37. Workshop über Komplexitätstheorie, Datenstrukturen und effiziente Algorithmen

Dienstag, den 23. März 1999,

Hörsaal III.101, Institut für Informatik, HU-Berlin

Humboldt-Universität zu Berlin
Institut für Informatik
Lehrstuhl Algorithmen und Komplexität

Clemens Gröpl, Stefan Hougardy, Till Nierhoff, Ralf Oelschlägel,
Deryk Osthus, Hans Jürgen Prömel, Mark Proksch, Anusch Taraz

Programm

ab 9:00	**Kaffee**
09:25	Begrüßung
09:30	*Frank Reffel (Karlsruhe)*
	Cyclic BDDs
09:55	*Detlef Sieling (Dortmund)*
	Die Komplexität der FBDD-Minimierung
10:20	*Jörg Rothe (Jena)*
	Creating Strong, Total, Commutative, Associative One-Way Functions from Any One-Way Function in Complexity Theory
10:45	**Pause (15 Minuten)**
11:00	*Hanno Lefmann (Dortmund)*
	On Heilbronn's Problem in Higher Dimensions
11:25	*Helmut Alt, Christian Knauer (FU Berlin)*
	Matching shapes wrt. the symmetric difference
11:50	*Friedhelm Meyer auf der Heide (Paderborn), Berthold Vöcking (Berkeley), Matthias Westermann (Paderborn)*
	Data Management in Networks
12:15	**Mittagspause (1 Stunde 15 Minuten)**
13:30	*Johannes Köbler, Wolfgang Lindner, Rainer Schuler (Ulm)*
	Derandomisierung von RP und Erlernbarkeit von Booleschen Schaltkreisen
13:55	*Sven Kosub, Klaus W. Wagner (Würzburg)*
	Die Boolesche Hierarchie der NP-Partitionen
14:20	*Stefan Schroedl, Stefan Edelkamp (Freiburg)*
	Loops and Branches in End User Programming
14:45	**Pause (15 Minuten)**
15:00	*Bernd Borchert, Frank Stephan (Heidelberg)*
	Limits of alternating 4-way automata
15:25	*Andreas Jakoby, Rüdiger Reischuk (Lübeck)*
	Average Case Complexity of Unbounded Fanin Circuits
15:50	**Pause (15 Minuten)**
16:05	*Stefan Droste, Thomas Jansen (Dortmund)*
	Analyse eines einfachen, adaptiven evolutionären Algorithmus
16:30	*Petra Berenbrink (Paderborn), Tom Friedetzky, Angelika Steger (TU München)*
	Randomized and Adversarial Load Balancing
16:55	**Ende des Workshops**

Cyclic BDDs

Frank Reffel[1]
Institut für Logik, Komplexität und Deduktionssyteme
Universität Karlsruhe
reffel@ira.uka.de

BDDs [1] are the state-of-the-art representation for boolean functions and are widely used in model checking [2] and hardware verification. They helped to overcome memory problems of state enumeration techniques but cannot always avoid the typical exponential blow-up. Most of the systems have a modular architecture and for several of them all modules have the same structure. In this case, regarding the structure of the *BDD* representing the reachable states and the transition relation, many structural identities are observed. Classical *BDDs* are not able to share these structures because they occur in different layers of the *BDD*.

A new *BDD*-variation is presented which can benefit from such structural identities. The *BDD* is divided into segments corresponding to the modules of the system. Special nodes with several exits are introduced between the segments of the *BDD*. As this allows a multiple use of *BDD*-parts cycles can be introduced into *CBDDs*. One node can represent different functions dependent on the layer for which it is encountered. This is transparent for the user of the *BDD*-library and the efficient classical *BDD*-operations are not affected. An additional hash-table for BDD segments is introduced which depends on the structure of the *BDD*-parts.

This new *BDD*-variation does not preserve the uniqueness of *BDDs* but can still guarantee that equal boolean functions are mapped to unique *CBDDs*. Therefore the important property which is responsible for the efficiency of *BDDs* is preserved and the test for equality of *CBDDs* can still be done in constant time by a comparison of the root nodes. Some initial experiments were performed with a prototypical implementation. The tree-arbiter and Milners scheduler showed that the described structural identities occur in such systems and can be exploited by *CBDDs*. Furthermore there are examples of scalable functions whose *BDDs* grow linearly while it exists a *CBDD* with a constant number of nodes and only the size of the special nodes increases.

Literatur

[1] R. Bryant, *Graph based algorithms for boolean function manipulation*, IEEE Transactions on Computers (C-35(8), 1986), pages 677-691.

[2] K.L. McMillan, *Symbolic Model Checking*, Kluwer Academic Press, 1993.

[1] supported by DFG within graduate program on controllability of complex systems

Die Komplexität der FBDD-Minimierung

Detlef Sieling[2]
Universität Dortmund, FB Informatik,
LS 2, D-44221 Dortmund
sieling@ls2.cs.uni-dortmund.de

In vielen Anwendungen im Bereich VLSI CAD werden Datenstrukturen für Boolesche Funktionen benötigt, die einerseits die kompakte Darstellung von möglichst vielen Funktionen und andererseits die effiziente Manipulation der dargestellten Funktionen ermöglichen. Neben OBDDs (Ordered Binary Decision Diagrams) wurde die Verwendbarkeit von vielen weiteren Varianten von BDDs als Datenstruktur für Boolesche Funktionen untersucht. Eine diese Varianten sind die FBDDs (Free Binary Decision Diagrams), die in der Komplexitätstheorie auch Read-once Branchingprogramme genannt werden. Es ist bekannt, daß für FBDDs viele, wenn auch nicht alle wichtigen Operationen auf Booleschen Funktionen effizient ausgeführt werden können, wenn nur FBDDs bezüglich einer festgelegten Graphordnung benutzt werden. Dies ist ähnlich zu den OBDDs, wo wichtige Operationen auch nur dann effizient ausgeführt werden können, wenn eine Variablenordnung fixiert ist. Graphordnungen sind eine Verallgemeinerung von Variablenordnungen, da die Variablen für verschiedene Inputs in verschiedenen Reihenfolgen getestet werden dürfen. Ähnlich wie bei den OBDDs kann die Wahl der Graphordnung entscheiden, ob ein FBDD bezüglich dieser Graphordnung polynomielle oder exponentielle Größe hat. Dies motiviert das Problem OptGraphOrdering, zu einem gegebenen FBDD für eine Funktion f eine Graphordnung zu berechnen, die die FBDD-Größe für f minimiert. Ein verwandtes Problem ist MinFBDD, das Problem, zu einem FBDD für eine Funktion f ein FBDD minimaler Größe für f zu berechnen.

Wir zeigen, daß beide Probleme vermutlich keine polynomiellen Approximationsschemata haben: Falls OptGraphOrdering ein polynomielles Approximationsschema hat, folgt ZPP=NP, und falls MinFBDD ein polynomielles Approximationsschema hat, folgt P=NP. Also müssen wir uns in Anwendungen mit suboptimalen Lösungen dieser Probleme zufriedengeben.

[2] Gefördert durch die Deutsche Forschungsgemeinschaft unter Projekt We 1066/8.

Creating Strong, Total, Commutative, Associative One-Way Functions from Any One-Way Function in Complexity Theory

Jörg Rothe
Institut für Informatik
Friedrich-Schiller-Universität Jena
07740 Jena, Germany
rothe@informatik.uni-jena.de

Rabi and Sherman presented novel digital signature and unauthenticated secret-key agreement protocols, developed by themselves and by Rivest and Sherman. These protocols use strong, total, commutative (in the case of multi-party secret-key agreement), associative one-way functions as their key building blocks. Though Rabi and Sherman did prove that associative one-way functions exist if P $\neq$ NP, they left as an open question whether any natural complexity-theoretic assumption is sufficient to ensure the existence of strong, total, commutative, associative one-way functions. In this talk, this question will be answered: If P $\neq$ NP then strong, total, commutative, associative one-way functions exist.

The results presented in this talk are based on joint work with Lane A. Hemaspaandra.

On Heilbronn's Problem in Higher Dimensions

Hanno Lefmann
Lehrstuhl Informatik 2,
Universität Dortmund,
D–44221 Dortmund, Germany
lefmann@Ls2.informatik.uni--dortmund.de

Heilbronn conjectured that given arbitrary n points from the 2–dimensional unit square, there must be three points which determine a triangle of area at most $O(1/n^2)$. This conjecture was disproved by a nonconstructive argument of Komlós, Pintz and Szemerédi [3] who showed that for every n there is a configuration of n points in the unit square where all triangles have area at least $\Omega(\log n/n^2)$.

Using a discretization of Heilbronn's problem combined with derandomization techniques a polynomial time algorithm to achieve this lower bound on the area was given in [2].

Very recently, Barequet [1] considered a generalization of this problem to dimension d for fixed values of $d \geq 3$. Namely, he constructed in polynomial time n points in the d-dimensional unit cube such that the minimum volume of any simplex spanned by $(d+1)$ of these points is at least $\Omega(1/n^d)$. Here we we improve on this lower bound by the logarithmic factor $\Theta(\log n)$. In our arguments we use uncrowded hypergraphs.

Literatur

[1] G. Barequet, *A Lower Bound for Heilbronn's Triangle Problem in d Dimensions.* To appear in: Proc. 10th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'99), 1999.

[2] C. Bertram–Kretzberg, T. Hofmeister and H. Lefmann, *An Algorithm for Heilbronn's Problem,* Proc. Third Intl. Computing and Combinatorics Conference (COCOON'97), eds. T. Jiang and D. L. Lee, LNCS 1276, Springer Berlin, 1997, 23-31.

[3] J. Komlós, J. Pintz and E. Szemerédi, *A Lower Bound for Heilbronn's Problem,* Journal of the London Mathematical Society, 25, 1982, 13–24.

Matching shapes wrt. the symmetric difference

Helmut Alt, Christian Knauer
Institut für Informatik,
Freie Universität Berlin,
Takustraße 9, D–14195 Berlin, Germany
{alt,knauer}@inf.fu-berlin.de

The problem of deciding how 'similar' two geometric shapes are has numerous applications e.g. in GIS, CAD, etc. If the shapes are polygons in the plane, then the *area of their symmetric difference* can be used as a distance measure.

We consider the following problem: Given two sets A and B of pairwise disjoint triangles, find a transformation φ from a given set of *admissible transformations* $\mathcal{T}$, that maximizes the area of overlap between $\varphi(A)$ and B, i.e. minimizes the area of their symmetric difference. Reasonable sets of admissible transformations are, f.i., translations, rigid motions or affine mappings; we will concentrate on translations although the technique is applicable in the other cases as well.

To compute φ we split up the *configuration space* of $\mathcal{T}$ into what we call *blocks*. A block $\mathcal{B}$ is defined in such a way that the combinatorial type of overlap between the shapes remains fixed for all transformations in $\mathcal{B}$.

As it turns out the blocks are formed by the cells of an appropriate arrangement of *algebraic* surfaces (of fixed degree) in configuration space. For translations, for example, the blocks correspond to the faces of a line arrangement. Furthermore the area of overlap between each pair of triangles is a *rational function* in the transformation parameters on $\mathcal{B}$. This implies that the total area of overlap is also a rational function on $\mathcal{B}$. Therefore we can find an optimal transformation in $\mathcal{B}$ by enumerating all maxima on all *faces* of the block. Repeating the process for all blocks and keeping track of the current optimal transformation, we can compute a $\varphi \in \mathcal{T}$ that maximizes the area of overlap between B and the image of A.

We will show that the number of blocks is polynomial for rigid motions and provide the more practical bound of $O(m^2n^2)$ (where $n = \#A$, $m = \#B$) for the case of translations. Furthermore we will show, that $\mathbf{OL}(\Delta_A, \Delta_B)$ is a polynomial for translations and a rational function for rigid motions on each block $\mathcal{B}$, so that local optimization can be done in time proportional to the complexity (i.e. the number of the lower dimensional faces) of the blocks using standard techniques from analysis.

For translations we give a $O(m^2n^2)$ time, $O(mn)$ space algorithm that enumerates the blocks with a topological sweep; in that case the optimization over the faces only requires the solution of linear equations.

Data Management in Networks

Friedhelm Meyer auf der Heide[3] Berthold Vöcking[4] Matthias Westermann[5]

This talk deals with data management for large parallel and distributed systems such as massively parallel processor systems (MPPs) and networks of workstations (NOWs) that consist of a set of nodes each having its own local memory module. These nodes are usually connected by a relatively sparse network such that communication is often the major bottleneck. The only way to bypass this bottleneck is to reduce the communication overhead by exploiting locality. A dynamic data management service allows to access shared data objects from the individual nodes in the network. These objects are, e.g., global variables in a parallel program, pages or cache lines in a virtual shared memory system, shared files in a distributed file system, or pages in the World Wide Web. In this talk, we analyze theoretically and evaluate experimentally a data management strategy called the "access tree strategy" that we have introduced in [1].

The theoretical analysis of the access tree strategy considers data management in a competitive model. It is shown that the access tree strategy minimizes the congestion up to small factors by anticipating the locality included in an application. Thus, the access tree strategy prevents that some of the links become a communication bottleneck. Several classes of networks are considered. For example, it is shown that the access tree strategy achieves competitive factor $O(d \cdot \log n)$ for every application running on d-dimensional meshes with n nodes. This upper bound on the competitive ratio is optimal for meshes of constant dimension. Furthermore, Internet-like clustered networks are investigated.

In the experimental evaluation of the access tree strategy we test several variations of this strategy on two different applications of parallel computing, which are matrix multiplication and Barnes-Hut N-body simulation. We compare the congestion and the execution time of the access tree strategy and their variations with a standard caching strategy that uses a fixed home for each data object. Additionally, we do comparisons with hand-optimized message passing strategies producing minimal communication overhead. At first, we will see that the execution time of the applications heavily depends on the congestion produced by the different data management strategies. At second, we will see that the access tree strategy clearly outperforms the fixed home strategy and comes relatively close to the performance of the hand-optimized strategies. In particular, the larger the network is the more superior the access tree strategy is against the fixed home strategy.

Literatur

[1] B. M. Maggs, F. Meyer auf der Heide, B. Vöcking, and M. Westermann. Exploiting locality for networks of limited bandwidth. In *Proc. of the 38th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 284–293, 1997.

[3] Department of Mathematics and Computer Science, and Heinz Nixdorf Institute University of Paderborn, Germany. Email: fmadh@uni-paderborn.de. Supported in part by DFG-Sonderforschungsbereich 376 and EU ESPRIT Long Term Research Project 20244 (ALCOM-IT).

[4] International Computer Science Institute, Berkeley, CA 94704-1198. Email: voecking@icsi.berkeley.edu. Supported by a DAAD postdoc fellowship. This research was conducted while he was staying at the Heinz Nixdorf Institute, with support provided by DFG-Sonderforschungsbereich 376.

[5] Department of Mathematics and Computer Science, and Heinz Nixdorf Institute University of Paderborn, Germany. Email: marsu@uni-paderborn.de. Supported in part by DFG-Sonderforschungsbereich 376 and EU ESPRIT Long Term Research Project 20244 (ALCOM-IT).

Derandomisierung von RP und Erlernbarkeit von Booleschen Schaltkreisen

Johannes Köbler, Wolfgang Lindner, Rainer Schuler
Universität Ulm

Die effiziente Erlernbarkeit von Booleschen Schaltkreisen ist eine wichtige offene Frage in der Erlernbarkeitstheorie. Es ist bekannt, daß Boolesche Schaltkreise in Valiants PAC Modell nicht erlernbar sind, falls effizient berechenbare Pseudozufallsgeneratoren existieren. Dies gilt auch für den Fall, daß Mitgliedschaftsfragen erlaubt sind und das Lernerfolgskriterium nur bezüglich der uniformen Verteilung gefordert wird [GGM86]. Außerdem hat die Nichterlernbarkeit von Booleschen Schaltkreisen die Ungleichheit von RP und NP zur Konsequenz [PV88].

Wir zeigen, daß jede Sprache $L \in \mathrm{RP}$ für unendlich viele Eingabelängen deterministisch in subexponentieller Zeit approximiert werden kann, falls Boolesche Schaltkreise unter der uniformen Verteilung nicht mit Mitgliedschaftsfragen PAC-erlernbar sind. (Tatsächlich erhalten wir für jedes $\gamma > 0$ eine Polynomialzeit-Approximation für L mit n^γ vielen nichtdeterministichen Entscheidungen und somit eine deutliche Verbesserung gegenüber der bereits erwähnten Konsequenz "RP $\neq$ NP".)

Im Beweis benutzten wir die Konstruktion von Pseudozufallsgeneratoren von Nisan und Wigderson in einer ähnlichen Weise wie in der kürzlich erschienenen Arbeit von Impagliazzo und Wigderson zur Derandomisierung von BPP unter der Annahme EXP $\not\subseteq$ BPP [IW98].

Literatur

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, **33**(4):792–807, 1986.

[IW98] R. Impagliazzo and A. Wigderson. Randomness vs. time: De-randomization under a uniform assumption. In *Proc. 39th IEEE Symposium on the Foundations of Computer Science*, 734–743. IEEE Computer Society Press, 1998.

[PV88] L. Pitt and L. G. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, **35**(4):965–984, 1988.

Die Boolesche Hierarchie der NP-Partitionen

Sven Kosub **Klaus W. Wagner**
Theoretische Informatik, Universität Würzburg
Am Exerzierplatz 3, D-97072 Würzburg
{kosub,wagner}@informatik.uni-wuerzburg.de

Die klassische Komplexitätstheorie untersucht in erster Linie die Komplexität von Mengen, d.h. von Zerlegungen (Partitionen) der Grundmenge in zwei Teile. Mit dieser Arbeit soll ein (erster?) Schritt zum Studium der Komplexität von Zerlegungen der Grundmenge in k Teile getan werden. Dazu wird die Definition der Klassen der booleschen Hierarchie von Mengen verallgemeinert und zu jeder Funktion $f : \{0,1\}^m \rightarrow \{1,2,\ldots,k\}$ eine Klasse $\mathrm{NP}(f)$ von k-Partitionen definiert. Während im klassischen Fall $k = 2$ jede dieser Klassen zusammenfällt mit einer der bekannten Klassen $\mathrm{NP}(n)$ oder $\mathrm{coNP}(n)$, ist die Situation im Falle $k \geq 3$ viel komplizierter. Es wird eine hinreichende Bedingung dafür angegeben, daß für zwei Funktionen $f, g : \{0,1\}^m \rightarrow \{1,2,\ldots,k\}$ die Inklusion $\mathrm{NP}(f) \subseteq \mathrm{NP}(g)$ gilt, und es wird vermutet, daß diese Bedingung auch notwendig ist (unter der Voraussetzung, daß die Poynomialzeithierarchie nicht kollabiert).

Loops and Branches in End User Programming

Stefan Schroedl, Stefan Edelkamp
Institut für Informatik,
Albert-Ludwigs-Universität,
Am Flughafen 17, D-79110 Freiburg
{schroedl,edelkamp}@informatik.uni-freiburg.de

Some decades ago computer programs were designed specifically for the needs of the customer. With the advent of personal computing, however, the profile of "end users" changed: they now have to use a generic application developed by a distant, unknown and unreachable programmer. As a result, end users must map their activities into the capabilities of the generic system. The current approaches to "end user programming" can be lumped into three categories. *Preferences*, *Scripting Languages* and *Macro Recorders*. The proposed approach extends the idea of end user programming through macro recording and thus contributes to the the research area of *Programming by Example* [2, 3].

In the talk we present a supervised, interactive learning technique that infers control structures of computer programs from user-demonstrated traces. We aim to generate computer programs that contain iterative loops and conditional branches from a given set of examples. The core of the approach is based on Angluin's ID-algorithm for regular grammar inference [1].

A two-stage process is applied: first, a minimal deterministic finite automaton (DFA) M labeled by the instructions of the program is learned from a set of example traces and membership queries to the user.

It accepts all prefixes of traces of the target program. The number of queries is bounded by $O(k \cdot |M|)$, with k being the total number of instructions in the initial example traces. In the second step we parse this automaton into a high-level programming language in $O(|M|^2)$ steps, replacing jumps by conditional control structures.

Literatur

[1] D. Angluin, *A Note on the Number of Queries Needed to Identify Regular Languages*, Information and Control, 1981(51):1, pp 76–87.

[2] B. Gaines *Behaviour/Structure Transformations Under Uncertainty*, International Journal of Man-Machine Studies, 1976(8):3, pp 337–365.

[3] Allen Cypher *Watch What I Do: Programming by Demonstration*, MIT Press, 1993.

Limits of alternating 4-way automata

Bernd Borchert, Frank Stephan
Im Neuenheimer Feld 294, 69120 Heidelberg
bb/fstephan@math.uni-heidelberg.de

It is shown that the complement of a language accepted by an alternating 4-way automaton is recognizable (for the definitions see the survey paper on two-dimensional languages by Giammarresi and Restivo in the Handbook of Formal Languages). This allows to apply non-recognizablity results in order to show that certain languages cannot be accepted by alternating 4-way automata.

Average Case Complexity of Unbounded Fanin Circuits

Andreas Jakoby, Rüdiger Reischuk
Institut für Theoretische Informatik
Med. Universität zu Lübeck

Håstad has shown that functions like PARITY cannot be computed by unbounded fanin circuits of small depth and polynomial size. We generalize this result in two directions. First, we obtain the same tight lower bound for the average case. This is done by estimating the average delay – the natural generalization of circuit depth to an average case measure – of unbounded fanin circuits of polynomial size, resp. their error probability given an upper bound on the maximal delay. These bounds are obtained by extending the probabilistic restriction method to an average case setting.

Secondly, we completely classify the set of parallel prefix functions – for which PARITY is just one example – with respect to their average delay in unbounded fanin circuits of a given size. It is shown that only two cases can occur: a parallel prefix functions either has the same complexity as PARITY, that is the average delay has to be of order $\Theta(\log n / \log\log s)$ for circuit of size s, or it can be computed with *constant* average delay and almost linear size – there is nothing in between. This classification is achieved by analyzing the algebraic structure of the semigroups that correspond to parallel prefix functions.

Analyse eines einfachen, adaptiven evolutionären Algorithmus

Stefan Droste   Thomas Jansen
Fachbereich Informatik, Lehrstuhl 2,
Universität Dortmund,44221 Dortmund
`[droste, jansen]@ls2.cs.uni-dortmund.de`

Evolutionäre Algorithmen (EA) sind naturanaloge Verfahren, die unter anderem zur Funktionsoptimierung für Zielfunktionen $f : \{0,1\}^n \to \mathcal{R}$ eingesetzt werden können. Im Vergleich mit den reichhaltigen praktischen Erfahrungen im Umgang mit EA nehmen sich die theoretisch gesicherten Erkenntnisse bescheiden aus. Ein Ansatz für theoretische Untersuchungen ist die Analyse einfachster evolutionärer Algorithmen.

```
1. $x :=$ Initialisierung(); $t := 1$
2. $y :=$ Mutation($x, t$)
3. $x :=$ Selektion($x, y, t$)
4. $t := t{+}1$; Weiter bei 3.
```

Die durch die Verwendung echter Populationen und Crossover entstehenden Probleme werden ebenso wie die Wahl eines geeigneten Stopkriteriums ausgeblendet.

Viele EA führen in einem Lauf alle Schritte immer gleich aus. EA zeigen jedoch häufig ein besseres Verhalten, wenn während eines Laufes wichtige Parameter wie die Mutationswahrscheinlichkeit oder die Wahrscheinlichkeit des Akzeptierens von Verschlechterungen variiert werden. Wir nähern uns diesem Phänomen, indem wir für zwei einfache Ausprägungen des allgemeinen, einfachen EA einen exponentiellen Vorteil für sogenannte adaptive Parametrisierungen nachweisen. Wir betrachten folgende konkrete Realisierung des einfachen EA.

Initialisierung():	Wähle x zufällig gleichverteilt aus $\{0,1\}^n$.
Mutation(x, t):	Setze $y := x$. Wähle zufällig gleichverteilt $i \in \{1, \ldots, n\}$ und invertiere das i-te Bit in y.
Selektion(x, y, t):	Setze $x := y$ mit Wahrscheinlichkeit $\min\{1, \alpha(t)^{f(y)-f(x)}\}$.

Dabei analysieren wir die erwartete Zeit zum Erreichen eines globalen Maximums einmal mit konstanter und einmal mit variabler Funktion $\alpha : \mathcal{N} \to [1; \infty[$ für geeignet konstruierte Zielfunktionen.

Randomized and Adversarial Load Balancing

Petra Berenbrink[6] **Tom Friedetzky**[7] **Angelika Steger**[8]

In this paper we consider dynamic load balancing algorithms for randomized and adversarial load generation models.

Consider a system of n processors. In our randomized generation models each processor may generate a task with a certain probability at each time step, leading to an expected system load of $\mathcal{O}(n)$. We present a load balancing algorithm that assures that with high probability no processor has a load exceeding $\mathcal{O}(\log\log n)$ at an arbitrary point of time. This improves the $\mathcal{O}\left((\log\log n)^2\right)$ bound of [BFM98].

In the case of the adversarial load generation model each processor may change its load by some constant at each time step. This way, the system load may become arbitrarily large. We present a balancing algorithm and show that if at some point of time no processor has load exceeding $\mathcal{O}\left(\max\left\{\frac{m}{n}, \log\log^2 n\right\}\right)$, then with high probability this holds for the next polynomial number of steps. Here, m is the complete system load at said time step.

Literatur

[BFM98] Petra Berenbrink, Tom Friedetzky, and Ernst W. Mayr. Parallel continous randomized load balancing. In *Proceedings of the 10th Symposium on Parallel Algorithms and Architectures (SPAA '98)*, pages 192–201. ACM, 1998.

[6] Uni Paderborn, pebe@uni-paderborn.de

[7] TU München
friedetz@in.tum.de

[8] TU München
steger@in.tum.de