning Small Strategies in Markov Decision Processes*. CAV 2015.

3 Tomas Brazdil, Krishnendu Chatterjee, Jan Kretinsky, Viktor Toman: *Strategy Representation by Decision Trees in Reactive Synthesis*. TACAS 2018.

3.16 Verification, Analysis, Synthesis Optimization using UPPAAL Stratego

Kim Guldstrand Larsen (Aalborg University, DK)

I will present the framework of stochastic Timed Hybrid Automata and Games and show how the tools UPPAAL, UPPAAL SMC and UPPAAL Stratego allows to perform model checking providing in particular timing guarantees, performance evaluation as well as the ability to synthesize sage and near-optimal control strategies.

For the synthesis we show that the underlying simulation-based methods underlying UPPAAL Stratego including run-based reinforcement learning, Q- and M-learning.

A number of applications (floor heating, adaptive cruise control and intelligent traffic light) will be given.

3.17 Learning Adaptive Maintenance Policies for Cyber-Physical Systems

Alexis Linard (Radboud University Nijmegen, NL)

Joint work of Alexis Linard, Marcos L. P. Bueno
Main reference Alexis Linard, Marcos L. P. Bueno: "Towards Adaptive Scheduling of Maintenance for Cyber-Physical Systems", in Proc. of the Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques – 7th International Symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings, Part I, Lecture Notes in Computer Science, Vol. 9952, pp. 134–150, 2016.
URL http://dx.doi.org/10.1007/978-3-319-47166-2_9

Scheduling and control of Cyber-Physical Systems (CPS) are becoming increasingly complex, requiring the development of new techniques that can effectively lead to their advancement. This is also the case for failure detection and scheduling component replacements. The large number of factors that influence how failures occur during operation of a CPS may result in maintenance policies that are time-monitoring based, which can lead to suboptimal scheduling of maintenance. We investigate [1] how to improve maintenance scheduling of such complex embedded systems, by means of monitoring in real-time the critical components and dynamically adjusting the optimal time between maintenance actions. The proposed technique relies on machine learning classification models in order to classify component failure cases vs. non-failure cases, and on real-time updating of the maintenance policy of the sub-system in question. We modeled our simulations in Uppaal, a model checking tool. The results obtained from the domain of printers show that a model that is responsive to the environmental changes can enable consumable savings, while keeping the same product quality, and thus be relevant for industrial purposes.

References

1 Linard, A.; de Paula Bueno, M. L.: Towards adaptive scheduling of maintenance for cyber-physical systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I. LNCS, vol. 9952, pp. 134–150. Springer, Cham (2016)

3.18 Graph-Based Reductions for Model Checking and Learning MDPs

Guillermo A. Pérez (Free University of Brussels, BE)

Joint work of Suda Bharadwaj, Stephane Le Roux, Guillermo A. Pérez, Ufuk Topcu

We study the never-worse relation (NWR) for Markov decision processes with an infinite-horizon reachability objective. A state q is never worse than a state p if the maximal probability of reaching the target set of states from p is at most the same value from q, regardless of the probabilities labelling the transitions. Extremal-probability states, end components, and essential states are all special cases of the equivalence relation induced by the NWR. Using the NWR, states in the same equivalence class can be collapsed. Then, actions leading to sub-optimal states can be removed.

Our main results are as follows.

1. We show that the natural decision problem associated to computing the NWR is coNP-complete.
2. We also give a polynomial-time iterative algorithm to under-approximate the NWR.

Among other applications, NWR-based MDP reductions can be seen as a pre-processing of MDPs before model checking or as a way to reduce the number of experiments required to obtain a good approximation of an unknown MDP.

References

1 Stéphane Le Roux, Guillermo A. Pérez: The Complexity of Graph-Based Reductions for Reachability in Markov Decision Processes. FoSSaCS 2018: 367–383
2 Suda Bharadwaj, Stéphane Le Roux, Guillermo A. Pérez, Ufuk Topcu: Reduction Techniques for Model Checking and Learning in MDPs. IJCAI 2017: 4273–4279

3.19 Using Machine Learning Techniques for Verification of Configuration Files

Ruzica Piskac (Yale University – New Haven, US)

Joint work of Ruzica Piskac, Mark Santolucito, Ennan Zhai
Main reference Mark Santolucito, Ennan Zhai, Rahul Dhodapkar, Aaron Shim, Ruzica Piskac: "Synthesizing configuration file specifications with association rule learning", PACMPL, Vol. 1(OOPSLA), pp. 64:1–64:20, 2017.
URL http://dx.doi.org/10.1145/3133888

In this talk we show how to learn specification, using verification for configuration files, when the given examples is actually a set of configuration files. Software failures resulting from configuration errors have become commonplace as modern software systems grow increasingly large and more complex. The lack of language constructs in configuration files, such as types and grammars, has directed the focus of a configuration file verification towards building post-failure error diagnosis tools. We describe a framework which analyzes data sets of correct configuration files and derives rules for building a language model from the given data set. The resulting language model can be used to verify new configuration files and detect errors in them.

References

1 Mark Santolucito, Ennan Zhai, Rahul Dhodapkar, Aaron Shim, Ruzica Piskac: Synthesizing configuration file specifications with association rule learning. PACMPL 1(OOPSLA): 64:1–64:20 (2017)

2 Mark Santolucito, Ennan Zhai, Ruzica Piskac: Probabilistic Automated Language Learning for Configuration Files. CAV (2) 2016: 80–87

3.20 Verification and Design of Rectifier Networks as Controllers

Hasan Poonawala (Univ. of Texas at Austin, US)

Joint work of Hasan Poonawala, Ufuk Topcu

Robotic systems must operate autonomously in environments that are partially known, by relying on complex sensor measurements for control and decision making. A common approach for dealing with this scenario is to design controllers from previously collected sensor data using machine learning. The interaction of dynamics and machine learning errors can lead to suboptimal or even unsafe behavior, such as crashes of autonomous mobile robots. I describe methods to model control strategies that use rectifier networks (a popular type of deep learning architecture) for converting sensor measurements into control signals. The closed-loop model is a piece-wise linear (PWL) continuous-time dynamical system, whose safety and stability properties we can verify using PWL Lyapunov functions and PWL barrier certificates, by solving a linear program. More interestingly, we can design the rectifier network's parameters, by solving a bilinear program. We present an example involving navigation of a mobile robot using different optical sensors. The Lyapunov functions and barrier functions in these examples are chosen by hand. Ideally, we would like to automatically choose these functions based on the closed-loop dynamics, without human intervention. I discuss the challenges to developing such an automatic procedure, and avenues for applications of ideas from model checking of hybrid systems to this task.

3.21 A gentle introduction to games played on graphs

Jean-François Raskin (Free University of Brussels, BE)

This talk gives a quick overview of the models and concepts used for reactive synthesis. It reviews notions of game graphs, infinite duration games, omega-regular winning objectives, strategies, and it gives elements of the algorithms underlying the synthesis of winning strategies. Finally, it considers how two-player games can be combined with Markov Decision Processes to provide models and algorithms able to synthesize strategies that enforce some key properties with certainty and good expectation for other soft properties.

3.22 An introductory tutorial to Bayesian Machine learning and Gaussian Processes

Guido Sanguinetti (University of Edinburgh, GB)

In this talk, I give a tutorial overview of Bayesian machine learning methods, with a particular focus on Gaussian Processes, a nonparameteric Bayesian model for regression which works by imposing a prior distribution directly on a space of function. The talk is preparatory to the material covered by Luca Bortolussi on his talk on smoothed Model Checking.

3.23 Learning a SAT Solver from Single-Bit Supervision

Daniel Selsam (Stanford University, US)

Joint work of Daniel Selsam, Matthew Lamm, Benedikt Bunz, Percy Liang, Leonardo de Moura, David L. Dill
Main reference Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, David L. Dill: "Learning a SAT Solver from Single-Bit Supervision", CoRR, Vol. abs/1802.03685, 2018.
URL http://arxiv.org/abs/1802.03685

We present NeuroSAT, a message passing neural network that learns to solve SAT problems after only being trained as a classifier to predict satisfiability. Although it is not competitive with state-of-the-art SAT solvers, NeuroSAT can solve problems that are substantially larger and more difficult than it ever saw during training by simply running for more iterations. Moreover, NeuroSAT generalizes to novel distributions; after training only on random SAT problems, at test time it can solve SAT problems encoding graph coloring, clique detection, dominating set, and vertex cover problems, all on a range of distributions over small random graphs.

3.24 Oracle-Guided Synthesis of Machine Learning Models

Sanjit A. Seshia (University of California – Berkeley, US)

Main reference Susmit Jha, Sanjit A. Seshia: "A theory of formal synthesis via inductive learning", Acta Inf., Vol. 54(7), pp. 693–726, 2017.
URL http://dx.doi.org/10.1007/s00236-017-0294-5
Main reference Sanjit A. Seshia, Dorsa Sadigh: "Towards Verified Artificial Intelligence", CoRR, Vol. abs/1606.08514, 2016.
URL http://arxiv.org/abs/1606.08514

We consider the problem of designing machine learning models used within a larger system that must satisfy a formal specification, a step towards the goal of verified artificial intelligence (AI) [4]. This problem is an instance of a class of problems termed as formal inductive synthesis [5]. An illustrative example is the use of deep neural networks for perception in an autonomous driving system. We present a compositional falsification approach that combines a falsifier for cyber-physical system (CPS) models with a machine learning (ML) analyzer that performs a more detailed analysis of a machine learning model [1]. The ML analyzer performs semantic transformations to input data (images) to generate new data so as to find system-level counterexamples (e.g. safety violations). We show how retraining the models with generated images can both improve accuracy and eliminate system-level counterexamples [2]. Such counterexample-guided retraining is an instance of oracle-guided inductive synthesis, and may also be seen as a "semantic" approach to adversarial machine learning [3]. We describe our results using oracle-guided synthesis of ML models for autonomous driving.

References

1 Tommaso Dreossi, Alexandre Donze, and Sanjit A. Seshia. *Compositional Falsification of Cyber-Physical Systems with Machine Learning Components*. Proc. NASA Formal Methods Symposium (NFM), May 2017.
2 Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Kurt Keutzer, Alberto Sangiovanni-Vincentelli, and Sanjit A. Seshia. *Counterexample-Guided Data Augmentation*. Proc. International Joint Conference on Artificial Intelligence (IJCAI), July 2018.
3 Tommaso Dreossi, Somesh Jha, and Sanjit A. Seshia. *Semantic Adversarial Deep Learning*. In 30th International Conference on Computer Aided Verification (CAV), July 2018.
4 Sanjit A. Seshia, Dorsa Sadigh, and S. Shankar Sastry. *Towards Verified Artificial Intelligence*. ArXiv e-prints, July 2016.
5 Susmit Jha and Sanjit A. Seshia. *A Theory of Formal Synthesis via Inductive Learning*. Acta Informatica, 54(7):693–726, 2017.

3.25 Interpretability and Expressiveness of the ML/Synthesis boundary

Armando Solar-Lezama (MIT – Cambridge, US)

The talk describes some recent work applying ideas from synthesis and FM to problems in ML, such as interpreting a decision made by a neural network, as well as applying ideas from ML to make synthesis more efficient and expressive.

3.26 Adversarial Risk and the Dangers of Evaluating Against Weak Attacks

Jonathan Uesato (Google DeepMind – London, GB) and Pushmeet Kohli (Google DeepMind – London, GB)

Joint work of Jonathan Uesato, Brendan O'Donoghue, Aäron van den Oord, Pushmeet Kohli
Main reference Jonathan Uesato, Brendan O'Donoghue, Aäron van den Oord, Pushmeet Kohli: "Adversarial Risk and the Dangers of Evaluating Against Weak Attacks", CoRR, Vol. abs/1802.05666, 2018.
URL http://arxiv.org/abs/1802.05666

This paper investigates recently proposed approaches for defending against adversarial examples and evaluating adversarial robustness. The existence of adversarial examples in trained neural networks reflects the fact that expected risk alone does not capture the model's performance against worst-case inputs. We motivate the use of *adversarial risk* as an objective, although it cannot easily be computed exactly. We then frame commonly used attacks and evaluation metrics as defining a tractable surrogate objective to the true adversarial risk. This suggests that models may be *obscured* to adversaries, by optimizing this surrogate rather than the true adversarial risk. We demonstrate that this is a significant problem in practice by repurposing gradient-free optimization techniques into adversarial attacks, which we use to decrease the accuracy of several recently proposed defenses to near zero. Our hope is that our formulations and results will help researchers to develop more powerful defenses.

3.27 Active learning of state machines

Frits Vaandrager (Radboud University Nijmegen, NL)

Main reference Frits W. Vaandrager: "Model learning", Commun. ACM, Vol. 60(2), pp. 86–95, 2017.
URL http://dx.doi.org/10.1145/2967606

In this tutorial, I review the basic theory of active learning of state machines and recent applications in which this theory was used to learn models of (and find bugs in) smart cards, implementations of network protocols, and embedded systems controllers. I discuss some recent results and outline research challenges.

3.28 Learning from Demonstrations with High-Level Side Information

Min Wen (University of Pennsylvania – Philadelphia, US), Ivan Papusha, and Ufuk Topcu (University of Texas – Austin, US)

Main reference Min Wen, Ivan Papusha, Ufuk Topcu: "Learning from Demonstrations with High-Level Side Information", in Proc. of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, pp. 3055–3061, ijcai.org, 2017.
URL http://dx.doi.org/10.24963/ijcai.2017/426

We consider the problem of learning from demonstration, where extra side information about the demonstration is encoded as a co-safe linear temporal logic formula. We address two known limitations of existing methods that do not account for such side information. First, the policies that result from existing methods, while matching the expected features or likelihood of the demonstrations, may still be in conflict with high-level objectives not explicit in the demonstration trajectories. Second, existing methods fail to provide a priori guarantees on the out-of-sample generalization performance with respect to such high-level goals. This lack of formal guarantees can prevent the application of learning from demonstration to safetycritical systems, especially when inference to state space regions with poor demonstration coverage is required. In this work, we show that side information, when explicitly taken into account, indeed improves the performance and safety of the learned policy with respect to task implementation. Moreover, we describe an automated procedure to systematically generate the features that encode side information expressed in temporal logic.

Participants

- Alessandro Abate
University of Oxford, GB
- Erika Abraham
RWTH Aachen University, DE
- Ezio Bartocci
TU Wien, AT
- Roderick Bloem
TU Graz, AT
- Luca Bortolussi
University of Trieste, IT
- Tomáš Brázdil
Masaryk University – Brno, CZ
- Marc Brockschmidt
Microsoft Research UK – Cambridge, GB
- Rudy Bunel
University of Oxford, GB
- Michael Carbin
MIT – Cambridge, US
- Rayna Dimitrova
University of Leicester, GB
- Krishnamurthy Dvijotham
Google UK – London, GB
- Rüdiger Ehlers
Universität Bremen, DE
- Andreas Berre Eriksen
Aalborg University, DK
- Radu Grosu
TU Wien, AT
- Arnd Hartmanns
University of Twente, NL
- Laura Humphrey
AFRL – Wright Patterson, US
- Manfred Jaeger
Aalborg University, DK
- Nils Jansen
Radboud University Nijmegen, NL
- Sebastian Junges
RWTH Aachen University, DE
- Joost-Pieter Katoen
RWTH Aachen University, DE
- Pushmeet Kohli
Google DeepMind – London, GB
- Jan Kretinsky
TU München, DE
- Kim Guldstrand Larsen
Aalborg University, DK
- Alexis Linard
Radboud University Nijmegen, NL
- Tobias Meggendorfer
TU München, DE
- Daniel Neider
MPI-SWS – Kaiserslautern, DE
- Guillermo A. Pérez
Free University of Brussels, BE
- Ruzica Piskac
Yale University – New Haven, US
- Hasan Poonawala
Univ. of Texas at Austin, US
- Pavithra Prabhakar
Kansas State University – Manhattan, US
- Jean-Francois Raskin
Free University of Brussels, BE
- Guido Sanguinetti
University of Edinburgh, GB
- Daniel Selsam
Stanford University, US
- Sanjit A. Seshia
University of California – Berkeley, US
- Armando Solar-Lezama
MIT – Cambridge, US
- Ufuk Topcu
University of Texas – Austin, US
- Jana Tumova
KTH Royal Institute of Technology – Stockholm, SE
- Jonathan Uesato
Google DeepMind – London, GB
- Frits Vaandrager
Radboud University Nijmegen, NL
- Min Wen
University of Pennsylvania – Philadelphia, US
- Leonore Winterer
Universität Freiburg, DE

Report from Dagstuhl Seminar 18122

Automatic Quality Assurance and Release

Edited by

Bram Adams[1], Benoit Baudry[2], Sigrid Eldh[3], and Andy Zaidman[4]

1 **Polytechnique Montreal, CA,** `bram.adams@polymtl.ca`
2 **KTH Royal Institute of Technology – Stockholm, SE,** `baudry@kth.se`
3 **Ericsson AB – Stockholm, SE,** `sigrid.eldh@ericsson.com`
4 **TU Delft, NL,** `a.e.zaidman@tudelft.nl`

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 18122 "Automatic Quality Assurance and Release". The main goal of this seminar was to bridge the knowledge divide on how researchers and industry professionals reason about and implement DevOps for automatic quality assurance. Through the seminar, we have built up a common understanding of DevOps tools and practices, but we have also identified major academic and educational challenges for this field of research.

Seminar March 18–21, 2018 – https://www.dagstuhl.de/18122
2012 ACM Subject Classification Software and its engineering → Software configuration management and version control systems, Software and its engineering → Software creation and management, Software and its engineering → Software verification and validation, Software and its engineering → Software post-development issues, Software and its engineering → Collaboration in software development, Software and its engineering → Software infrastructure
Keywords and phrases DevOps, automated quality assurance, Continuous Integration, Continuous Deployment, software testing
Digital Object Identifier 10.4230/DagRep.8.3.94
Edited in cooperation with Gerald Schermann

1 Executive Summary

Bram Adams (Polytechnique Montreal, CA)
Benoit Baudry (KTH Royal Institute of Technology – Stockholm, SE)
Sigrid Eldh (Ericsson AB – Stockholm, SE)
Andy Zaidman (TU Delft, NL)

The seminar explored the relationship between DevOps and quality assurance from a software engineering perspective. DevOps has been gaining traction since around 2012, with initiatives formed both in industry and academia. While the importance of DevOps as an enabler in higher quality software is intuitively clear to both industry and academia, we have discussed commonalities in views, but also the challenges that lie ahead for this discipline.

In essence, human factors are very important, because DevOps is not only a technology, it is a way of working and organizing teams. In this light, we have also discussed the resistance that some team members or even entire organisations seem to have towards automating quality assurance through DevOps. Section 4.2 summarizes a group discussion that eventually triggered a set of reflections on this topic of human aspects of DevOps. Yet, we have also discussed how DevOps can be an enabler for onboarding new team members through the availability of a standardized DevOps infrastructure (Section 4.4). The whole group observed the general lack of empirical evidence on the importance and benefits of DevOps in modern software engineering. This final point is tightly connected to another important theme in our discussion: educating software engineers in the ways and associated technologies of DevOps.

The main goal of this seminar was to bridge the knowledge divide on how researchers and industry professionals reason about and implement DevOps for automatic quality assurance. Through the seminar, we have built up a common understanding of DevOps tools and practices, but we have also identified major challenges for this field of research as well as for the teaching of DevOps principles and practices.

This Dagstuhl was a 2.5 day seminar, which we structured around 4 invited talks that served as keynotes to introduce key topics for discussions. These talks, summarized in Sections 3.1 through 3.3, were given at the beginning of each morning and afternoon to inspire topics for further discussions on a given topic. The group split into smaller sub-groups after each keynote, in order to focus discussions and reflections on a specific topic. All these discussions have been summarized in the form of a blog post, while in Dagstuhl, and are provided in this report.

In addition to keynotes and subgroup discussions, we had a plenary session to start the seminar, where each participant had 2 slides for a short introduction; we had a “speed-dating” session on Tuesday evening; and we organized a panel discussion about the future of the field on the last morning (Section 6.3).

2 Table of Contents

3 Overview of Talks

3.1 Understanding the DevOps Concept

Lucy Ellen Lwakatare (University of Oulu, FI)

The DevOps concept in the software industry was first coined in 2009. A decade later, its actual meaning, characteristics and impacts are starting to get established in the software industry and in academia. The key idea of DevOps is the automation of activities that span software development and operations, in order to rapidly and reliably release software changes to the target environment. This requires a change of mindset and consideration of other non-technical aspects, especially when taken into context inside an organization. However, the practices promoted by DevOps do not ignore prior research, since DevOps is "standing on the shoulders of giants" in the form of Agile and Lean practices.

3.2 Perspectives on Teaching a DevOps Course

Christopher J. Parnin (North Carolina State University – Raleigh, US)

Continuous deployment is a software engineering process where incremental software changes are automatically tested and frequently deployed to production environments. Unfortunately, the skills required in automating and accelerating changes to production infrastructure require expertise and training that is even more rare and highly sought than data science skills. Parnin has been teaching a DevOps course since 2015. In the course, students learn how to build an automated deployment pipeline and manage the infrastructure required by a program.

To support the development of the concepts and practices related to this course, Parnin co-organized continuous deployment summits with companies such as Google, Facebook, and Netflix to understand how industry is using these tools and processes in practice. To support learning, Parnin is using active learning and mutual engagement through the use of workshops, which enable students to work hands-on with tools and code examples. Students primarily work on course projects with four milestone deliverables: configuration management + build; test + analysis; deployment + infrastructure; monitoring + operation. An example task for a milestone would be: provisioning and configuring a Jenkins server automatically using Ansible.

Teaching challenges include costs and risks associated with operating infrastructure, students left behind because of skill gaps, and difficulty of assessing the quality of infrastructure for grading and evaluation. Curriculum challenges include keeping practices up to date, deciding on the balance between teaching tools vs. ideas and the need to validate the processes and practices with empirical studies.

3.3 Agile Transformation Journey to DevOps from a Test Quality Perspective at Ericsson

Sigrid Eldh (Ericsson AB – Stockholm, SE)

The Agile process paradigm changed the way industries work, despite the lack of scientific basis. Agile ways of working promised fast deliveries and high changeability. During Ericsson's transformation of its Ultra-Large-Scale Systems, a key focus was to establish and evaluate what was lost and gained in this fundamental shift. Test automation improvements are in the center of this change supported by the continuous integration practices. Closing the gap to operations by improving upon the release procedures is still complicated, since it is aiming for manual steps to be fully automated. As Ericsson's vast experience on non-functional tests and systems view are key practices, e.g., robustness testing, one can still find difficulties in fully automating these abilities, as they heavily depend on the hardware, i.e., test environment. Complete system "end-to-end" practices are key elements of quality assurance. The goal to test with operation data used directly to achieve a more realistic execution profile, is performed through capturing data with real-time analytics and e.g. utilizing data from various logs. The early use of actual data for test, will change the way testing is performed, and will be a grand shift enabling new opportunities to assess quality.

3.4 Challenges of Automatic QA and Release in Industry

Hyrum K. Wright (Duolingo – Pittsburgh, US)

Drawing on years of experience in open source, large companies, and startups, Hyrum described the two major factors involved in implementing good QA and release processes, as well as a challenge for those present to impose the state of the art into practice. The two areas of focus are the social and the technical issues involving solid QA and release processes, with the social issues being the most difficult to overcome. Our challenge as a community is to build technology and standards that enable social improvement in areas such as testing and automation. Time will tell if we can rise to the challenge.

4 Working Groups

The topic of quality assurance and release covers multiple areas. It includes not only DevOps-concepts from a technical perspective (e.g., build, test, deployment automation), but also human factors as it changes how teams interact with each other. Furthermore, in the breakout groups we discussed differences between industry and open source projects (e.g., the process of "onboarding" new employees or contributors to a project), how we can transfer verification or testing techniques from software engineering to the DevOps world (e.g., applying those concepts on the pipeline-level), and how bots can support developers and release engineers.

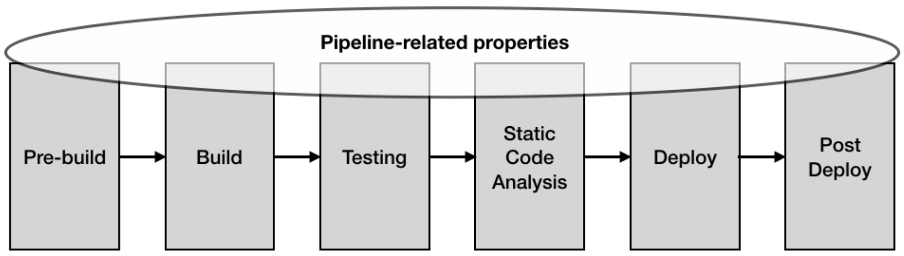


Figure 1 Various stages of a pipeline.

4.1 Desirable Properties of Pipelines and How to Verify Them

Bram Adams (Polytechnique Montreal, CA), Foutse Khomh (Polytechnique Montreal, CA), Philipp Leitner (Chalmers University of Technology – Göteborg, SE), Shane McIntosh (McGill University – Montreal, CA), Sarah Nadi (University of Alberta – Edmonton, CA), Andrew Neitsch (Cisco Systems Canada Co. – Toronto, CA), Christopher J. Parnin (North Carolina State University – Raleigh, US), Gerald Schermann (University of Zurich, CH), Weiyi (Ian) Shang (Concordia University – Montreal, CA), and Hui Song (SINTEF ICT – Oslo, NO)

Release/deployment/delivery pipelines, what we call "pipelines", are becoming fundamental artifacts in modern software organizations. These pipelines automate different stages of a deployment process and have the ability to filter out poor quality commits and target specific delivery endpoints (such as a staging or production environment). Figure 1 shows an overview of such a pipeline. Broadly speaking, a pipeline includes:

1. *Pre-build*: Making decisions about which configurations/platforms/variants to check throughout the pipeline.
2. *Build*: Transforming the sources into deliverable format.
3. *Testing*: Execution of various types of automated tests (e.g., unit, integration, performance, regression).
4. *Static Code Analysis*: Automated analysis of source code for common problematic patterns.
5. *Deploy*: Shipping newly built deliverables to staging and production environments.
6. *Post Deploy*: Making sure that the released application performs/behaves as expected (e.g., through monitoring and production testing).

The recent shift towards continuous deployment has resulted in the need to deploy changes into production environments at an ever faster pace. With continuous deployment, the elapsed time for a change made by a developer to reach a customer can now be measured in days or even hours. Such ultra-fast and automatic changes to production means that testing and verifying the design and implementation of the pipelines is increasingly important. The high-level idea is that pipeline artifacts are also susceptible to the same sorts of problems that we encounter when writing code (e.g., defects, anti-patterns); however, pipeline quality assurance practices are rarely applied (if at all).

So how can we apply quality assurance practices to pipelines? In this blog post, we begin by defining properties that stakeholders would consider desirable (or undesirable) in the pipeline and each phase within it. We provide examples of what could go wrong if a property is violated and some potential avenues for verifying these properties in practice. Finally, future research and tool smiths can leverage these properties in the design of better pipeline engines and tools.

These observations are based on brief discussions between DevOps researchers and practitioners. Our goal is not to advocate for the completeness of this list, but rather to start a discussion within the community around these properties.

4.1.1 Overarching Pipeline Properties

- *Scalability/Cost*: A pipeline needs to be able to deal with a high load of commits coming in
- *Repeatability*: Treating a pipeline as an artifact itself, installing the same pipeline under the same configuration should lead to the exact same results/outcome when fed with the same data
- *Simplicity*: It should be easy to understand of what phases/stages a pipeline consists of and how changes flow through the pipeline.
- *Trustworthiness*: Trusting tools or bots that have privileges to modify the configuration or properties of the pipeline.
- *Security*: Making sure that all phases/steps along the pipeline are reasonably secured (e.g., ports only open when needed)
- *Robustness/Availability*: Involves the different phases/steps that need to be up and running (e.g., staging environments).
- *Velocity/Speed*: The time needed to run through all the phases of the pipeline.
- *Traceability*: At which specific stage/phase of a pipeline is a (code) change currently? Is the pipeline able to route each change along the right sequence of tasks and to detect each problem as soon as possible?

4.1.2 Do I really need to care about the pipeline?

At this point, you may be wondering how violations of the above properties manifest themselves as problems in practice. My code works well, do I really need to care about the pipeline and this long list of properties you provide? The answer is yes! Before diving into the details of each of the above properties and how they are related to each stage of the pipeline, let us talk about some real-world examples first.

Starting with the importance of the scalability of the pipeline: Major projects like Openstack receive so many commits per hour and have thousands of tests to be executed, such that they are not able to trigger the full CI build for each individual commit. Instead, OpenStack's testing automation team[1] came up with custom algorithms to run the CI build on a group of commits, while not losing the ability of singling out the specific commit responsible for failing the (group) build.

Another property to think about is security. If my product code is secure, as well as that of the data servers it connects to etc., what can possibly go wrong? Well, the server that you eventually deploy to, e.g., for others to download your application from, may itself be insecure. A recent example is Eltima's Media Player[2], where a trojan was inserted in the Mac app after it was already deployed to the download server. An older example is a backdoor code change that was added into the Linux kernel CVS mirror of the official BitKeeper repo[3], without any trace where it came from (i.e., it seemingly escaped code review, and likely was

[1] https://archive.fosdem.org/2014/schedule/event/openstack_testing_automation/
[2] https://www.macrumors.com/2017/10/20/eltima-software-infected-with-malware/
[3] https://freedom-to-tinker.com/2013/10/09/the-linux-backdoor-attempt-of-2003/

Table 1 Overview of pipeline properties and in which phases they should be enforced.

	Pipeline	Pre-build	Build	Testing	Static Code Analysis	Deploy	Post Deploy
Scalability	✓			✓			
Repeatability	✓		✓	✓		✓	✓
Simplicity/Understandability	✓						
Trustworthiness	✓						
Security	✓						
Robustness	✓						
Availability	✓					✓	✓
Velocity/Speed	✓		✓	✓	✓	✓	
Cost	✓						
Coverage		✓		✓			
Incrementality			✓		✓	✓	
Correctness			✓		✓	✓	
Regularity							✓
Traceability	✓						✓
Independence			✓				

hacked into the repository). Luckily, it did not end up in kernel releases, since the commit was just inserted into a mirror, but who knows what would have happened if this was not the mirror server!

4.1.3 What are examples of what I can do to avoid this?

In terms of scalability, one solution is that instead of letting pipelines process and/or run every type of file (e.g., shell script) and third-party library as part of a code change that is checked in, the pipeline should verify the type of changes that are allowed to trigger the pipeline, as well as specify explicitly who is allowed to check in what (humans, bots, ...).

To ensure that all servers you use in your pipeline have the same properties (e.g., security, installed software etc.), we can use the concept of Pipeline-as-Code (e.g., https://jenkins.io/blog/2017/09/25/declarative-1/), which allows specifying the pipeline infrastructure in textual form and to automatically instantiate the pipeline, similar to infrastructure-as-code. This enables instantiating identical pipeline instances across different machines, to track changes in the specification (Git), etc. Another option is to implement the concept of *flow testing*. *Flow testing* can be performed in the style of integration testing by validating that a given test commits indeed performs as expected. Did the pipeline hit the major pipeline steps that it should hit (build, test, deployment etc.). For a commit that causes a performance problem, was this problem caught at the right stage and flagged? Each kind of problem requires a "barrage" point behind which we do not want the problem to pass. Finally, specific steps can be taken to ensure the security of all pipeline servers by using checksums and reproducible builds (i.e., a build process that always generates the same binary, bitwise, for a given input), which can further avoid intermediate tampering with build artifacts.

Now that we have given you a big picture of things, let's get down to the details a bit. Table 1 summarizes the different stages of the pipeline and the properties we believe each stage should enforce. In the following sections, we discuss these properties in more detail and provide examples of what can be done to enforce these properties to avoid problems such as those previously described.

Pre-build

Desirable properties:

- *Cost*: The minimal set of configurations/variants/branches should be used within the subsequent phases (Tradeoff with Coverage)
- *Coverage*: The largest set of configurations/variants/branches should be used within subsequent phases (Tradeoff with Cost)
- *Understandability*: How could the engineers understand the configurations as specified in the pipeline

Example problem: It is very easy to waste resources on running test cases on multiple configurations.

Example verification: Through different traceability links, the pipeline can assess if a test case touches certain configuration points and plan accordingly by reducing the number of representative configurations.

Build

Desirable properties:

- *Correctness*: Dependencies are fully expressed.
- *Velocity/Speed*: Builds are completed in a reasonable amount of time (reasonable varies from project to project).
- *Incrementality*: Builds (re)execute the minimal subset of necessary commands to update deliverables without missing any necessary commands.
- *Repeatability*: Given the same input, builds perform the same commands (i.e., deterministic builds). Moreover, it should be possible to reproduce a past build in the future (e.g., if a package service goes down or is no longer available).
- *Independence*: Builds should be isolated from each other.

Example problem: A build phase that does not have the independence property may suffer from builds that interfere with each other. In turn, the build phase may become non-deterministic. For example, builds that are running in parallel or sequentially may access resources from each others' environments by mistake.

Example verification: A possible step towards checking this property could be to apply a form of taint analysis, i.e., track all outputs of a build and check who reads those outputs. Taint analysis has been effectively applied to the analysis of the surface area that is exposed to security issues (e.g., SQL injections). The same concepts may apply to the leakage of state within the scope of builds.

Testing

Desirable properties:

- *Scalability/Cost*: The testing stage needs to be "smart" about its decisions. Depending on the size of the test suite and types of tests available, not every test needs to be run for each commit. Only tests affected by updated artifacts should be run. Good traceability links between code and the test suites, as well as test prioritization can be used to make the testing stage more scalable.
- *Repeatability*: Running the same test suite on the same commit should always produce the same result in terms of passed and failed tests. Flaky tests[4] are especially problematic for repeatability. One solution is to identify/flag flaky tests in order to have special handling for them.
- *Velocity/Speed*: Execution time of test suites is a major bottleneck in the overall velocity of the pipeline. In this phase, velocity/speed is related to the scalability/cost property since smarter test selection will probably eventually lead to better speed of the testing phase.
- *Coverage*: As many possible variants/configurations of a product need to be tested, without sacrificing speed.

Example problem: If no test prioritization/selection strategy is used, large amounts of testing resources can be wasted on not impacted artifacts, delaying the delivery.

Example verification: With a predefined set of mappings between code and tests, given the code changes, the pipeline should trigger and only trigger those tests.

Static Code Analysis

Desired properties:

- *Incrementality*: Only the needed analysis is applied in the release pipeline. For example, the static analysis is only applied on the code that is impacted by the code change. (may remove: Capture the intended properties of the analysis)
- *Correctness*: A static analysis should yield a low rate of false positives, since false positives reduce the trustworthiness of results and lead of gain adoption from the practitioners.
- *Performance*: The static code analysis should be able to finish within a reasonable time, since a long duration of the analysis will affect the deliverable of the product into the next step in the pipeline (e.g., testing or deployment).

Example problem: A typical off-the-shelf static analysis tool often report a large number of issues, while not all of them are of interest to impact the release pipeline. Reporting all the issues, or having all the issues to determine the next step in the pipeline is problematic in practice.

Example verification: A dataset with the issues that may be detected by static code analysis needs to be labeled into whether they are of interest of the practitioners to impact the release pipeline. To test the correctness of the static analysis, a randomly generated sample from the dataset is the test input of this phase and the precision and recall with threshold can be used to assert whether the output is satisfactory.

[4] https://testing.googleblog.com/2016/05/flaky-tests-at-google-and-how-we.html

Deploy

Desired properties:

- *Repeatability*: The deployment results of the software should not be impacted by the deployment environment.
- *Availability*: It is desirable that the service is continuously available during deployment, instead of having to choose between either working nights/weekends to release, or bringing down the service during peak times.
- *Velocity*: A regular release rhythm and/or immediate releases that let developers say that their feature is truly "done done" since it is released to production, can aid development velocity.
- *Incrementality*: An incremental release that initially sends only a fraction of traffic to the new version, and is ramped up until the new version is handling all traffic, can limit the "blast radius" of a problematic release.
- *Correctness*: Deploying software can be risky. A correct process that will not leave the service in a broken state on error is highly desirable.

Example problem: The new version of the service requires configuration for a new feature, but the new configuration has not been applied to the target environment. After deploy, the new version crashes on startup with an error message about invalid configuration.

Example verification: Recently added and changed acceptance tests are run against the newly-deployed service before it is exposed to external clients. Automatic rollback is triggered if the service crashes, or the tests fail.

Post Deploy

Desirable properties:

- *Availability*: Are the deployed artifacts available (e.g., appropriate ports open, heartbeat)
- *Regularity*: Configuration and code changes perform within nominal operational metrics.
- *Traceability*: Data is "collectable" on a continuous basis from various parts of the system, and configuration changes should be auditable and mapable to code changes.
- *Repeatability*: To what extent is the infrastructure resilient to changes in external dependencies, versions, and tools.

Example problem: Inadvertently reusing a feature flag's name can make dead code active again, as happened with the Knight Capital bankruptcy.

Example verification: Turning on an old feature flag could violate two properties: a) *Traceability*, code associated with a feature flag may not have been recently changed, which could set off a warning. b) *Regularity*, the performance of the code with the wrong feature flag may generate metrics that are not consistent with recent range of metrics.

4.2 Human Factors in DevOps

Lucy Ellen Lwakatare (University of Oulu, FI), Tamara Dumic (Ericsson Research – Stockholm, SE), Sigrid Eldh (Ericsson AB – Stockholm, SE), Daniele Gagliardi (Engineering Ingegneria Informatica S.p.A – Roma, IT), Andy Zaidman (TU Delft, NL), and Fiorella Zampetti (University of Sannio – Benevento, IT)

4.2.1 Introduction

Change of process, ownership and technology has a direct impact on individuals, organizations and groups of people. DevOps is a process that includes philosophy, mindset changes, and also changes in human interactions and communications. Tackling human factors is beyond the mere technology shift that DevOps introduces. One can say that transforming to DevOps is a software process change that embodies the human performing the process as an essential part. Specifically, DevOps adoption implies a transformation of the entire software community including the way to release, monitor, and interact with the users of the system. Through its extensive approach to human tasks automation, it also transforms ownership, organization and could be considered a major technology change in the way we create, produce and use the software. As a process, it enables new technologies and makes data collection shared and available for analysis in a completely new way. The following items summarize the five main challenges that we have identified as key topics belonging to the human factors of process change:
1. It is hard to convince people to do/adopt DevOps
2. Need to have a shared approach/vision of all stakeholders/organizations involved
3. Difficulty in accountability/taking ownership for people
4. Overcoming fear of new technology and change
5. Other topics

For each of these five challenges, we have clarified the context and given examples in order to avoid "misunderstandings". Moreover, when possible, we have highlighted some possible solutions in addressing them.

We have identified and observed a number of signs of difficulties in the DevOps transformation, e.g., resisting change, split responsibility, ownership, and assumptions and expectations on the software process and life-cycle that might not be accurate, as well as basic lack of skills in different fields to enable the process change. This list can be expanded and discussed, to aid others in the DevOps change.

Moreover, it is important to highlight that, in our discussion, we have also identified some key human factors that go beyond the scope of this challenge that could be addressed by expertise in other scientific communities, enabling cross-science collaborations.

4.2.2 Challenges

It is hard to convince people to do DevOps

DevOps transformation and adoption in an industrial setting, including large organizations and those in public sector, requires change agents to make a convincing argument "business case" for its enactment. The lack of empirical analysis aimed at demonstrating the advantages of using a DevOps approach despite the initial effort required and the lack of contextualization inevitably implies resistance to change and, consequently, resistance to the adoption of new technologies and/or practices. Further, not having scientific evidence of the advantages of adopting a DevOps process in different contexts/domains, but having only anecdotes, makes it difficult to create a convincing argument to be used with the various stakeholders involved in the change. As a matter of fact, automation in DevOps not only requires new resources but also changes existing processes and organizational structures thereby affecting people's work and competences.

It is not unknown that DevOps emerged as a hype word in the software industry in the past decade. Leading software companies, such as Facebook, Google and Microsoft, reported a paradigm shift towards multiple release of new features to end users on a daily basis instead of the typical lengthy release cycle times. While technical aspects, and the ideas behind them, are relatively clear, onboarding them in practice requires the acknowledgment by the people involved in it. The latter is particularly true since often people either out of fear or out of laziness tend to reject new solutions when they have something equivalent and working at their disposal (solutions perceived to be sufficient). A different aspect to take into account is related to the initial investment (effort) required in order to setting up a Continuous Deployment pipeline. In this context it is very important and needed to make clear the return value (ROI) and the impacts for the different stakeholders involved (e.g., it improves release cycle times, early identification of problems and reduction in the overall costs of maintenance activities). This is not so easy to achieve since that in many cases the change process includes different stakeholders having different backgrounds and needs.

A way forward would be to ensure that people are convinced that the new practices are worth the effort and investment. In order to obtain the above goal it is very important and needed a strong collaboration between both industry and academia in performing studies that are able to highlight its benefits. Indeed, it is well known that scientific facts are more easy to be believed compared to anecdotal evidence. The lack of scientific evidence has led industries to look at DevOps and continuous deployment in general as "black magic".

Another key aspect that needs to be taken into account is related to the awareness and visibility of the whole "systems thinking" rather than isolation of system life cycle phases. Very often developers do not know what is the effect of their "little" contribution on the whole system under development. A common statement to avoid/remove is: "But it works/worked on my machine". The latter can be obtained augmenting the developers' knowledge. It is very important that each developer knows that their contribution/change can break the whole system when integrating it and this will inevitably imply that other developers/teams can be blocked as a consequence of their failures. Also in this case, scientific evidence can help in augmenting the awareness of developers working in a large group or in a large project on the fact that they can block the whole system even with a little change. Finally, another interesting point to highlight is related to the economic perspective in the sense that it can be interesting for industries to know the economic loss in missing developers' contribution awareness.

Need to have a shared/approach vision of all stakeholders/organizations involved

Implementing DevOps is not something that one single team or even one single person can decide. It requires that the development team, the operational team, and the customer are willing to do so. As an example, if the development team wants to do it, but the company is unwilling to pay for the operational/monitoring aspect, you cannot move to DevOps.

Another aspect to consider is the team composition. Indeed, many teams also include IT consultants but, if many consultancy firms are cooperating, it is required to have an agreement amongst them.

We have the feeling that the customers are the driving force behind moving to DevOps and not the service company (or development team). It is the customer that approaches them with a buzzword like DevOps and asks for it. There are also cases in which the customers want and pay for a product, without explicitly making mention of the quality assurance practices/standards that need to be upheld. This sometimes leaves the door open for service companies to cut back on quality assurance during the engineering/development phase, only to get a maintenance contract for the product that they are delivering afterwards. In other words, sometimes the customers decide to not pay for testing (and also quality checking of the software product) since their main aim is to have a prototype ready for usage. Only in a subsequent step, they will define a new contract in order to check whether the software product effectively fits their needs. As a Software Engineering community, we know that doing this will inevitably increase the cost belonging to finding defects but, most important, the cost of fixing them. DevOps adoption can reduce this cost since each developer can immediately see whether her change is inducing a failure (test case failure, not respect of rules/standards to follow in terms of quality, etc) and can easily identify where the failure is located inside her code.

As a consequence, it is very important to have a shared/common approach/vision between different stakeholders/organizations involved. The main problem to address is that, very often, different stakeholders show different backgrounds and, most important, have different goals. Of course, this is not so easy to obtain when each organization is close-minded in the sense that it does not want to look at the whole picture, but is only focused on its own needs. Finally, it is very important that in order to be competitive on the market you cannot focus only on the cost but you have to focus on the quality of what you are going to put on the market.

Difficulty in accountability/taking ownership for people

As already highlighted, DevOps adoption means automation but also collaboration between people. Obviously, collaboration implies that each party involved shall take the ownership for something inside the collaborative process but also for the relations it has with other actors. A "silo mentality" is what we actually have in different teams involved in the realization of the same software product. In other words, a team or department shares common tasks but derives their power and status from their group with no accountability about relations. "I do my best in my tasks. Period". An example can be the one related to a "Misunderstanding of requirements": when the developer appears to be responsible for a requirement not well-understood, she usually says "the functional analyst gave me wrong information". And the analyst will blame someone else for something else. This is related to the fact that often companies or, maybe better, single managers act following a "blame culture": if something goes wrong, the first action is finding the culprit, not the solution. The latter is part of the corporate culture.

A first step towards the promotion of accountability could be to define as much as possible what each actor is accountable for, and how what she does will impact the whole process: knowing the "what" is necessary to find the "how" (how can I improve to serve better the others? Do I see improvements that others can do to help me to improve?). As an example, in many situations it is not taken for granted what is Dev accountability and Ops accountability. More in general, Dev accountability is related to verification and validation of the compliance between software requirements and the product under development. On the other hand, Ops accountability is related to availability and reliability of software services in production systems. If we introduce explicitly the concept of "accountability for quality", it becomes easy to see that each actor cannot act as she was in an isolated context with no relations with others: well-defined requirements implies well-developed software that requires an effective test suite (set of test cases). If test cases are considered as “executable specifications”, they can help to define in a better way the requirements. As also suggested in the other two challenges already exploited, in developing a software product you cannot have tasks/phases considered as second-class citizens (unfortunately, this usually happens for test cases that are not written before starting the real development but also not written together with production code).

Another aspect is that acting within DevOps processes (and ultimately being accountable for something) means also to commit on continuous learning (not just new tools or new processes, but also gaining awareness of the big picture). Continuous learning can be hard to accomplish (people need time, resources). Another obstacle in being accountable for something is related to the fear of people in being responsible for something that goes wrong (different from what is desired). Related to the continuous learning it is important to consider non-technical stakeholders as functional analysts and/or clients. Since that, very often a company has to see what the clients want as a cornerstone: “the customer is always right” causes people have unsustainable pressure, and asking people for being accountable for something cannot be required.

Overcoming fear of new technology and change

A change is often associated with uncertainty and fear, especially if it has direct impact on individuals. Automation of different process steps is not an exception to that. It can lead to showing resistance and loss of motivation for some individuals, especially in cases when an existing expertise is threatened by the automation. This is typically a case when an individual has been doing a type of work that can be replaced by an automated process (for example a tool or a script). If at the same time a threshold for learning new technologies is simply too high for that individual, the resistance and loss of motivation tend to be even greater. Driving change is certainly not a trivial thing to do, especially in large organizations. It requires a lot of communication and having a dialogue with those individuals who might have concerns related to the change. Creating an environment where each individual feels included and listened to, improves chances for getting a buy-in. Also, in the process of communicating a change it is very important to keep coming back to the big picture (the reason for the change) and the benefits that the change will bring.

Other challenges

This last category is a “remaining” section of areas that could be observable issues related to human factors, and not necessarily fitting in the above sections. We will here shortly explain each of these challenges:

Awareness about what quality really is: A common theme is that people often tend to underestimate the work that is happening, especially in the context of test and quality assurance. If testing practices are not a real part of the software development, there is bound to be resistance to, e.g., automation of tests – since the people are not mature in testing – and cannot understand the long term benefits of such investments. There is not sufficient nuance in categorizing phases of company growth that can be used – as maturity on collaborations and understanding of both development, operations technical aspects of CI/CD is still flawed.

Organization set-ups: Since some organizations are set up in ways where often development and operations have separated – the ownership, and the challenge DevOps is the management of both of them – this is must engage both aspects. For some communities, this poses a huge difficulty and prevention of progress. There is not sufficient studies on solving this – and it might even not be in peoples own interest to, e.g., loose their "power" over a part in the process – when new ways are asking for shared ownership.

Education and fact-based benefits of doing things: There are two aspects here. One is that education should be based on scientific evidence – and DevOps is also addressing operations, release aspects, monitoring aspects etc. Second, gathering information and research from these areas. A special break out report on this from Dagstuhl, named "Dagstuhl – DevOps and the Need for Improved Evidence from Industrial Studies" dives deeper into this topic (see Section 4.5).

4.2.3 Solutions and Considerations

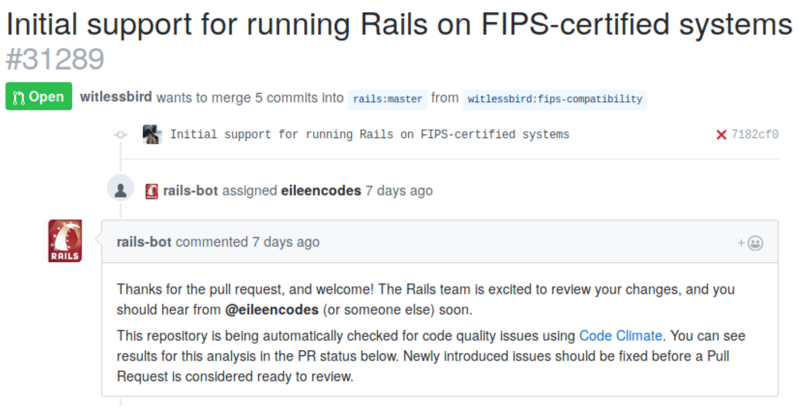
- Knowledge/education can overcome Challenge 1,2,4
- Scientific studies (proof of benefit) can overcome 2
- Trade-off between "ok to break" (and being fast and have resilient software) vs the culture of "do not break the builds"...
- Promoting the value of continuous learning can address 3: being aware of the relations and the impact requires training, going towards the concept of cross-functional teams. It is not needed that a developer becomes an expert sysadmin, but it's crucial that she knows how the quality of her code (as an example) impact on operations, and ultimately on software quality attributes (security, performance, availability, reliability)
- Blameless build features – "it is not the person, it is the process"... what process failure led to this happening...
- Despite good processes in place, humans can fail (tiredness, psychological personal situation, etc): we should accept it "by culture". But maybe good processes could prevent consequences of human failures (i.e., checklists to face the emergency, etc)
- Can we provide tools to mitigate human issues?

4.3 Bots for DevOps

Martin Monperrus (KTH Royal Institute of Technology – Stockholm, SE), Benoit Baudry (KTH Royal Institute of Technology – Stockholm, SE), Moritz Beller (TU Delft, NL), Benjamin Danglot (INRIA Lille, FR), Zhen Ming (Jack) Jiang (York University – Toronto, CA), Vincent Massol (XWIKI – Paris, FR), Oscar Luis Vera Perez (INRIA – Rennes, FR), and Hyrum K. Wright (Duolingo – Pittsburgh, US)

Parts of the automation in the DevOps pipeline are provided by so-called "bots". Who are they? What makes them "bots"? In this post, we explore the essence of bots for DevOps.

We first consider some concrete examples of bots in the context of DevOps. Then we derive a set of common characteristics from these bots. Finally, we discuss the values of these bots and various challenges associated with developing bots for DevOps.


Initial support for running Rails on FIPS-certified systems #31289

Open witlessbird wants to merge 5 commits into rails:master from witlessbird:fips-compatibility

Initial support for running Rails on FIPS-certified systems 7182cf0

rails-bot assigned eileencodes 7 days ago

rails-bot commented 7 days ago

Thanks for the pull request, and welcome! The Rails team is excited to review your changes, and you should hear from @eileencodes (or someone else) soon.

This repository is being automatically checked for code quality issues using Code Climate. You can see results for this analysis in the PR status below. Newly introduced issues should be fixed before a Pull Request is considered ready to review.

Figure 2 A welcome bot giving a greeting to a newcomer to the project. (Source: M. Beller, An Empirical Evaluation Of Feedback-Driven Development, 2018)

Examples:

Let us first consider a list of concrete examples of bots (meant to be illustrative, not comprehensive):
A bot that ...

- replaces all calls to deprecated methods with the new methods (*DEPRECATEBOT*)
- checks licensing and intellectual property (*CLABOT*)
- spells check comments and identifiers (*NAMEBOT*)
- reformats code according to some coding styles/convention (STYLEBOT)
- applies static analysis to detect bugs (*BUGBOT*)
- applies program repair techniques to fix bugs (*REPAIRBOT*)
- says "Welcome" to new contributors when they post their first pull-request (*WELCOME-BOT*)

Those bots have very different characteristics, but what is sure is that:

"A bot is not defined by the complexity / size of the task that is automated"

For instance, a WELCOMEBOT like the "rails-bot" in Figure 2 can be technically simple, but a REPAIRBOT can be very complex to develop and maintain. It seems that what makes a bot is more related to the intention behind the bot. This makes us propose not a single definition, but a set of characteristics.

4.3.1 Characteristics of DevOps Bots

To us, a bot satisfies at least one of the following definitions (logical "or", not "and"):
A bot ...

- does its job in an autonomous manner, not when humans ask for it (a.k.a., it is not a command)
- is something that is able to perform transformative actions on software artefacts, including code
- performs tasks that could not possibly be done by humans
- performs tasks that are traditionally done by humans
- outputs something that could also be produced by humans (a.k.a., producing contents which look like human-created content)

- is something that interacts with humans with the same medium as humans (e.g., in the pull-request conversation thread)
- it's output consumes a human's brain bandwidth

Many of those facets focus on the interaction with humans. To this extent, bots may be most valuable when they need some input by humans to perform a task (otherwise, it can be fully automated, e.g., by the compiler).

4.3.2 Values of Bots

Bots can add value to the DevOps pipeline in different ways:

- Bots are required when there is no way to change the root cause of a problem because it is handled by different stakeholders (e.g., you cannot change the tool "Javadoc" for handling missing dots at the end of the first sentence, so you create a bot for this).
- Second, bots are particularly interesting in the gray area between hard tasks (only done by humans) and tasks that can be completely automated (no bots needed, but an automated tool that does the job).
- Third, and quite pragmatically, bots are valuable if they do a task for which one would be ready to pay a human.

Even as known bots are deployed in production (e.g., the WELCOMEBOTS from GitHub), we present the following series of grand research challenges to embed bots in DevOps.

4.3.3 Grand Challenges

In the following, we define a series of five grand challenges in the area of DevOps.

- **Conversation:** We do not know how to build conversational code bots, i.e., bots that can propose changes and later respond to the questions of developers about the nature and the rationale of the change, and improve the proposed changes accordingly.
- **Trust:** Humans and bots interact. It is teamwork where trust plays an important role. "How to build and manage bots' reputation?" "How do software engineers develop trust in bots?" "What prevents developers from blindly trusting code bots?" These are open questions for future scientific research.
- **Interoperability:** Bot designers have to make assumptions on the software artifacts to be analyzed and transformed. How to organize a repo to get the most out of bots? Could we set standards for bot-friendly repo organization?
- **Configuration:** Current bots are mostly hard-wired at deployment time. However, rules are not all equally relevant and interesting among many different organizations. How to build self-configuring bots that automatically learn the rules, criteria, thresholds to be used?
- **Bot teams:** In the future, there will be multiple bots working on the same code base: how to set up teams of bots with different goals, characters, which together form a dream bot team?

This makes an exciting research agenda for the DevOps community.

4.4 Onboarding a Software Project

Vincent Massol (XWIKI – Paris, FR), Benoit Baudry (KTH Royal Institute of Technology – Stockholm, SE), Benjamin Danglot (INRIA Lille, FR), Daniele Gagliardi (Engineering Ingegneria Informatica S.p.A – Roma, IT), and Hyrum K. Wright (Duolingo – Pittsburgh, US)

When you are developing a project, be it some internal project or some open source project, one key element is how easy it is to onboard new users to your project. For open source projects it is essential to attract more contributors and have a lively community. For internal projects, it is useful to be able to have new employees or newcomers in general be able to get up to speed rapidly on your project.
This brainstorming session was about ideas of tools and practices to use to ease onboarding. Here is the list of ideas we had (in no specific order):

- Tag issues in your issue tracker as onboarding issues to make it easy for newcomers to get started on something easy and be in success quickly. This also validates that they're able to use your software.
- Have a complete package of your software that can be installed and used as easily as possible. It should just work out of the box without having to perform any configuration or additional steps. A good strategy for applications is to provide a Docker image (or a Virtual Machine) with everything set up.
- Similarly, provide a packaged development environment. For example, you can provide a VM with some preinstalled and configured IDE (with plugins installed and configured using the project's rules). One downside of such an approach is the time it takes to download the VM (which could be several GB in size).
- A similar and possibly better approach would be to use an online IDE (e.g., Eclipse Che) to provide a complete pre-built dev environment that would not even require any downloads. This provides the fastest dev experience you can get. The downside is that if you need to onboard a potentially large number of developers, you will need some important infra space/CPU on your server(s) hosting the online IDE, for hosting all the dev workspaces. This makes this option difficult to implement for open source projects for example. But it's viable and interesting in a company environment.
- Obviously having good documentation is a given. However too many projects still don't provide this or only provide good user documentation but not good developer documentation with project practices not being well documented or only a small portion being documented. Specific ideas:
  - Document the code structure
  - Document the practices for development
  - Develop a tool that supports newcomers by letting them know when they follow / don't follow the rules
  - Good documentation shall explicit assumptions (e.g. when you read this piece of documentation, I assume that you know X and Y)
  - Have a good system to contribute to the documentation of the project (e.g. a wiki)
  - Different documentation for users and for developers
- Have homogeneous practices and tools inside a project. This is especially true in a company environment where you may have various projects, each using its own tools and practices, making it harder to move between projects.

- Use standard tools that are well known (e.g., Maven or Docker). That increases the likelihood that a newcomer would already know the tool and be able to develop for your project.
- It's good to have documentation about best practices but it's even better if the important "must" rules be enforced automatically by a checking tool (can be part of the build for example, or part of your IDE setup). For example instead of saying "this @Unstable annotation should be removed after one development cycle", you could write a Maven Enforcer rule (or a Checkstyle rule, or a Spoon rule) to break the build if it happens, with a message explaining the reason and what is to be done. Usually humans may prefer to have a tool telling them that they haven't been following the best practices documented at such location.
- Have a bot to help you discover documentation pages about a topic. For example, by having a chat bot located in the project's chat, that when asked about will give you the link to it.
- Projects must have a medium to ask questions and get fast answers (such as a chat tool). Forum or mailing lists are good but less interesting when onboarding when the newcomer has a lot of questions in the initial phase and requires a conversation.
- Have an answer strategy so that when someone asks a question, the doc is updated (new FAQ entry for example) so that the next person who comes can find the answer or be given the link to the doc.
- Mentoring (human aspect of onboarding): have a dedicated colleague to whom you're not afraid to ask questions and who is a referent to you.
- Supporting a variety of platforms for your software will make it simpler for newcomers to contribute to your project.
- Split your projects into smaller parts. While it's hard and a daunting experience to contribute to the core code of a project, if this project has a core as small as possible and the rest is made of plugins/extensions then it becomes simpler to start contributing to those extensions first.
- Have some interactive tutorial to learn about your software or about its development. A good example of nice tutorial can be found at https://www.katacoda.com/ (for example for Docker, https://www.katacoda.com/courses/docker).
- Human aspect: have an environment that makes you feel welcome. Work and discuss how to best answer Pull Requests, how to communicate when someone joins the project, etc. Think of the newcomer as you would of a child: somebody who will occasionally stumble and need encouragement. Try to have as much empathy as possible.
- Make sure that people asking questions always get an answer quickly, perhaps by establishing a role on the team to ensure answers are provided.
- Last but not least, an interesting thought experiment to verify that you have some good onboarding processes: imagine that 1000 developers join your project / company on the same day. How do you handle this?

If you own a project, we would be interested to hear about your ideas and how you perform onboarding. You could also use the list above as a way to measure your level of onboarding for your project and find out how you could improve it further.

4.5 DevOps and the Need for Improved Evidence from Industrial Studies

Lucy Ellen Lwakatare (University of Oulu, FI) and
Sigrid Eldh (Ericsson AB – Stockholm, SE)

4.5.1 Challenge

One challenge, particularly when considering DevOps transformation in organizations, is the limited availability of supporting evidence of its implementation and quantifiable value in companies. In academic forums, anecdotal evidence is often presented in form of experience reports from leading innovative companies like Facebook and Netflix. For other software intensive organizations, particularly of safety critical, public sector and embedded domain, there exists very limited scientific support of DevOps implementation. This raises the question on whether the different facets of DevOps implementation and the corresponding impacts from pioneering companies are generalizable considering various constraints, e.g., legal and context, that do not necessarily make DevOps transformation easier. Furthermore, it is not clear what measures/indicators are established to clearly assess the value achieved through DevOps transformation. This is important because it is severally reported that DevOps adoption in a software-intensive organization often brings about changes to existing organizational structures, system architecture, e.g., to microservices and software development process, e.g., incorporation of deployment pipeline.

While emerging scientific studies make an initial step to describe the phenomenon, including to define and describe DevOps concept, practices and lessons learned from its early adoption, additional empirical studies are needed to crystallize core contribution of DevOps phenomenon. Practices should (at best) be explained and categorized before and after the DevOps introduction, where particular note should be to evaluate "loss" of practices (1) and "new enabled practices" not previously practiced (2). One should also take into account contextual factors that impact the success or failures.

4.5.2 Our Proposal

Based on this need, we propose studies that focus on identifying

- What are software-intensive companies that have adopted DevOps doing, i.e., *what aspect of DevOps practice is taken into use, in what context*
- What is (1) lost, (2) modified, (3) unchanged, and (4) added in existing practices as a result of DevOps in context. The numbers 1–4 are presented in Figure 3 below.
  - Building evidence for what aspects of practices lost is not making things better – and how to make sure these good practices are sustained
- The added value given by DevOps and to which stakeholder(s)
  - Giving new possibilities that did not exist before
- Measures/indicators used to assess the value, and at what duration/time period as well as whether they can be monetized ($)
- More support for change and transformation – how to address issues in process change "Onboarding"

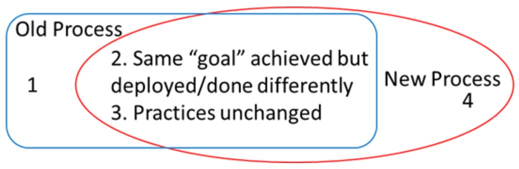


Figure 3 Comparing process practices in DevOps.

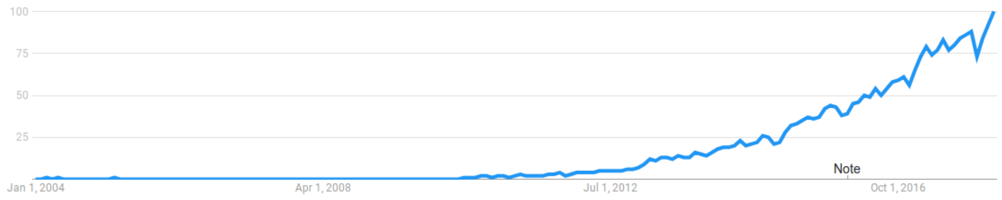


Figure 4 Google Trends for "DevOps". We can see the term "DevOps" pickup traction in 2012.

5 Working Groups – Teaching

Starting with the talk of Christopher Parnin (see Section 3.2) we shifted the focus of our discussion towards teaching DevOps-related concepts. Three breakout groups formed tackling the topic from multiple perspectives involving the infrastructural challenges for hosting such a course, but also what concepts and technologies should actually be part of a curriculum.

5.1 Designing a DevOps Course

Andrew Neitsch (Cisco Systems Canada Co. – Toronto, CA), Georgios Gousios (TU Delft, NL), and Moritz Beller (TU Delft, NL)

5.1.1 Introduction

Teaching a novel topic like DevOps (see Figure 4), which is driven by industry needs, has always been a challenge for academia. It is not like teaching, say, an introductory course in compilers, where the basic theory has been codified in textbooks for decades, and the major difference between tools is a choice of target programming languages.

In this blog post, we discuss whether DevOps courses at university level should focus on teaching the principles or a set of tools, and if so, which principles, and which tools, and to which level of depth?

5.1.2 Challenges in teaching DevOps

The particular issues of infrastructure and operation automation that DevOps tries to solve only arise in fairly large industrial systems that have hit the boundaries of what can be achieved with manual human work. Naturally, most students are alien to this level of scale and therefore lack the understanding for why DevOps practices are needed. One way to make students see value in DevOps, could be to bring in the industrial perspective early-on in a course as a motivating example. Moreover, students usually appreciate guest lectures

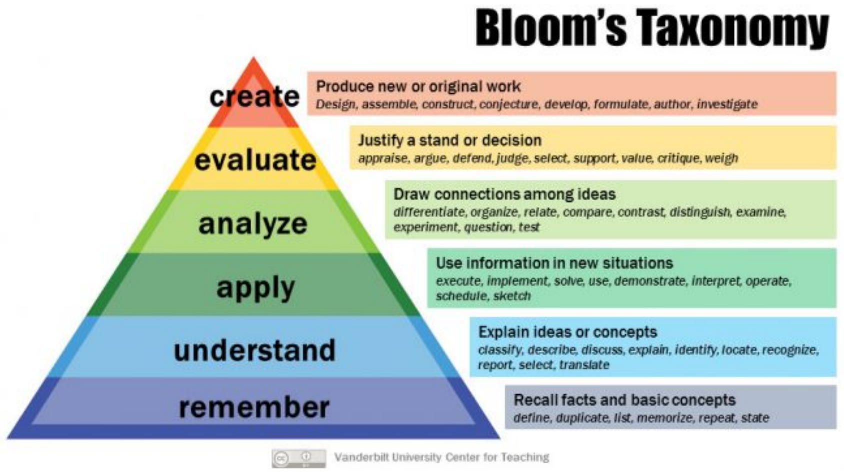

Bloom's Taxonomy
create
Produce new or original work
Design, assemble, construct, conjecture, develop, formulate, author, investigate
evaluate
Justify a stand or decision
appraise, argue, defend, judge, select, support, value, critique, weigh
analyze
Draw connections among ideas
differentiate, organize, relate, compare, contrast, distinguish, examine, experiment, question, test
apply
Use information in new situations
execute, implement, solve, use, demonstrate, interpret, operate, schedule, sketch
understand
Explain ideas or concepts
classify, describe, discuss, explain, identify, locate, recognize, report, select, translate
remember
Recall facts and basic concepts
define, duplicate, list, memorize, repeat, state
Vanderbilt University Center for Teaching

Figure 5 Bloom's Taxonomy of Learning Domains.

from industry. A firsthand story about a problem in the field, such as a website going down without anyone noticing until a major customer complained, can provide motivation for why these topics are important.

Another related challenge is that, while we seem to converge on a mutually shared understanding of DevOps, its boundaries are not yet clearly defined. As an evolving concept, it is not clear which material a good course on DevOps should cover. For example, to what extent should it cover testing on the Continuous Integration server? This is a choice left to the individual teacher and the surrounding courses offered at that university. Due to DevOps's contemporary nature, by the time we cover material in a course, it might be outdated.

Lastly, university courses usually aim at teaching principles rather than tool specifics. However, in DevOps, the distinction between what is tool-specific and what is general enough to serve as a principle is somewhat blurry. In some sense, the principles behind DevOps are easy to understand, but exactly their implementation in practice is hard. A large part of the complexity in DevOps stems from making specific tools interact successfully in a pipeline.

5.1.3 Suggested learning objectives for developing new courses in DevOps

Courses are usually described in terms of learning objectives. What should students know after taking the course? There is not enough time to teach everything to a point of mastery, but Bloom's taxonomy[5] in Figure 5 is useful to describe to what extent the objectives should be learned. It is a continuum from basic to very advanced knowledge.

A good course on DevOps should prepare students with both the theory and the practice of continuous delivery pipelines and automated infrastructures. In that sense, it needs to emphasize both topics related to automating software engineering tasks and topics that have to do with automating the infrastructure. The description of our course relates to the DevOps pipeline view in Figure 6.

In our discussions, we came up with the following topics that we believe are essential for students to understand. Initially, all students need to be up to speed with modern *version control*; even though this is usually taught at basic software engineering courses in most

[5] https://cft.vanderbilt.edu/guides-sub-pages/blooms-taxonomy/

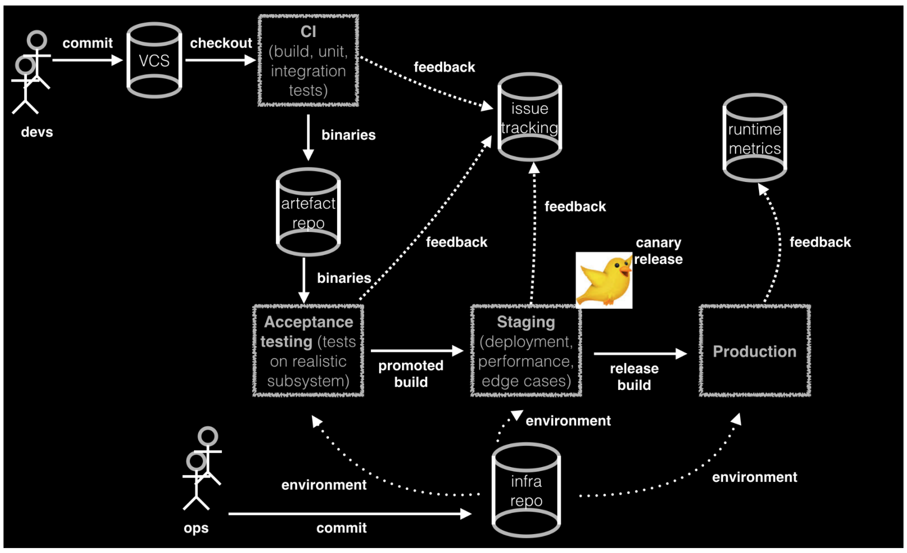


Figure 6 DevOps Workflow (source: Georgios Gousios).

modern programs, its significance in DevOps pipelines (everything starts with a commit!) makes it an important topic to revisit. Then, students also need to come up to speed with practices pertaining to *automated testing*, especially in the context of using testing for quality assurance.

Automated testing can come in multiple flavors (e.g., unit testing, acceptance testing etc) that are being used in various stages in a DevOps pipeline; the students should learn to analyze what testing results mean and what their effects are on the actual product. *Continuous integration* is usually the process that drives the whole pipeline; consequently the students must know how to apply appropriate tools, how to combine them with version control tools (that act as the continuous delivery trigger), how to apply appropriate static analysis tools, and how to store build artifacts (one important concept is immutability).

Automated deployment is the process of applying a build artifact in a production or testing environment; there is a big variation in the tools that can be used for this purpose and the process is also context specific, so we believe that students should learn to apply basic tools (e.g., scripting languages) before trying for specialized ones. The industry does seem to be on a path to convergence on using Docker and Kubernetes in the future, but traditional application deployments will still be around for a long time.

The above would make for good knowledge on continuous integration pipelines; what is missing is how to automate infrastructure to deploy on. For this, the students need a different set of, mostly practical, skills that usually is not part of computer science curricula. This means that a DevOps course needs to emphasize practical system administration and also how to automate it. Again, a plethora of tools, each with a different philosophy behind it, makes it difficult to extract and teach generic principles; thus, the students need to learn to apply one of them (e.g., Ansible or Puppet) in practice.

Production monitoring is to ensure that, once deployed, a service stays up, and what to do when it does not. There are various third-party services that can email or text someone when something goes down. More detailed metrics gathering can be used for capacity planning, and can provide useful business metrics such as how many customers there are and how they are using the application.

5.1.4 Recommendations

In the following, we identified six distinct concepts that a DevOps course should cover. We annotate each concept with the depth of knowledge level that we would expect to be taught at a university-level course. We think that in principle the concepts covered in a DevOps course can be understood by Bachelor-level students. However, practical constraints might make it feasible to only teach a DevOps course at the Master level, even though some concepts such as CI might be covered earlier in the Bachelor. In the last column we give an example of tools that should be covered for the given concept.

Table 2 Concepts of a DevOps course mapped to Bloom Knowledge Levels.

Concept	Bloom Knowledge Level	Example Tools
Version Control	Apply	Git, GitHub
Automated tests: unit tests, integration testing, acceptance testing, ...	Analyze	xUnit, Selenium
Continuous Integration	Apply	Travis CI, Jenkins
Automated Deployments	Apply	Shell scripts, Docker, Kubernetes
Automated Infrastructure	Apply	Chef, Puppet, Ansible, SaltStack
Production Monitoring	Evaluate	Pingdom, Grafana, ELK Stack

5.2 What are the Infrastructural Challenges in Hosting a DevOps Course?

Weiyi (Ian) Shang (Concordia University – Montreal, CA), Shane McIntosh (McGill – Montreal, CA), Christopher J. Parnin (North Carolina State University – Raleigh, USA), Sarah Nadi (University of Alberta – Edmonton, CA), Benjamin Danglot (INRIA Lille, FR), Zhen Ming (Jack) Jiang (York University – Toronto, CA), Lucy Ellen Lwakatare (University of Oulo, FI), Hui Song (SINTEF ICT – Oslo, NO), and Oscar Luis Vera Pérez (INRIA Rennes, FR)

Due to the wide adoption of DevOps in practice, there is a clear need to introduce such DevOps practices into the higher education curriculum, as a course, or even throughout the educational program. While clearly there is benefit in offering such knowledge in the school, the course(s) comes with a cost. The question that we would like to touch on in this blog post is simply, what are the challenges that we face, when offering a DevOps course.

5.2.1 Context of the DevOps course

DevOps is a very practical discipline which has received a lot of attention from industry and academia. The designed course will consists of lectures, hands-on labs, and projects. Hence, in order to provide practical hands-on experience with the students, we intend to provide accessible infrastructures representing state-of-the-art practices for students to experiment with.

5.2.2 Hardware (virtual hardware) support

Do-it-yourself or using a service?

Free (sort of) services and provisioning tools are available to support in each phase of the pipeline. For example, Docker and Vagrant can help in provision a service of particular phases in the DevOps pipeline. Travis CI or CircleCI can be used as readily available services for the build and testing phase of the pipeline, while Jenkins servers can be setup to assist in the build and testing phase as well. Finally, Heroku can be used as a deployment service. There is a decision to make on whether a group of student should take off-the-shelf services to accomplish each phase or pick a readily available service to help. In general, the decisions can be made by considering two aspects:

1. The focus of the course project. While the service facilitates having the pipeline ready, the students may miss the opportunities to exercise or learn what is under the hood. For example, using Travis may miss the chance to learn how to provision and install a build/test server (e.g., Jenkins). If such exercise is considered important by the instructor of the course, those readily available services may be avoided.
2. The ability on special needs. A readily available service may on one hand accomplish many tasks hassle free, while it also may lack the ability to be customized. For example, an off-the-shelf hosting solution like Heroku may not have the flexibility to implement complex deployment strategies (e.g., feature toggle consolidation, blue/green deployment).

The challenge does not stop here. Even without a readily available service, there is still a level of detail you want the student to experience. For example, a student may want to use a Docker image to host a Jenkins server, which may let the student miss the chance of experiencing deploying a Jenkins server from scratch.

In-house or on the Cloud?

If the students are required to provision their own service to support a phase in the pipeline, they need to infrastructure support to host such services. Here are the availabilities:

1. *Public provisioning providers:* AWS, Azure, Digital Ocean, etc. You name it. The list goes on. But using these providers is not free. You will be noticed if your student receive a thousand dollar bill. In fact, we know that there are educational funding supports from some providers (e.g., AWS and Azure). However, it should not be considered as a long term plan. Another issue with leveraging public provisioning providers is the issue of privacy. The course material may be considered as an intellectual property of the instructor and/or the university. You may risk the chance of breaking the rule if the material is shared or even stored by public provisioning providers.
2. *Institutional and regional provider:* Universities, or governments often host their own infrastructure that can be used to host services for educational needs. Since those providers may not be optimized for the course context, local services may be needed to support the pipeline. For example, a local maven repository within the provider's local network may need to be created.
3. *Students' local machines:* The last resort of these infrastructure is to provision on student's machines, e.g., their laptops by creating VMs or Docker images. An apparent issue of such an option is the ability of host multiple VMs or images on their laptops. More importantly, the solution seems to be easier to try out but may be different to running with providers, which is more realistic in practice. Moreover, the student do not get the chance to exercise the operation of the system in the field.

5.2.3 Material (or process) support

How can students be graded?

First off, the form of examinations are not preferred for such a practical course. The students should be graded based on their project of engineering a DevOps pipeline. Here are the options that are available:

- Screencast,
- Coverage graphs,
- Analysis results,
- (or any empirical evidence that they have done the work)

However, grading such materials can be strongly subjectively biased, and on the other hand, resource costing.

How to include development history through the pipeline during the course?

Although the focus on the course is the DevOps pipeline, there is no pipeline useful without actual development on the subject system code. Requesting students to make development to the actual subject system code becomes a simple but naive solution since 1) the student may already be overwhelmed by the pipeline itself, and 2) more importantly, it can bring confusion about what is the real focus of the course. Two possible directions are clear at this stage:

Fake the development: Since the development is not the real focus of the course, it does not seem to be of much value to do a real development on the side. With this spirit, simulated development, (as simple as randomly change a line of code), can be done to exercise the pipeline.

"Outsourcing" the development: No we do not mean to hire people to develop a subject project on the side. In reality, students often need to through team projects in various courses, such as software design, software process, etc. With those courses typically including a development project, the DevOps course can be coordinated with those courses to provide the pipeline of those projects, while using the development of those projects to exercise the pipeline. In such cases, coordination issues becomes the showstopper. Here are some (while we believe more exist):

- Not every professor wants to include DevOps.
- Not every student has the same prerequisites or having pair classes
- The different courses needs to sync their teaching schedule.
- (the list goes on)

5.2.4 Summary

To conclude, hosting a DevOps course is not a trivial task. Due to the nature of the subject, it comes with both technological and non-technological challenges. On the one hand, we encourage people start considering such a subject in their courses, while proper solutions to the above mentioned challenges may be considered to deliver a successful course.

5.3 Teaching Topics Requirements

Bram Adams (Polytechnique Montreal, CA), Tamara Dumic (Ericsson Research – Stockholm, SE), Foutse Khomh (Polytechnique Montreal, CA), and Andy Zaidman (TU Delft, NL)

Three challenges that evolved from our working group about typical requirements for setting up and conducting a DevOps courses will be discussed in the following. We conclude with a brief discussion on topics to consider for a DevOps course and the final question covers potential ways to assess students in such a course.

5.3.1 Challenge 1: DevOps has a strong dependency on other courses

We feel that DevOps is a cross-cutting topic. There are elements of software testing in it, but also relations to software architecture, software design, software requirements and software process.

For example, a microservice architecture makes it easy to frequently redeploy specific services, without having to take the entire system offline. But this requires advance thinking at the level of the software architecture. Also the design level can be important, as at this level the observability of the state of the system can be influenced, which in turn influences how well you can test your system.

Similarly, the requirements can stipulate that rapid releasing is necessary, or that advanced monitoring of the deployed software is necessary. Both can be traced back to DevOps principles. So this stage can also influence the decision making with regards to DevOps.

Your software process (e.g., waterfall, agile, ...) is also an influencing factor on whether and how you want to install your DevOps pipeline.

Our idea is that it would be good that already in these courses this link is highlighted, but vice versa, if a separate course on release engineering is set up, this course should also reflect on the importance of these other subfields, as they are influencing the scope of DevOps to a large degree.

Also, many software engineering programs have project courses in which students develop and release software, such courses could also serve as a vehicle to distill DevOps practices to student throughout their education.

5.3.2 Challenge 2: Raise the awareness of students of (1) the impact their commits have on the whole project and (2) the scale of the release engineering process

Students typically think that writing code and taking care of merge conflicts when pushing to version control is all that needs to be done. However, the reality is that students have a hard time imagining what the potential impact is of making changes to their component when they are working in a much larger project that has many build dependencies. By installing a DevOps pipeline for their school projects their insight will increase, but even then this does not compare to some massive and complex build processes that can be seen in large companies.

We find that this is an argument to also involve guest speakers from industry, as they are likely able to show the scale issues that arise.

5.3.3 Challenge 3: How to keep the course relevant amidst rapid changes in concepts and technologies

As the field of DevOps is still emerging and people are finding their way in the area, it is still not entirely clear which set of technologies will emerge as "standard". While it seems that Ansible and Docker are gaining traction here, it might be that new technologies or rival technologies will still emerge. Early experiences are also that existing tools are rapidly evolving still, thus requiring frequent updates to course material.

An important question that we have asked ourselves during the discussion is whether we should focus our teaching on technology or on principles. Or whether the principles can be taught through technology. We feel that the technology should be taught and that this should be enough for most students to get the necessary awareness and insights into the general principles.

5.3.4 Topics to cover in a DevOps course

Given the cross-cutting nature of DevOps activities, there is a wide range of skills that students need to acquire to able to contribute efficiently to a software release. Hence, as mentioned earlier it is important that courses that cover key activities of the software development and release process includes aspects of DevOps. However, setting up and running an efficient release pipeline requires specific knowledge that should be covered in a DevOps course. Students in a DevOps course should be taught configuration management with an artifact repository, i.e., how to manage dependencies, traceability between software products and requirements, etc. Another important topic is dynamic scheduling/provisioning of test environments across different levels ("what hardware and software do I need without blocking any limited resource unnecessarily?"). We should also teach students to build and provision efficient release pipelines (i.e., ensuring that their pipelines are capable of catching issues efficiently without making an excessive use of the limited resources available) and to ensure the security of these pipelines to prevent malwares from slipping into released products. Other interesting topics to cover are available in the summary of the 2016 workshop on DevOps education[6]. The Linux Foundation also provides interesting pointers on topics to cover in a DevOps course.

5.3.5 How to assess students in a DevOps course?

To assess the knowledge acquired by students we propose the following mechanisms:

- Cross-evaluation of assignments/projects
- Make students write some kind of reflection about the DevOps experience of the class (a 2 page paper)
- Make the students do screencasts to explain their work
- Write a personal diary of an operator after a hard day at work

Pre-requisite skills include networking, virtualization, and troubleshooting skills. Further, different kinds of testing could be involved and assessed: unit, functional, non-functional testing, staging, etc. Code could be assessed as well, e.g., clean code: design patterns, principles, best practices. Further, architectural styles (e.g., microservices) applied, aggregated pipelines, and quality assurance in general.

[6] https://drive.google.com/open?id=1LwNLmF252uFtQPp6Q6QEmP0am-OBul_n

- Requirement courses, architecture courses, quality assurance course should cover “Ops” issues
- Teaching DevOps principle, environments, release pipelines
- Hands on experience with some specific tools for release through a project → it is better for students to have experience with some specific tools than no experience at all and only broad knowledge of concepts
- Huge application: system-level tests are too complicated for individual developers (too many interactions) → how to obtain feedback that is concrete enough for developers to know how to fix the issue?
- Industry/practice: missing knowledge on different levels of testing, link with clean code, mock testing, 10s of pipelines: follow your commit until certain level

Not necessarily directly linked to assessing students, but teaching support is also an important aspect to take into account. Thus, how to train teaching assistants for such a DevOps course? One idea would be to ask the best students of the previous year to TA the course. Or, to limit the first edition to a few students and manage the course and then in the subsequent editions of the course, pick the best students of the previous edition to TA.

6 Panel Discussions

After the talks from Sigrid (see Section 3.3) and Hyrum (see Section 3.4), we discussed two main topics, namely the architecture(s) that enable DevOps (Section 6.1) and what to do with the data collected in the Ops phase (Section 6.2). Finally, we concluded the seminar with discussing potential next steps.

6.1 Microservices for DevOps

All Participants

The architecture discussion that followed the talks went pretty quickly towards microservices. While there was consensus that a microservice architecture is not the only possible architecture for enabling DevOps, it seems that microservices are associated with the concept of DevOps quite frequently. The fact that these services are relatively small and loosely coupled, make it easy for them to get re-deployed while the system is running. Nevertheless, also more monolithic architectures can be build and deployed in a DevOps fashion, just consider mobile apps for example that might also send in post-deployment data.

Another aspect of the discussion is then how we test this microservice architecture. Each microservice can be tested at design time, but how the microservices interact can only be tested at runtime through some form of online testing, thus collecting data on how these microservices interact becomes all the more important. Related to the previous point, the discussion also went into the direction that some companies actually need to reverse engineer the interactions between the microservices to get an understanding of the actual architecture of the microservices system. So some of the business logic is in how the services interact, and not only in the microservices themselves anymore. This again raises the importance of online testing.

6.2 Dealing with Large Amounts of Data

All Participants

DevOps processes generate huge amounts of data, which need to be acted upon in order to feed back operations insights to development teams (in the broad sense). Needless to say, those insights can be pretty well hidden within heaps of noise.

For example, the popular TravisTorrent data set providing 60 attributes of 3.7 million Travis CI build jobs is based on a raw data set of over 2 TB of build logs (https://blog.travis-ci.com/2017-01-16-travis-ci-mining-challenge/). To make analysis of this data feasible for researchers, the logs were aggregated into 3.2 GB of SQL dumps, focusing on the most commonly requested attributes of the builds.

In industry, the scale of things is even more extreme due to the wide variety of telemetry data gathered for each transaction and event. In 2016, Netflix' Keystone data pipeline[7] had to process 500 billion events (1.3 petabyte of data!) per day. At peak times, 8 million events (24 GB of data) are generated per second(!). In 2017, Twitter's event logs[8] processed more than a trillion of events per day, amounting to 3 PB of uncompressed data per day.

These numbers are increasing each year, forcing companies to also scale up and improve their data pipeline. In December 2015, Netflix' Keystone pipeline was already the third incarnation of the company's data pipeline. Apart from being able to process such data in real-time, companies need to consider the need for long-term storage of this data, which adds additional nightmares. Legislation like the Sarbanes-Oxley Act[9] requires certain kinds of data to be retained for a long time, with certain requirements regarding confidentiality and privacy.

Given these facts, this grand challenge focuses on the following major questions:

- How to identify the types of operations data that are useful in the context of DevOps?
- How to efficiently collect the identified data?
- How to efficiently process the identified data in order to extract actionable insights for "Dev"?
- How to back the data up in an efficient way, safeguarding privacy and confidentiality?
- How to give data scientists access to portions of the data for advanced analysis, without exposing too many privacy details?
- How to succinctly report the data back to "Dev"?

[7] https://medium.com/netflix-techblog/evolution-of-the-netflix-data-pipeline-da246ca36905
[8] https://events.static.linuxfound.org/sites/events/files/slides/Routing%20Trillion%20Events%20per%20day%20%40Twitter.pdf
[9] https://en.wikipedia.org/wiki/Sarbanes%E2%80%93Oxley_Act

6.3 Next Steps

All Participants

One idea raised was to launch a follow-up event of the workshop on Release Engineering at ICSE'19. We briefly discussed similar events that are upcoming or took place recently:

- DevOps 2018: Bertrand Meyer organized the "First international workshop on software engineering aspects of continuous development and new paradigms of software production and deployment"[10]
- Philipp Leitner was involved in organizing the "Vienna Software Seminar"[11] covering Continuous Delivery from an architecture perspective.

A potential reboot of the ICSE workshop could focus on technical challenges. Furthermore, it should be no problem if the workshop occurs only once, twice or three times. "Let us not fear to stop a workshop series." Another idea raised was to take a look at industry conferences such as DEVOXX[12], Velocity[13], or FOSDEM[14]. A Dagstuhl representative also mentioned that Dagstuhl can also host summer schools.

We also discussed potential goals of such a follow-up event:

- Education
- Community building
- Involve academia, software industry who contribute to DevOps, software industry who want to adopt DevOps, clients who see their software suppliers adopt DevOps
- Crystalize the core research problems in DevOps
- Awareness of the human factors

A further item of our discussion was on how we can reach out to "Ops" people:

- Who are they? What do they study? What conferences do they attend? (e.g., Velocity, AWS conference)
- They build tools that enable developers to do their job better (DB, systems, networks)
- From an academic perspective, they are in the "network and system" department
- They are people who like to see a system running
- LISA[15] is a Usenix conference where Ops people go
- Related to DB, systems, resource management

One challenge that we did not touch upon is how to adopt DevOps for embedded systems. The key issue there is that the gap between Dev and Ops is very difficult to bridge. That is mainly because the deployment on the embedded platforms is very specific, which makes it hard to automate.

[10] https://www.laser-foundation.org/devops/2018/
[11] https://vss.swa.univie.ac.at/2017/
[12] https://devoxx.com/
[13] https://conferences.oreilly.com/velocity
[14] https://fosdem.org/2018/
[15] https://www.usenix.org/conference/lisa18

Participants

- Bram Adams
Polytechnique Montreal, CA
- Benoit Baudry
KTH Royal Institute of Technology – Stockholm, SE
- Moritz Beller
TU Delft, NL
- Benjamin Danglot
INRIA Lille, FR
- Tamara Dumic
Ericsson Research – Stockholm, SE
- Sigrid Eldh
Ericsson AB – Stockholm, SE
- Daniele Gagliardi
Engineering Ingegneria Informatica S.p.A – Roma, IT
- Georgios Gousios
TU Delft, NL
- Zhen Ming (Jack) Jiang
York University – Toronto, CA
- Foutse Khomh
Polytechnique Montreal, CA
- Philipp Leitner
Chalmers University of Technology – Göteborg, SE
- Lucy Ellen Lwakatare
University of Oulu, FI
- Vincent Massol
XWIKI – Paris, FR
- Shane McIntosh
McGill University – Montreal, CA
- Martin Monperrus
KTH Royal Institute of Technology – Stockholm, SE
- Sarah Nadi
University of Alberta – Edmonton, CA
- Andrew Neitsch
Cisco Systems Canada Co. – Toronto, CA
- Christopher J. Parnin
North Carolina State University – Raleigh, US
- Gerald Schermann
Universität Zürich, CH
- Weiyi (Ian) Shang
Concordia University – Montreal, CA
- Hui Song
SINTEF ICT – Oslo, NO
- Oscar Luis Vera Perez
INRIA – Rennes, FR
- Hyrum K. Wright
Duolingo – Pittsburgh, US
- Andy Zaidman
TU Delft, NL
- Fiorella Zampetti
University of Sannio – Benevento, IT