COMPLEXITY ABSTRACTS 2003. Vol XIII

Abstract

This is a collection of one-page abstracts of recent results of interest to the Complexity community. The purpose of this document is to spread this information, not to judge the truth or interest of the results therein.

TABLE OF CONTENTS

SBP-Completeness and Generalized SBP Computations

Elmar Böhler, Lehrstuhl für Theoretische Informatik, Universität Würzburg, 97074 Würzburg, GERMANY (boehler@informatik.uni-wuerzburg.de)
Christian Glaßer, Lehrstuhl für Theoretische Informatik, Universität Würzburg, 97074 Würzburg, GERMANY (glasser@informatik.uni-wuerzburg.de)
Daniel Meister, Lehrstuhl für Theoretische Informatik, Universität Würzburg, 97074 Würzburg, GERMANY (meister@informatik.uni-wuerzburg.de)

Abstract Number 03-1

SBP is a probabilistic complexity class that is located between BPP and PP. This class generalizes BPP as follows: The probability limit of BPP is $\frac{1}{2}$. This means that an input is accepted if and only if the acceptance probability is $\geq \frac{1}{2}$. Additionally, BPP computations have a probability gap, i.e., the acceptance probability must never belong to some interval around the probability limit $\frac{1}{2}$. In the definition of SBP we still demand a probability gap, but now we allow probability limits that are exponentially small. So already a small acceptance probability suffices to accept the input.
The main result in the first part of the paper gives the following evidence against the existence of many-one complete sets for SBP: BPP is contained in several subclasses of SBP. So in this sense, BPP is a quite strong restriction of SBP. We construct an oracle relative to which SBP does not have a set that is many-one hard for BPP. So any single set in SBP lacks to be powerful enough to solve arbitrary problems in BPP. As a consequence, relative to this oracle, SBP does not have many-one complete sets, and is therefore not uniformly enumerable.
In a second part of this paper we investigate a new operator. Coming from BPP, Schöning defined the operator BP$\cdot$. Similarly, we start from SBP, capture the main ingredients of its definition, and define the operator SB$\cdot$. Our main result in this second part shows that $\mathrm{BP} \cdot \exists \cdot \mathcal{C}$ contains all complexity classes defined by arbitrary application of operators U$\cdot$, $\exists\cdot$, BP$\cdot$, and SB$\cdot$ in any order and number to the complexity class $\mathcal{C}$ if $\mathcal{C}$ fulfills some basic properties.

The paper is available at
http://www.informatik.uni-wuerzburg.de/reports/tr.html.

The Complexity of Boolean Constraint Isomorphism

Elmar Böhler, Theoretische Informatik, Universität Würzburg, Am Hubland, 97074 Würzburg, Germany. (boehler@informatik.uni-wuerzburg.de)
Edith Hemaspaandra, Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623 (eh@cs.rit.edu)
Steffen Reith, Lengfelderstraße 35b, 97078 Würzburg, Germany. (streit@streit.cc)
Heribert Vollmer, Abteilung Theoretische Informatik, Universität Hannover, Appelstr. 4, 30167 Hannover, Germany. (vollmer@thi.uni-hannover.de)

Abstract Number 03-2

In 1978, Schaefer proved his famous dichotomy theorem for generalized satisfiability problems. He defined an infinite number of propositional satisfiability problems (nowadays usually called Boolean constraint satisfaction problems) and showed that all these satisfiability problems are either in P or NP-complete.
In recent years, similar results have been obtained for quite a few other problems for Boolean constraints. Almost all of these problems are related to the satisfiability problem, for example, unique satisfiability, maximum satisfiability, computing the the number of satisfying assignments, finding a minimal satisfying assignment, inverse satisfiability, equivalence, and approximability of some of these problems.
In this paper, we address a problem that is not related to satisfiability, namely, the isomorphism problem for Boolean constraints. Isomorphism is a more complicated problem than satisfiability. The exact complexity of the isomorphism problem for Boolean formulas is still unknown: It is trivially coNP-hard and in Σ_2^p and Agrawal and Thierauf showed that it is not Σ_2^p-hard unless the polynomial hierarchy collapses.
For Boolean constraints, previous work by Böhler et al. showed that the isomorphism problem is either coNP-hard or reducible to the graph isomorphism problem (a problem that is in NP, but not known to be NP-hard), thus distinguishing a hard case and an easier case. However, they did not classify which cases are truly easy, i.e., in P. This paper accomplishes exactly that. It shows that Boolean constraint isomorphism is coNP-hard (and GI-hard), or equivalent to graph isomorphism, or in P, and it gives simple criteria to determine which case holds. We thus establish a tight connection between the isomorphism problem for graphs and that for certain types of Boolean constraints.

A full paper is available ACM Computing Research Repository Technical Report cs.CC/0306134.

The Complexity of Polynomial Time Generation Problems

Elmar Böhler, Lehrstuhl für Theoretische Informatik, Universität Würzburg, 97074 Würzburg, GERMANY (boehler@informatik.uni-wuerzburg.de)
Klaus Wagner, Lehrstuhl für Theoretische Informatik, Universität Würzburg, 97074 Würzburg, GERMANY (wagner@informatik.uni-wuerzburg.de)

Abstract Number 03-3

Many algebraic problems can be described as generation problems. By that we mean the question, whether or not some target element can be generated by applying an operation on a finite set of elements repeatedly. For example the primality problem can be described as the question, whether we can generate n out of the set $\{2, \ldots, \lfloor \frac{n}{2} \rfloor\}$ using multiplication. We make a generalized approach to such problems, and regard the sets GEN(A,$\circ$) which are closures of finite sets A with binary operations $\circ$ that are computable deterministically in polynomial-time. It is not difficult to see that there are operations that have undecidable generation problems. Therefore we additionally want the operations to be *length-monotonic*, which means that the length of a result of a computation has to be at least as large as the length of each of its arguments. Thus restricting the size of a generation tree leads to generation problems that are always in EXPTIME. We find an operation for which the respective generation problem is complete for EXPTIME. For associative and length-monotonic functions the generation problem turns out to be in PSPACE and there are problems which are complete for PSPACE. A similar situation holds for associative, commutative, and length-monotonic operations and NP. An example for generation problems that are NP complete are GEN($\mathbb{N}, +$) or GEN($\mathbb{N} \setminus \{0\}, \cdot$), such that $+$ is the normal addition on $\mathbb{N}$ and $\cdot$ is the normal multiplication on $\mathbb{N} \setminus \{0\}$. Finally, we show that for all bivariate polynomials p the closure GEN($\mathbb{N}, p$) is always in NP.

Full paper is available at:
http://www.informatik.uni-wuerzburg.de/reports/tr.html

Nearly Optimal Language Compression in NP

Harry Buhrman, CWI and University of Amsterdam, P.O. Box 94709, Amsterdam, The Netherlands, (buhrman@cwi.nl)
Troy Lee, University of Amsterdam, P.O. Box 94709, Amsterdam, The Netherlands (tlee@cwi.nl)

Abstract Number 03-4

The CD complexity of a string x is the length of the shortest polynomial time program which accepts only x. The language compression problem is to upper bound the $\mathrm{CD}^{A^{=n}}$ complexity of the strings of length n in some set A. Previous work on the language compression problem has given an upper bound of $2\log\|A^{=n}\| + O(\log n)$, and demonstrated a set A where where this upper bound is tight. This work left open the question of whether or not the factor of 2 is necessary for nondeterministic polynomial time distinguishing complexity, CND.

We answer this question, showing in fact a nearly optimal upper bound of $\log\|A^{=n}\| + O(\log^3 n)$ for the $\mathrm{CND}^{A^{=n}}$ complexity of any set A. A key ingredient of our proof is the connection of Trevisan between extractors and pseudorandom generators.

We also introduce a two-sided randomized version of distinguishing complexity, CBPD, and show there is a set A where the factor of 2 is still necessary for CBPD complexity. On the other hand we show an optimal upper bound of $\log\|A^{=n}\| + O(\log n)$ for CBPD complexity with an NP oracle.

Preliminary version available from www.cwi.nl/~tlee

Arithmetic Constant-Depth Circuit Complexity Classes

Hubie Chen, Department of Computer Science, Cornell University, Ithaca, NY 14853, USA. (hubes@cs.cornell.edu)

Abstract Number 03-5

The boolean circuit complexity classes $AC^0 \subseteq AC^0[m] \subseteq TC^0 \subseteq NC^1$ have been studied intensely. Other than NC^1, they are defined by constant-depth circuits of polynomial size and unbounded fan-in over some set of allowed gates. One reason for interest in these classes is that they contain the boundary marking the limits of current lower bound technology: such technology exists for AC^0 and some of the classes $AC^0[m]$, while the other classes $AC^0[m]$ as well as TC^0 lack such technology.

Continuing a line of research originating from Valiant's work on the counting class $\sharp P$, the arithmetic circuit complexity classes $\sharp AC^0$ and $\sharp NC^1$ have recently been studied. In this paper, we define and investigate the classes $\sharp AC^0[m]$ and $\sharp TC^0$. Just as the boolean classes $AC^0[m]$ and TC^0 give a refined view of NC^1, our new arithmetic classes, which fall into the inclusion chain $\sharp AC^0 \subseteq \sharp AC^0[m] \subseteq \sharp TC^0 \subseteq \sharp NC^1$, refine $\sharp NC^1$. These new classes (along with $\sharp AC^0$) are also defined by constant-depth circuits, but the allowed gates compute arithmetic functions. We also introduce the classes $\mathit{Diff}\,AC^0[m]$ (differences of two $\sharp AC^0[m]$ functions), which generalize the class $\mathit{Diff}\,AC^0$ studied in previous work. We study the structure of three hierarchies: the $\sharp AC^0[m]$ hierarchy, the $\mathit{Diff}\,AC^0[m]$ hierarchy, and a hierarchy of language classes. We prove class separations and containments where possible, and demonstrate relationships among the various hierarchies. For instance, we prove that the hierarchy of classes $\sharp AC^0[m]$ has exactly the same structure as the hierarchy of classes $AC^0[m]$:

$$AC^0[m] \subseteq AC^0[m'] \text{ iff } \sharp AC^0[m] \subseteq \sharp AC^0[m']$$

We also investigate closure properties of our new classes, which generalize those appearing in previous work on $\sharp AC^0$ and $\mathit{Diff}\,AC^0$.

A full paper is available at http://www.cs.cornell.edu/hubes.

Inverse NP Problems

Hubie Chen, Department of Computer Science, Cornell University, Ithaca, NY 14853, USA. (hubes@cs.cornell.edu)

Abstract Number 03-6

One characterization of the class NP is as the class of all languages for which there exists a verifier, operating in polynomial time, with the following properties: for every member of the language, there exists a polynomially-sized proof causing the verifier to accept; and, for every non-member, there is no proof causing the verifier to accept. Relative to a particular verifier, every member x of the language induces a set of proofs, namely, the set of proofs causing the verifier to accept x as a member. This paper studies the complexity of deciding, given a set Π of proofs, whether or not there exists some x inducing Π (relative to a particular verifier). We call this decision problem the inverse problem for the verifier. We develop a new notion of reduction which allows one to compare the complexity of inverse problems. Using this notion, we classify as coNP-complete the inverse problems for the "natural" verifiers of many NP-complete problems. We also show that the inverse complexity of a verifier for a language L cannot be predicted solely from the complexity of L, but rather, is highly dependent upon the choice of verifier used to accept L. In this context, a verifier with a Σ_2^p-complete inverse problem is exhibited, giving a new and natural example of a Σ_2^p-complete problem.

A full paper is available at http://www.cs.cornell.edu/hubes.

A Note on the Classical Lower Bound for a Quantum Walk Problem

Stephen A. Fenner, Department of Computer Science and Engineering, University of South Carolina, Columbia, South Carolina 29208 USA. (fenner@cse.sc.edu)
Yong Zhang, Department of Computer Science and Engineering, University of South Carolina, Columbia, South Carolina 29208 USA. (zhang29@cse.sc.edu)

Abstract Number 03-7

Childs, Cleve, Deotto, Farhi, Gutmann, and Spielman [STOC '03, quant-ph/0209131] recently described a black-box graph reachability problem that can be solved in polynomial time by a quantum random walk, but which cannot be solved by any classical algorithm in subexponential time. This exponential speed-up of a quantum algorithm over the best classical algorithm is significant in that the quantum algorithm does not use any version of the quantum Fourier transform.

In this note, we improve the lower bound for the classical algorithm significantly, as well as providing a straightforward and complete analysis of the bound.

The exponential lower bound of Childs, et al. states that, for input size n, a classical algorithm running in time t has success probability bounded by $O(t^2 2^{-n/2})$. We improve this bound on the probability to $O(t^2 n 2^{-n})$ and show that this bound is almost tight. In particular, a classical algorithm running in time $2^{n/3}$ has success probability $O(n2^{-n/3})$. The constant factor in this bound can be made arbitrarily close to one.

A full paper will be available shortly at
http://www.cse.sc.edu/~fenner/papers/walk.ps.

Reductions between Disjoint NP-Pairs

Christian Glaßer, Lehrstuhl für Informatik IV, Universität Würzburg, 97074 Würzburg, GERMANY. (glasser@informatik.uni-wuerzburg.de)
Alan L. Selman, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260. (selman@cse.buffalo.edu)
Samik Sengupta, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260. (samik@cse.buffalo.edu)

Abstract Number 03-8

We prove that all of the following assertions are equivalent: There is a many-one complete disjoint NP-pair; there is a strongly many-one complete disjoint NP-pair; there is a Turing complete disjoint NP-pair such that all reductions are smart reductions; there is a complete disjoint NP-pair for one-to-one, invertible reductions; the class of all disjoint NP-pairs is uniformly enumerable.
Let A, B, C, and D be nonempty sets belonging to NP. A *smart* reduction between the disjoint NP-pairs (A, B) and (C, D) is a Turing reduction with the additional property that if the input belongs to $A \cup B$, then all queries belong to $C \cup D$. We prove under the reasonable assumption UP $\cap$ co-UP has a P-bi-immune set that there exist disjoint NP-pairs (A, B) and (C, D) such that (A, B) is truth-table reducible to (C, D), but there is no smart reduction between them. This paper contains several additional separations of reductions between disjoint NP-pairs.
We exhibit an oracle relative to which there exists a truth-table-complete disjoint NP-pair while no disjoint NP-pair is many-one complete.

A full paper is available at Electronic Colloquium on Computational Complexity, TR03-027.
(ftp://ftp.eccc.uni-trier.de/pub/eccc/reports/2003/TR03-027/index.html)

Complexity of Cycle Length Modularity Problems in Graphs

Edith Hemaspaandra, Department of Computer Science, Rochester Institute of Technology, Rochester, NY 14623, USA. (eh@cs.rit.edu)
Holger Spakowski Institut für Informatik, Heinrich-Heine-Universität Düsseldorf, 40225 Düsseldorf, Germany. (spakowsk@cs.uni-duesseldorf.de).
Mayur Thakur, Department of Computer Science, University of Rochester, Rochester, NY 14627. (thakur@cs.rochester.edu)

Abstract Number 03-9

The even cycle problem for both undirected and directed graphs has been the topic of intense research in the last decade. In this paper, we study the computational complexity of *cycle length modularity problems*. Roughly speaking, in a cycle length modularity problem, given an input (undirected or directed) graph, one has to determine whether the graph has a cycle C of a specific length (or one of several different lengths), modulo a fixed integer. We denote the two families (one for undirected graphs and one for directed graphs) of problems by (S, m)-UC and (S, m)-DC, where $m \in \mathcal{N}$ and $S \subseteq \{0, 1, \ldots, m-1\}$. (S, m)-UC (respectively, (S, m)-DC) is defined as follows: Given an undirected (respectively, directed) graph G, is there a cycle in G whose length, modulo m, is a member of S? In this paper, we fully classify (i.e., as either polynomial-time solvable or as NP-complete) each problem (S, m)-UC such that $0 \in S$ and each problem (S, m)-DC such that $0 \notin S$. We also give a sufficient condition on S and m for the following problem to be polynomial-time computable: (S, m)-UC such that $0 \notin S$.
A full paper is available as URCS-TR-2003-802 from
http://www.cs.rochester.edu/trs/theory-trs.html

Lower Bounds and the Hardness of Counting Properties

Lane A. Hemaspaandra, Department of Computer Science, University of Rochester, Rochester, NY 14627. (lane@cs.rochester.edu)
Mayur Thakur, Department of Computer Science, University of Rochester, Rochester, NY 14627. (thakur@cs.rochester.edu)

Abstract Number 03-10

Rice's Theorem states that all nontrivial language properties of recursively enumerable sets are undecidable. Borchert and Stephan started the search for *complexity-theoretic* analogs of Rice's Theorem, and proved that every nontrivial counting property of boolean circuits is UP-hard. Hemaspaandra and Rothe improved the UP-hardness lower bound to $\mathrm{UP}_{O(1)}$-hardness. The present paper raises the lower bound for nontrivial counting properties from $\mathrm{UP}_{O(1)}$-hardness to FewP-hardness, i.e., from constant-ambiguity nondeterminism to polynomial-ambiguity nondeterminism.
Furthermore, we prove that no relativizable technique can raise this lower bound to FewP-$\leq^p_{1\text{-}tt}$-hardness. We also prove a Rice-style theorem for NP, namely that every nontrivial language property of NP sets is NP-hard.

A full paper is available as URCS-TR-2002-768 from
http://www.cs.rochester.edu/trs/theory-trs.html

One-Way Permutations and Self-Witnessing Languages

Christopher M. Homan, Department of Computer Science, University of Rochester, Rochester, NY 14627. (choman@cs.rochester.edu)
Mayur Thakur, Department of Computer Science, University of Rochester, Rochester, NY 14627. (thakur@cs.rochester.edu)

Abstract Number 03-11

A desirable property of one-way functions is that they be total, one-to-one, and onto—in other words, that they be permutations. We prove that one-way permutations exist exactly if $P \neq UP \cap coUP$. This provides the first characterization of the existence of one-way permutations based on a complexity-class separation and shows that their existence is equivalent to a number of previously studied complexity-theoretic hypotheses.
We study permutations in the context of witness functions of nondeterministic Turing machines. A language is in PermUP if, relative to some unambiguous, nondeterministic, polynomial-time Turing machine accepting the language, the function mapping each string to its unique witness is a permutation of the members of the language. We show that, under standard complexity-theoretic assumptions, PermUP is a strict subset of UP.
We study SelfNP, the set of all languages such that, relative to some nondeterministic, polynomial-time Turing machine that accepts the language, the set of all witnesses of strings in the language is identical to the language itself. We show that $SAT \in SelfNP$, and, under standard complexity-theoretic assumptions, $SelfNP \neq NP$.

A full paper is available as URCS-TR-2001-760 from
http://www.cs.rochester.edu/trs/theory-trs.html

On Characterizations of the BFFs, Part II

Robert J. Irwin, Computer Science Dept., SUNY/Oswego, Oswego, NY 13126, USA. rjirwin@cs.oswego.edu
Bruce M. Kapron, Dept. of Computer Science, University of Victoria, Victoria, BC V8W 3P6, Canada. bmkapron@maclure.csc.uvic.ca
James S. Royer, Dept. of Elec. Eng. and Computer Science, Syracuse University, Syracuse, NY 13244, USA. royer@ecs.syr.edu

Abstract Number 03-12

This paper's predecessor (On Characterizations of the BFFs, Part I, by R. Irwin, B. Kapron, and J. Royer, *J. of Functional Programming* **11** (2001) 117–153) concerned Cook and Kapron's class of basic feasible functionals (BFFs) at type-level 2. This sequel concerns the full class at all simple types. The move beyond type-level two entails broadening the investigation in two ways:

- Along with BFF we are lead to consider D, a closely related class of functionals introduced by Seth. Seth conjectured that above type-level two, D is a strictly larger than BFF. We confirm this conjecture and show that the complexity classes D and BFF separate at type-level 3.
- We are also led to study how different semantic domains can support distinct notions of feasible computation. Most prior work on higher-type feasibility concerned classes of functionals defined on $\mathcal{F}ull$, the full type hierarchy over the natural numbers (which consists of the straightforward set-theoretic interpretation of the simple types over $\mathbb{N}$). In many respects $\mathcal{F}ull$ is a highly unnatural setting in which to study computation. Difficulties with $\mathcal{F}ull$ and with the class of continuous functionals lead us to introduce the domain of *bounded continuous functionals* (denoted $\mathcal{BC}ont$) at all simple types and to study BCBFF and BCD, the $\mathcal{BC}ont$ versions of BFF and D, respectively. $\mathcal{BC}ont$ seems a more natural setting for the ideas behind D than does $\mathcal{F}ull$. Moreover, relative to $\mathcal{BC}ont$, there turns out to be a simple, compositional sense in which the elements of BCBFF and BCD are polynomial-time computable. We show that there is also a corresponding sense in which, relative to $\mathcal{F}ull$, the elements of BFF and D are polynomial-time computable, but this sense seems to be inherently more convoluted.

The paper's basic goal is to plainly explain the senses in which the elements of D, BFF, BCD, and BCBFF are polynomial-time computable relative to their respective semantic settings. Machine models for higher-type computation play an important role in this.

A full paper is available via ftp://ftp.cis.syr.edu/users/royer/p2US.pdf

On Type-2 Complexity Classes

Chung-Chih Li, Computer Science Department, Lamar University, Beaumont, TX 77706, USA. cli@gt.rr.com
James S. Royer, Department of Elec. Eng. and Computer Science, Syracuse University, Syracuse, NY 13244, USA. royer@ecs.syr.edu

Abstract Number 03-13

There are now a number of things called "higher-type complexity classes." The most promenade of these is the class of *basic feasible functionals*, a fairly conservative higher-type analogue the (type-1) polynomial-time computable functions. There is, however, currently no satisfactory general notion of what a higher-type complexity class should be. In this paper we propose one such notion for type-2 functionals and begin an investigation of its properties.

The most striking difference between our type-2 complexity classes and their type-1 counterparts is that, because of topological constrains, the type-2 classes have a much more ridged structure. *Example:* It follows from McCreight and Meyer's Union Theorem that the (type-1) polynomial-time computable functions form a complexity class. The analogous result *fails* for the class of type-2 basic feasible functionals.

A preliminary version this paper is available via ftp://ftp.cis.syr.edu/users/royer/ttcc.ps

Time-bounded Quantum Kolmogorov Complexity

Harumichi Nishimura, School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada, K1N 6N5. (anishi@site.uottawa.ca)
Tomoyuki Yamakami, School of Information Technology and Engineering, University of Ottawa, Ottawa, Ontario, Canada, K1N 6N5. (yamakami@site.uottawa.ca)

Abstract Number 03-14

The notion of Kolmogorov complexity gives us numerous applications in various fields. Recently, Vitányi studied its quantum analogue that measures the shortest classical programs approximating given quantum states in finite time. Berthiaume, van Dam, and Laplante, in contrast, studied the shortest quantum programs (that is, quantum states). Moreover, a different approach toward quantum algorithmic entropy was made by Gács. However, quantum Kolmogorov complexities by their studies were time-unbounded since they were from a quantum information theoretic viewpoint. In our research, we rather focus on quantum Kolmogorov complexity of classical strings based on polynomial-time quantum computations. In addition to quantum variant of "regular" time-bounded Kolmogorov complexity, QCG-complexity, we introduce quantum variants of distinguishing complexity (CD-complexity) and instance complexity (IC-complexity) by classical programs, QCD-complexity and QIC-complexity. We also consider their quantum program versions for these quantum variants, qQCG-complexity, qQCD-complexity and qQIC-complexity. We show fundamental properties of these complexity measures and discuss gaps among their quantum measures and classical measures for clssical strings. In particular, we show the existence of strings x such that: (i) CD-complexity is at least $\log|x| - 1$ and QCG-complexity is $O(1)$ in a relativized world; (ii) QCG-complexity is $\Omega(|x|)$ and CD-complexity is $O(\log|x|)$ in a relativized world; (iii) qQCD-complexity is $O(\log|x|)$ and qQCG-complexity is $\Omega(|x|)$. Furthermore, we investigate relationships between these measures and complexity classes, which includes a new relativized world that EQP is not included in a class defined under instance complexity, $\mathrm{IC}[n/5 + O(1), 2^{n/2} + O(1)]$.

A full paper will appear soon at http://arxiv.org/archive/quant-ph

On the Computational Complexity of Longley's H Functional

James S. Royer, Department of Elec. Eng. and Computer Science, Syracuse University, Syracuse, NY 13244, USA. `royer@ecs.syr.edu`

Abstract Number 03-15

In the late 1960s Dana Scott introduced the prototype programming language PCF in the course of his study of programming language semantics. Scott noted that all of his semantic models of PCF included "junk." That is, PCF is an intuitively sequential (one thing definitely following another) programming language, but all of Scott's models included nonsequential elements. Constructing a "no junk" model for PCF, was a major open problem for about twenty years. The reason for its prominence was that capturing higher-type sequentiality in a semantics seemed important and proved tricky. When solutions to the PCF problem finally appeared, it became apparent that PCF itself was too restrictive to embody "higher-type sequentiality."

In the late 1990s van Oosten introduced a class of sequentially computable higher-type functionals that properly includes the PCF-computable functionals. This class now goes under the name of the *sequentially realizable functionals* (denoted SR) and can be explained via simple dialog games. SR relates to the work of Curien and others on sequentiality and has very strong and natural mathematical properties. In particular, SR can be viewed as a natural generalization of Turing reducibility to higher types.

Recently John Longley discovered an SR-functional H that, when added to the language PCF, yields a language PCF+H that computes exactly SR. Longley hesitated to recommend PCF+H as the basis of a practical programming language because the only known ways of computing H seemed to involve high computational costs. In this paper we confirm Longley's doubts. We establish that if P is different from NP, then the computational complexity of H (and similar functionals) is inherently infeasible.

The paper is to appear in *Theoretical Computer Science*. A version of the paper is available via `ftp://ftp.cis.syr.edu/users/royer/lh.pdf`

On a D-N-optimal acceptor for TAUT

Zenon Sadowski, Institute of Mathematics, University of Białystok, 15-267 Białystok, ul. Akademicka 2, POLAND, (sadowski@math.uwb.edu.pl)

Abstract Number 03-16

In this paper we introduce and study a new type of optimal acceptor, a D-N-optimal acceptor for $TAUT$ (the previously investigated optimal acceptors are named in our nomenclature as D-D-optimal and N-N-optimal). A deterministic algorithm recognizing $TAUT$ is a D-N-optimal acceptor for $TAUT$ if no other nondeterministic algorithm accepting $TAUT$ has more than a polynomial speed-up over its running time on instances from $TAUT$. The existence of such an acceptor implies that **NP**=co-**NP** is equivalent to **P**=**NP**.

We develop further the earlier observed connection between optimal acceptors for $TAUT$, optimal propositional proof systems, and the structure of easy subsets of $TAUT$. Namely, we prove that the existence of a D-N-optimal acceptor for $TAUT$ is equivalent to the existence of an optimal and automatizable propositional proof system and to the existence of a suitable recursive presentation of the class of all **NP**-easy (acceptable by nondeterministic polynomial time machines) subsets of $TAUT$.

Additionally we show that the question of whether every proof system is weakly automatizable can be related to the Complexity Theory questions studied before: of whether every disjoint **NP**-pair is **P**-separable and of whether every function in **NPSV** has a total extension in **FP**. As a corollary we have deduced that the existence of a D-N-optimal acceptor for $TAUT$ implies that every disjoint **NP**-pair is **P**-separable.

A full paper is available by email to sadowski@math.uwb.edu.pl

Polylogarithmic-round Interactive Proofs for coNP Collapses the Exponential Hierarchy

Alan Selman, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260. (selman@cse.buffalo.edu)
Samik Sengupta, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY 14260. (samik@cse.buffalo.edu)

Abstract Number 03-17

Boppana, Haståd, and Zachos have shown that if every language in coNP has a constant-round interactive proof system, then the polynomial hierarchy collapses. On the other hand, Lund, Fortnow, Karloff, and Nisan have shown that every set in the polynomial hierarchy, and therefore every set in coNP, has a linear-round interactive protocol.

We show that if all sets in coNP have a polylogarithmic-round interactive proof then the exponential hierarchy collapses to the third level, i.e., $\mathrm{NEXP}^{\Sigma_2^p} = \mathrm{coNEXP}^{\Sigma_2^p}$. In order to prove this, we obtain an exponential version of Yap's result, and improve upon the exponential version of the Karp-Lipton theorem, obtained first by Buhrman and Homer.

A full paper is available by email to samik@cse.buffalo.edu