ICM 1998, Berlin, Aug. 18–27

Abstracts of Plenary and Invited Lectures

Section:

14. Mathematical Aspects of Computer Science

1991 MS Classification: 68Q10, 68Q15
Sudan, Madhu, Massachusetts Institute of Technology, USA
Probabilistic checking of proofs

The notion of efficient verification of proofs has been a central theme in the theory of computing. The fundamental complexity class NP arises out of the study of this concept; as does the well-known question "Is P=NP?". Over the years, the class NP has played a crucial role in separating potentially intractable problems from the tractable ones, especially in the study of combinatorial optimization. Research in this decade has seen a revival of the connection between proofs and optimization. This revival is motivated by the proposal of several alternatives to the notion of efficient verifiability of proofs; and results showing that seemingly restrictive notions of verification are surprisingly general. The revived connection shows that for many optimization problems finding a "nearly-optimal" solution is as hard as finding the optimal one. In this talk we will describe some of the early results in this sequence of new developments.
A computational view of the notion of a proof abstracts the semantics of the proof system into a verification procedure or *verifier* i. e., a polynomial time computable Boolean function described by a Turing machine. A purported theorem T and proof π are then just a sequence of bits; π proves T if the verifier accepts the pair (T, π). The purported theorem T is true if such a proof π exists. The class NP [5, 9] represents the class of all theorems with "short" proofs.
The new notions of proof systems proposed during the last ten years consider a probabilistic verifier who is allowed some small probability of making an error. One of these notions, known as a probabilistically checkable proof (PCP), is motivated by the following informally stated question: "How fast can the verifier be compared to the size of the proof?" It is easy to check that a deterministic verifier must at the very least "look" at the whole proof. This however need not be true for probabilistic ones. The notion of "looking at a bit of the proof". can be formalized by providing the verifier with oracle access to the proof, i. e., the verifier can specify the address of a location of the proof and gets back the bit written in that location and this entire process takes only as much time as required to write the address. The number of bits of the proof that are "looked" at is now the number of oracle queries. To contrast such a verifier with the traditional verifier, one also quantifies the amount of randomness used by such a verifier. Thus we define $(r(\cdot), q(\cdot))$-restricted PCP verifier to be a probabilistic verifier with

random access to a proof oracle, such that on input x of length n, the verifier tosses at most $r(n)$ coins and accesses the oracle at most $q(n)$ times, where the locations accessed are a function of the random coins. A language L is said to be in class $\mathrm{PCP(r, q)}$ if there exists an $(r(\cdot), q(\cdot))$-restricted PCP verifier V satisfying the following: If $x \in L$ there exists an oracle π such that the verifier V accepts x with probability 1 on oracle access to π. If $x \notin L$, for every π, V accepts x with probability at most 1/2. Notice that the verifier can make mistakes when $x \notin L$.
It is immediate from the definition that $\mathrm{NP} = \cup_{c>0}\mathrm{PCP}(0, n^c)$ (the verifier is not randomized, but is allowed unlimited access to the proof). The results of [3, 4, 6] showed that by allowing the verifier small amounts of randomness, the query complexity can be reduced dramatically and in particular $\mathrm{NP} \subseteq \cup_{c>0}\mathrm{PCP}(c \log n \log\log n, c \log n \log\log n)$. Subsequently, [1, 2], showed that it is possible to restrict the verifier even more significantly to just a *constant* number of queries (independent of the theorem, the proof or the system of axioms). They also reduce randomness to strictly logarithmic in input size. Specifically, they show

$$\exists q < \infty \text{ such that } \mathrm{NP} = \cup_{c>0}\mathrm{PCP}(c \log n, q).$$

The main focus of this talk will be to describe the ingredients of this result.

The consequences to combinatorial optimization may be described informally as follows: The notion of a PCP verifier allows one to formalize the notion of an "approximately" correct proof; and the strong results obtained above show that such a proof exists if and only if a perfectly correct proof exists. Thus the task of finding an approximately correct proof is as hard as the task of finding a perfectly correct proof. The traditional connection between proofs and optimization [8, 9] now indicates that for some optimization problems (unfortunately, not necessarily natural ones) finding near-optimal solutions should be as hard as finding optimal ones. The statement can actually be formalized and made applicable even to natural optimization problems. This connection was discovered by [6] and its applicability was further extended in [2] to apply to a large number of optimization problems considered in [10].

References

[1] S. Arora and S. Safra. Probabilistic checking of proofs: a new characterization of NP. *Proc. FOCS*, 1992.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof verification and the hardness of approximation. *Proc. FOCS*, 1992.

[3] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[4] L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. *Proc. STOC*, 1991.

[5] S. Cook. The complexity of theorem-proving procedures. *Proc. STOC*, 1971.

[6] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *J. of the ACM*, 43(2):268-292, 1996.

[7] J. Håstad. Some optimal inapproximability results. *Proc. STOC*, 1997.

[8] R. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, ed., *Complexity of Computer Computations*, Advances in Computing Research, 85–103. Plenum Press, 1972.

[9] L. Levin. Universal'nyĭe perebornyĭe zadachi. *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.

[10] C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *J. Computer and System Sciences* 43:425–440, 1991.