To be published in **Annotated Bibliographies in Combinatorial Optimization**

Hardness of Approximation

Viggo Kann
Royal Institute of Technology, Stockholm

Alessandro Panconesi
Freie Universität, Berlin

CONTENTS

The aim of computational complexity theory is to devise tools that enable us to classify computable functions according to how expensive it is to compute them. The cost is usually (but by no means always) given by the *time* needed for the computation, a notion that can be defined in a technology independent manner by means of Turing machines or other equivalent abstractions.

A turning point in the development of computational complexity was the introduction of the theory of NP-completeness. This theory has now found a stable place in introductory books and courses on algorithms and computational complexity and we assume familiarity with it. The standard reference on the subject is the book [Garey and Johnson 1979] (see §1.1). The other textbooks cited in §1.1 are also good introductions to the subject and to the other notion that we assume familiar to the reader, that of *optimization problems.*

An optimization problem can be seen as a function $f : I \to N$ between the set of input instances and the natural numbers. Ideally, we would like to determine the function $t_f(n)$ describing the running time needed to compute f on inputs of size n.

NP-completeness can be used as a first rough approximation to this classification problem. The standard way to proceed is as follows: instead of considering the problem "given x, compute $f(x)$" one introduces an integer parameter k and considers the problem "given (x, k), is $f(x) > k$?" Note that the first problem asks for computing a function, whereas the second is a problem of *language membership.* That is, for a given pair (x, k) the answer can be *yes* or *no.* In this fashion, corresponding to each function f, one can define the language L_f as the set of all pairs (x, k) that have a *yes* answer.

The theory of NP-completeness deals with languages. The well-known conjecture $P \neq NP$ states that if a language is NP-complete then it cannot be recognized in polynomial time. In particular, if one can prove for a certain function f that the corresponding language L_f is NP-complete then one has a lower bound result: f

cannot be computed in polynomial time, unless P = NP (because as soon as you know $f(x)$ you also know if $f(x) > k$). When this happens we say that computing f is NP-hard.

This mathematical fact is important because it *may* tell us a lot about the world. Literally hundreds, perhaps thousands, of optimization problems arising naturally in the most diverse contexts—economics, molecular biology, algebra, logic, computer science, all sorts of engineering, etc—are now known to be NP-hard. So, if the conjecture P $\neq$ NP is correct we must compute in a world that is inherently computationally intractable.

This is the starting point of the theory of approximation algorithms. Optimization problems arise in practical contexts and we need to do something about them. A natural suggestion is the following. An optimization problem is the task of maximizing (or minimizing) a certain objective function over a space of feasible solutions that usually cannot be searched efficiently because of its huge size. Given that computing the optimum would take too long (if P $\neq$ NP) then one could settle for a fast algorithm that is *guaranteed to always deliver* good solutions, say, solutions that, for maximization problems, are always at least 69 % of the best possible.

This leads to the following definition: a maximization problem F is *approximable within* $1+\rho$ if there exists a polynomial time algorithm A such that, for all inputs I,

$$\frac{opt_F(I)}{A(I)} \leq 1+\rho,$$

where $A(I)$ denotes the output of the algorithm and $opt_F(I)$ corresponds to $f(I)$, i.e. denotes the optimum value on input I. For minimization problems one considers the inverse ratio. The $1+\rho$ notation is suggestive of the fact that the closer we are to 1, the better the approximation algorithm. This definition poses a new computational complexity question: *given an optimization problem F, what is the smallest ρ for which it is approximable within $1+\rho$ in polynomial time?*

We remark that in this definition of approximation it is required that the approximation be constructive, i.e. that a feasible solution be delivered, instead of just a numerical value. Under this assumption one always has $A(I) \leq opt_F(I)$ for maximization problems and $A(I) \geq opt_F(I)$ for minimization problems.

It was quickly observed that NP optimization problems (i.e. functions f whose language L_f is in NP) display different kinds of behaviour:

- Some problems are not approximable within any $1+\rho$, unless P = NP. This is for instance the case with the travelling salesperson problem and with integer programming.

- Some problems are approximable within *some* constant $1+\rho$. The class consisting of these problems is called APX. For instance, MAX CUT is approximable within 2 (one can always find a cut whose size is at least half of the edges).

- Some problems are approximable within $1+\rho$ *for all* ρ's greater than 0. In other words, for each $\rho > 0$ there is a $(1+\rho)$-approximate polynomial time algorithm A_ρ. The infinite sequence $\{A_\rho : \rho > 0\}$ is called a *polynomial time approximation scheme* (*ptas*). The class of problems admitting such a scheme is called PTAS. Examples of problems in PTAS are MAX INDEPENDENT SET for planar graphs and finding the minimum length schedule on parallel machines. Notice that each A_ρ must run in time polynomial in the size of the input, but could depend exponentially (or worse) in $1/\rho$.

- The last inconvenience is obviated if we can find a *fully polynomial approximation scheme*, that is, a *ptas* such that the running time of each A_ρ is polynomial in both $|I|$ and $1/\rho$. The resulting class is denoted as FPTAS. An example of a problem in FPTAS is MAX KNAPSACK.

The theory of approximation algorithms tries to place problems in these classes by proving upper bounds—i.e. the existence of an approximation algorithm or scheme—and lower bounds—by showing that a problem is not approximable within a certain accuracy. Sometimes it is possible to prove that a problem is approximable or non-approximable within $1+\rho(n)$, where $\rho(n)$ is a function of the input size. In these cases, upper bounds are of particular interest when $\rho(n)$ is a slowly growing function such as $\log n$, whereas lower bounds are particularly depressing when $\rho(n)$ grows as fast as $\sqrt{n}$ or worse.

The field of approximation algorithms has developed tremendously in the 20 years that have passed since its birth. In this survey we first give a list of books and lecture notes dealing with the field in general, and then focus on hardness results, i.e. proving that approximating within some accuracy is impossible. So far, no absolute hardness results for approximation have been proven or are in sight. Rather, hardness results are established assuming some widely held complexity theoretic conjecture; typically, but not exclusively, that P $\neq$ NP. Even then, proving hardness results for approximation has proven next to impossible for a long time, but the situation has changed dramatically in the past 3 or 4 years with the advent of the PCP revolution, still in progress. This is still very much an open field with a lot of frenetic activity and plenty of challenging open problems.

When available, we shall always give the journal reference. The reader should be advised that this might substantially alter the chronology of the various contributions, since many results are published in journals only several years after their first conference appearance. When applicable, we shall also add a pointer to a web page or a short reference to the two premiere algorithm conferences in the computer science community, namely the ACM Symposium on Theory of Computing (STOC) and the IEEE conference on the Foundations of Computer Science (FOCS). These extra references might be useful to the reader since they are often easily available.

1 Books and Surveys

1.1 Books

We list some books, starting with some suitable for the newcomer to the area of approximation, and proceeding with increasingly specialized ones.

T. Cormen, C. Leiserson, R. Rivest (1989). *Introduction to Algorithms*, The MIT Press, Cambridge, Massachusetts.
E. Horowitz, S. Sahni (1978). *Fundamentals of Computer Algorithms*, Pitman, London.

C. H. Papadimitriou, K. Steiglitz (1982). *Combinatorial Optimization, Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, New Jersey.

These are three good undergraduate level books in algorithms which contain some clearly written introductory chapters to the field of approximation algorithms.

C. H. Papadimitriou (1994). *Computational Complexity*, Addison-Wesley, Amsterdam.
D. P. Bovet, P. Crescenzi (1994). *Introduction to the Theory of Complexity*, Prentice Hall, London.

These books deal with complexity theory and therefore approach approximation from this angle. Roughly speaking, while the books cited earlier focus on algorithms, these ones focus on lower bounds. The treatment in the book by Papadimitriou is more extensive and covers up-to-date topics such as MAX SNP in some detail.

M. R. Garey, D. S. Johnson (1979). *Computers and Intractability: a guide to the theory of NP-completeness*, W. H. Freeman and Company, San Francisco.

This book is the encyclopedia of NP-completeness, but it also contains a chapter on approximation with definitions of how to measure approximability and examples of how to show upper and lower bounds on the approximability of some problems. The notation introduced here is still used.

D. S. Hochbaum (ed.) (1997). *Approximation Algorithms for NP-Hard Problems*, PWS Publishing Company, Boston.

This is a great new book covering more advanced and up-to-date topics, which makes it perhaps suitable for a graduate course. It consists of 13 chapters, each a self-contained survey on some aspects of approximation algorithms. Most chapters cover algorithms for different types of problems together with general techniques for approximating those problems. Noteworthy is the chapter about hardness of approximation by Arora and Lund. It surveys recent results while at the same time serving as a tutorial for anyone who wants to learn how to construct lower bounds on approximability. This book should be of interest to students and researchers alike.

Finally, once you are ready to engage in research this resource might prove valuable:

P. Crescenzi, V. Kann (1995). *A compendium of NP optimization problems*, Technical Report SI/RR-95/02, Dipartimento di Scienze dell'Informazione, Università di Roma "La Sapienza". The list is updated continuously. The latest version is available as `http://www.nada.kth.se/theory/problemlist.html`.

This is a Garey-and-Johnson style list of about two hundred NP-hard optimization problems and several variations thereof. For each problem the best currently known upper and lower bounds on approximability are given (to the best of the authors' knowledge), together with references.

1.2 Surveys and lecture notes

There are several surveys that give a good overview of the field.

D. B. Shmoys (1995). Computing near-optimal solutions to combinatorial optimization problems. W. Cook, L. Lovász, P. Seymour (eds.). *Special year on combinatorial*

optimization, DIMACS Series in Discrete Math. and Theoretical Comput. Sci., 20, American Math. Soc.

This is a comprehensive, clear and insightful survey, and perhaps the broadest of the surveys we mention. Although its viewpoint is more algorithmic than structural, a good deal about lower bounds can be learned from it, including recent developments following the PCP theorem.

R. Motwani (1992). *Lecture notes on approximation algorithms – volume I*, Technical Report STAN-CS-92-1435, Department of Comput. Sci., Stanford University. Available as `http://theory.stanford.edu/people/motwani/postscripts/approximations.ps.Z`.

These lecture notes are a clear introduction to approximation algorithms and cover a large part of the basic general results, but also contain some clearly explained advanced material such as asymptotic approximation schemes for MIN BIN PACKING. Recent hardness topics such as MAX SNP or PCP are not covered.

M. X. Goemans (1994). *Lecture notes*, Technical report, Department of Math., MIT. Available as `ftp://theory.lcs.mit.edu/pub/classes/18.415/notes-approx.ps`.

The lecture notes by Goemans concentrate on recent algorithmic results, and complement Motwani's lecture notes quite well.

There are also surveys dealing with structural issues instead of methods for finding upper and lower bounds:

D. Bruschi, D. Joseph, P. Young (1989). A structural overview of NP optimization problems. *Proc. Optimal Algorithms*, Berlin. Springer-Verlag, 205–231. Lecture Notes Comput. Sci. 401. It also appeared 1991 in *Algorithms Rev.* 2, 1–26.

This survey is quite detailed and covers most known structural results up to 1989, but as much has happened since, it is now rather outdated.

V. Kann (1992). *On the Approximability of NP-complete Optimization Problems*, Technical Report TRITA-NA-9206, Department of Numer. Anal. and Computing Sci., Royal Institute of Technology, Stockholm.

This thesis defines, discusses and compares different measures of approximation, types of approximation preserving reductions, and approximation complexity classes, including the syntactically defined classes. It also contains some hardness results.

G. Ausiello, P. Crescenzi, M. Protasi (1995). Approximate solution of NP optimization problems. *Theoret. Comput. Sci.* 150, 1–55.

This is a survey dealing with completeness in approximation classes and related topics. It also discusses the difference between computing an approximate value and constructing a solution with that value.

Finally, we cite some surveys describing the interplay between interactive proofs and approximation:

D. S. Johnson (1992). The NP-completeness column: an ongoing guide; The tale of the second prover. *J. Algorithms* 13, 502–524.
L. Babai (1994). Transparent proofs and limits to approximation. *Proc. First*

European Congress of Mathematicians, Basel. Birkhäuser. Expanded version available as `ftp://ftp.cs.uchicago.edu/pub/publications/tech-reports/TR-94-07.ps`.
J. Håstad (1994). Recent results in hardness of approximation. *Proc. 4th Scand. Workshop Algorithm Theory*, Berlin. Springer-Verlag, 231–239. Lecture Notes Comput. Sci. 824.
M. Bellare (March 1996). Complexity theory column 12: Proof checking and approximation: towards tight results. *SIGACT News*, 27, The Association for Computing Machinery, 2–13. Expanded version available as
`http://www-cse.ucsd.edu/users/mihir/papers/pcp-survey.ps.gz`.

The first three are good introductions to the PCP theorem and the role of interactive proofs in proving lower bounds for approximation. The fourth is a good introduction to further improvements of the PCP theorem and their consequences for approximation.

2 Early Lower Bounds

Although the notion of approximation algorithm for combinatorial optimization problems had appeared before, to our knowledge the organized study of algorithms with guaranteed approximation began in the seventies with the following paper:

D. S. Johnson (1974). Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.* 9, 256–278.

Johnson here presents and analyzes some important (and still useful) approximation algorithms for common optimization problems, such as MAX SATISFIABILITY, MIN DOMINATING SET, MAX CLIQUE and MIN GRAPH COLORING. The paper emphasizes the notion of *performance guarantee* as opposed to a heuristic that is expected to work fine in practice but without any proven guarantee.

The field developed quite rapidly and many clever approximation algorithms and schemes were developed. A survey on early approximation algorithms is the following:

M. R. Garey, D. S. Johnson (1976). *Approximation algorithms for combinatorial problems: an annotated bibliography*, Academic Press, New York, 41–52.

Of course this annotated bibliography is outdated now, but in any case it shows that already in 1976 quite a large number of approximation results were known.

While many approximation algorithms were discovered, lower bounds seemed to be extremely difficult to come by. The situation changed radically in the last few years with the advent of the PCP theorem, to be discussed in § 4. In the rest of this section we list some of the most interesting early lower bounds which do not rely on it.

S. K. Sahni, T. F. Gonzalez (1976). P-complete approximation problems. *J. ACM* 23, 555–565.

The quest for lower bounds starts with this paper, where hardness results under the hypothesis P $\neq$ NP are given for several problems, among them the travelling salesperson problem.

M. R. Garey, D. S. Johnson (1976). The complexity of near-optimal graph coloring.

J. ACM 23, 43–49.

Among the first lower bounds is the result that graph coloring cannot be approximated within a factor better than 2, unless P = NP. This result was unthreatened for almost 20 years, until dramatically harsher lower bounds were found using the new PCP theorem, see §4.

Much research on lower bounds has focused on the clique problem, not only in the case of approximation. For instance, a famous result of Razborov states that monotone (without NOT-gates) boolean circuits that compute the clique function must have super-polynomial size. (See, for instance, chapter 14 in the Handbook of Theoretical Computer Science, volume A, Elsevier, Amsterdam). But the clique problem has also been central to the study of approximation algorithms. [Garey and Johnson 1979] (see §1.1) contains a nice proof of a puzzling result known as the self-improvability of MAX CLIQUE: this problem is either non-approximable within any constant or it is approximable within all constants. Although we now probably know the ultimate answer concerning the non-approximability of MAX CLIQUE (see §4), there are some very interesting lower bound results whose proof does not make use of the PCP theorem.

P. Berman, G. Schnitger (1992). On the complexity of approximating the independent set problem. *Inform. and Comput.* 96, 77–94.

By making use of randomized reductions, this paper establishes an interesting connection between MAX 3-SAT and MAX CLIQUE: even an $O(n^\delta)$-approximation algorithm for MAX CLIQUE (for some small enough $1 > \delta > 0$) would yield a *randomized ptas* for MAX 3-SAT.

N. Alon, U. Feige, A. Wigderson, D. Zuckerman (1995). Derandomized graph products. *Computational Complexity* 5, 60–75.

Here, the construction of [Berman and Schnitger 1992] is derandomized by making use of *expander graphs*, to which we return in §4. Thanks to the derandomization, the result can be restated as: there exists $\delta > 0$ such that, if MAX CLIQUE is approximable within $O(n^\delta)$ then MAX 3-SAT is in PTAS.

An important class of problems for which lower bounds could be shown using pre-PCP methods is that of scheduling problems. See the chapter "Nasty Gaps" in the survey of [Shmoys 1995] (see §1.2). Among papers containing early lower bound results the following is worth mentioning:

D. S. Hochbaum, D. B. Shmoys (1986). A unified approach to approximation algorithms for bottleneck problems. *J. ACM* 33, 533–550.

Very nice matching lower and upper bounds are shown for several natural problems in APX including MIN K-CLUSTERING, MIN K-CENTER, MIN K-SUPPLIER, and MIN K-SWITCHING NETWORK. See also the references cited there for a few more hardness results.

In spite of these early efforts, the field of approximation algorithm was characterized by a lack of a unifying framework and was made of many scattered, albeit clever, results. The situation was to change with the introduction of the notion of MAX SNP-hardness, which successfully captured a large class of problems. The PCP theorem later showed that none of these problems are in PTAS, unless P = NP. These two important developments are the topic of the next sections.

3 Structural Results

A successful way to characterize a class is to give a set of intuitive, "easy–to–use", necessary and/or sufficient conditions for membership in the class. In this respect, the notion of NP-completeness can be considered quite successful, as in many cases one can "smell" that a problem is NP-complete and quickly prove it. As far as approximation is concerned, the quest for intuitively compelling characterizations and easy–to–use tools of problem classification started in the late seventies.

M. R. Garey, D. S. Johnson (1978). Strong NP-completeness results: Motivation, examples, and implications. *J. ACM* 25, 499–508.

This paper gives a very successful characterization of the class Fptas by formalizing the idea that some problems, e.g. knapsack, are NP-complete only if exponentially large weights are allowed in the input and become polynomially solvable whenever the weights are polynomial in the input size. Such problems are called solvable in *pseudo-polynomial time*. The complementary class is comprised of problems whose NP-completeness does not depend on the presence of large weights, e.g. clique or the travelling salesperson problem. Problems of this kind are called *strongly NP-complete*. Garey and Johnson prove that, if the optimum value $opt_F(I)$ is polynomially bounded in both the number of objects in the input and the largest weight, and the corresponding decision problem is strongly NP-complete, then F cannot be in Fptas (unless P = NP). These assumptions on the input are satisfied by all known natural examples. This implies that if a problem with bounded optimum value is in Fptas then it must be solvable in pseudo-polynomial time.

A. Paz, S. Moran (1981). Non deterministic polynomial optimization problems and their approximations. *Theoret. Comput. Sci.* 15, 251–277.

Under assumptions verified by many natural problems, this paper shows the converse: if a problem is solvable in pseudo-polynomial time then it is in Fptas. An attempt to characterize Ptas is also done, and perhaps the first approximation preserving reduction is defined.

G. Ausiello, A. D'Atri, M. Protasi (1980). Structure preserving reductions among convex optimization problems. *J. Comput. Syst. Sci.* 21, 136–153.

A more restrictive type of reduction preserving also some structure of the solution is introduced.

In spite of these early efforts the question of how to prove that a problem is not in Ptas or Apx was as open as before. A tempting way to proceed is to introduce approximation complexity classes and to find problems that are complete in these classes (with respect to suitable reductions), mimicking the complexity theory for decision problems.

P. Orponen, H. Mannila (1987). *On approximation preserving reductions: Complete problems and robust measures*, Technical Report C-1987-28, Department of Comput.

Sci., University of Helsinki.

In this much underappreciated paper some approximation preserving reductions are introduced, and the travelling salesperson problem is shown to be complete in the class of NP minimization problems.

P. Crescenzi, A. Panconesi (1991). Completeness in approximation classes. *Inform. and Comput.* 93, 241–262.

This paper extends the approach of Orponen and Mannila to the classes APX and PTAS and shows that several problems, none of them very natural though, are complete in these classes. Moreover, it shows the existence of intermediate (non-complete) problems in these classes.

V. Kann (1994). Polynomially bounded minimization problems that are hard to approximate. *Nordic J. Comput.* 1, 317–331. Presented at ICALP '93.

Here, natural complete problems for MIN NPO PB, the class of minimization problems whose objective function is bounded by a polynomial in the size of the input, are found. The cited [Berman and Schnitger 1992] (see §2) contains similar results for MAX NPO PB.

A turning point in this line of research was reached thanks to this seminal paper:

C. H. Papadimitriou, M. Yannakakis (1991). Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.* 43, 425–440. See also STOC '88: 229–234.

In an attempt to avoid the difficulty of capturing the notion of *approximately correct computation*, Papadimitriou and Yannakakis use a logical (machine independent) characterization of NP given by Fagin. They define a class of problems called MAX SNP and show that MAX SNP $\subseteq$ APX. Moreover, they show that many natural problems such as MAX 3-SAT, and MAX CUT are MAX SNP-complete under an approximation preserving reduction called the L-reduction. The class MAX SNP was introduced in an attempt to group together APX problems that seemed (and were conjectured) not to be in PTAS. The definition of MAX SNP-hardness is crafted in such a way that if one could show that just one MAX SNP-complete problem does not belong to PTAS it would follow automatically that no MAX SNP-hard problem does. This is precisely what the PCP theorem does (see §4); it shows that one such problem, MAX 3-SAT, does not belong to PTAS. Therefore, no MAX SNP-hard problem is in PTAS (unless P = NP).

That MAX SNP captures the "right" level of completeness is attested to the fact that dozens of problems have been shown to be MAX SNP-complete or MAX SNP-hard (see the problem list [Crescenzi and Kann 1995] in §1.1). This notion is so successful that nowadays standard MAX SNP-completeness results are considered routine and are therefore rejected by journals!

The definition of MAX SNP via logic raised the possibility of an unexpected connection between logic and approximation. This problem has been investigated rather thoroughly:

A. Panconesi, D. Ranjan (1993). Quantifiers and approximation. *Theoret. Comput. Sci.* 107, 145–163. See also STOC '90: 446–456.

This paper defines other logical classes and shows completeness of many natural problems for them, e.g. MAX CLIQUE and MAX SET PACKING. It also shows by simple model theoretic arguments that, under some natural assumptions on the encoding, lots of natural optimization problems, e.g. MAXIMUM MATCHING, are not in MAX SNP (*without* any complexity theoretic assumption). This shows that MAX SNP does not contain P. This disappointing fact is however to be expected. If one could find a logically defined class containing P and prove by model theoretic arguments that it is a proper subclass of NP optimization problems, one would have separated P from NP.

P. G. Kolaitis, M. N. Thakur (1994). Logical definability of NP optimization problems. *Inform. and Comput.* 115, 321–353.
P. G. Kolaitis, M. N. Thakur (1995). Approximation properties of NP minimization classes. *J. Comput. Syst. Sci.* 50, 391–411.
T. Behrendt, K. J. Compton, E. Grädel (1993). Optimization problems: expressibility, approximation properties and expected asymptotic growth of optimal solutions. *Proc. 6th Workshop Comput. Sci. Logic*, Berlin. Springer-Verlag, 43–60. Lecture Notes Comput. Sci. 702.

The first two papers systematically investigate the relationship between logic and approximation for, respectively, maximization and minimization problems. Unfortunately the relationship between syntax and approximation seem to be rather weak in the latter case. The third paper extends the logical approach by introducing more sophisticated logical notions such as fix point logic.

S. Khanna, R. Motwani (1996). Towards a syntactic characterization of PTAS. *Proc. Twenty eighth Annual ACM Symp. Theory of Comput.*, 329–337.

Here Khanna and Motwani find a way to capture a large chunk of PTAS via a logical characterization. The idea is to specify a problem syntactically in such a way that the input instances exhibit a planar structure so that powerful graph separator theorems can be invoked to derive *ptas*'s in a uniform manner (i.e. for all problems in the class).

N. Creignou (1995). A dichotomy theorem for maximum generalized satisfiability problems. *J. Comput. Syst. Sci.* 51, 511–522.
S. Khanna, M. Sudan, D. P. Williamson (1997). A complete classification of the approximability of maximization problems derived from boolean constraint satisfaction. *Proc. 29th Annual ACM Symp. Theory of Comput.* To appear.
S. Khanna, M. Sudan, L. Trevisan (1997). Constraint satisfaction: the approximability of minimization problems. *Proc. 12th Annual IEEE Conf. Comput. Complexity.* To appear.

These recent papers show that the logical approach is quite successful for the restricted but rich class of constraint satisfaction problems (CSP's). Interestingly, they show a restatement of an earlier result by Schaefer in the context approximation (see Schaefer (1978), The complexity of satisfiability problems, STOC '78, 216–226). Namely, these classes are neatly divided into a finite number of levels, i.e. a CSP problem can only have a certain (finite) number of behaviours w.r.t. approximation.

Returning to MAX SNP we can see that rather than MAX SNP itself the class of interest is the *closure* of MAX SNP w.r.t. approximation preserving reductions, i.e. the set of problems that are reducible to some MAX SNP problem. Several papers investigate the closure of MAX SNP w.r.t. various reducibilities:

S. Khanna, R. Motwani, M. Sudan, U. Vazirani (1994). On syntactic versus computational views of approximability. *Proc. 35th Annual IEEE Symp. Found. Comput. Sci.*, 819–830. Full version available as ECCC Report TR95-023, `ftp://ftp.eccc.uni-trier.de/pub/eccc/reports/1995/TR95-023/index.html`.

Surprisingly, using a slightly less restrictive notion of approximation preserving reduction than the L-reduction, called E-reduction, this closure turns out to be the whole of APX PB—the class of approximable problems whose objective function is polynomially bounded. This means, for example, that MAX 3-SAT is APX PB-complete. The proof relies on the PCP theorem (see §4).

P. Crescenzi, L. Trevisan (1994). On approximation scheme preserving reducibility and its applications. *Proc. 14th Annual Conf. Found. Software Tech. and Theoret. Comput. Sci.*, Berlin. Springer-Verlag, 330–341. Lecture Notes Comput. Sci. 880.

Using a less restrictive version of the E-reduction this paper finally captures the whole of APX as the closure of MAX SNP. Thus MAX SNP captures the core of hardest problems in APX, and therefore we can now prove APX-completeness, as opposed to just MAX SNP-completeness, for many natural problems.

In a different line of research, people have used tools from structural complexity theory, which can be roughly described as an offspring of Recursion Theory, to investigate approximation properties:

P. Crescenzi, V. Kann, R. Silvestri, L. Trevisan (1995). Structure in approximation classes. *Proc. 1st Annual Int. Conf. Comput. and Combin.*, Berlin. Springer-Verlag, 539–548. Lecture Notes Comput. Sci. 959.

Among other things, this paper shows that if an optimization problem has an *asymptotic ptas*, then it cannot be APX-complete under the complexity theoretic assumption that the Polynomial Hierachy (PH) does not collapse, a fact which is widely conjectured to be true. (An asymptotic *ptas* for, say, a minimization problem F is an infinite collection $\{A_\epsilon\}$ of polynomial time algorithms such that there exists a constant k such that, for all inputs I and $\epsilon > 0$, $A_\epsilon(I) \leq \epsilon opt_F(I) + k$). Two well-known problems which have asymptotic *ptas*'s are MIN BIN PACKING and MIN EDGE COLORING. This result shows the existence of natural intermediate problems within approximation classes (unless the PH collapses). The result is established by using connections between the approximability properties and the *query complexity* (the number of queries to an NP oracle).

Relating the query complexity to approximation has given rise to several other interesting papers. For instance, it is possible to show that, under suitable complexity theoretic assumptions, computing an approximate feasible solution is harder than just computing an approximate numerical value. The paper:

R. Chang (October 1994). The structural complexity column: A machine model for NP-approximation problems and the revenge of the boolean hierarchy. *Bulletin of the EATCS*, 54, 166–182.

... contains an excellent discussion of these results with the relevant bibliography.

4 The PCP Revolution

The recent dramatic developments in the theory of approximation are based on the so-called PCP theorem, which establishes an unexpected connection between interactive proofs and approximation algorithms. The PCP theorem is a deep result in many ways. From a technical point of view, the result makes use of new algebraic techniques for the testing of polynomial identities and combines them with a host of tools brought from diverse areas within and outside of theoretical computer science, such as interactive proof systems, self testing and correcting of polynomials, mathematical logic and coding theory. This result also intensifies the philosophical debate concerning the role of randomness in computing. Moreover, as we shall see, the result has far reaching consequences in the area of approximation algorithms.

S. Goldwasser, S. Micali, C. Rackoff (1989). The knowledge complexity of interactive proof-systems. *SIAM J. Comput.* 18, 186–208. See also STOC '85: 291–304.
L. Babai, S. Moran (1988). Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.* 36, 254–276. See also STOC '85: 421–429.

Unexpectedly, the paradigm that gave birth to the PCP theorem is the seemingly unrelated field of interactive proof systems which was introduced at about the same time in the first of these papers in the context of cryptography, and in the second as a game theoretic extension of NP. Two insightful and readable accounts describing the role of interactive proofs in the development of the PCP and related theorems are [Johnson 1992] and [Babai 1994] (see §1.2).

In order to understand the implications for approximation, a brief description of a probabilistic checker of proof is in order. One way to understand the class NP is that it deals with statements (theorems), made in some formal system, whose proof might be computationally hard to come by, but which has fast verification procedures of correctness. The typical example is Satisfiability: deciding whether a given formula is satisfiable might take exponential time in the length of the formula, but to verify that a given truth assignment satisfies it can be quickly checked in polynomial time. In this context, a satisfying assignment constitutes the proof that the formula is satisfiable.

Suppose now that in order to decide whether a given bit string is indeed a correct proof of a statement "$x \in L$", for some language $L \in$ NP, we do not want to look at the whole proof. Rather, we would like to perform a few random checks and then conclude with reasonable confidence that the proof is correct. This bold plan can be formalized as follows. An $(r(n), q(n))$-verifier for a language L is a Turing machine V, whose task is to determine the correctness of *alleged proofs* of statements of the form "$x \in L$". The machine is given the input x and an alleged proof of the statement. The alleged proof is contained in a large *database* D, which is simply a bit string. The verifier must work in time polynomial in $n = |x|$, but can take advantage of two extra computational resources: (a) $r(n)$-many *random bits* and (b) $q(n)$-many accesses to the database D, i.e. to the bits of the alleged proof. Intuitively, the random bits are used to generate random addresses of the $2^{r(n)}$-many database locations. This allows the verifier to sample the alleged proof at random places. The verifier must try to check whether the purported proof contained in the database is correct, in which case it accepts x. If it is incorrect the verifier rejects (regardless of whether $x \in L$). More precisely, we say that the verifier accepts a language L if the following two conditions are met:

- $x \in L \Rightarrow \exists D \ \Pr[V(D,x) \text{ accepts}] = 1$, i.e. when the proof is correct the verifier accepts for all random strings.
- $x \notin L \Rightarrow \forall D \ \Pr[V(D,x) \text{ accepts}] \leq 1/2$, i.e. the probability that the verifier accepts a fallacious proof is small.

Note that by repeating the verification procedure k times, where k is a constant, the error probability becomes as small as 2^{-k}.

The class $\mathrm{PCP}(r(n), q(n))$ is the class of languages accepted by some $(O(r(n)), O(q(n)))$-verifier. The PCP theorem states that

$$\mathrm{PCP}(\log n, 1) = \mathrm{NP}.$$

This result establishes the amazing fact that for all languages in NP there is a way of writing proofs (databases) such that a verifier need only probe a *constant number* of locations in order to conclude with error probability 1/2 whether the proof is correct or wrong. We now know that probing 9 bits suffices (see [Håstad 1997] cited at the end of §4)! Moreover, by running the protocol a few times the verifier can boost this probability to $1 - o(1)$.

What does the PCP theorem have to do with optimization? Given a verifier V for a language $L \in \mathrm{NP}$, the following optimization problem can be defined: on input x, the feasible solutions are all possible, exponentially many, databases $D \in \{0,1\}^{2^{r(n)}}$; the value of the objective function on a given feasible solution D is the number of random strings $r \in \{0,1\}^{r(n)}$ for which the verifier accepts. The optimum of this problem is

$$opt_V(x) = \max_D |\{r : V(D,x) \text{ accepts if the random string is } r\}|.$$

This is an NP optimization problem, because, crucially, $r(n) = O(\log n)$. This implies that the feasible solutions are polynomial size bit strings, and that the objective function is polynomially computable. Indeed, R, the total number of random strings, is $2^{r(n)}$—a polynomial—and checking each one of them for acceptance requires simulating the verifier, which can be done in polynomial time. Note now that this problem exhibits a *gap*: if the formula is satisfiable then $opt_V(x) = R$. Otherwise, $opt_V(x) \leq R/2$. It follows that no polynomial time algorithm can approximate the problem within a factor smaller than 2, unless P = NP. Indeed, if $x \notin L$, such an algorithm must deliver a value $\leq R/2$ whereas, if $x \in L$, the algorithm must deliver a value $> R/2$. Hence, one can use the approximation algorithm to decide whether $x \in L$, where L can be any NP-complete language.

A crucial fact about the PCP theorem is that it makes it possible to create instances of natural optimization problems (as opposed to artificial examples like the one above) that exhibit such gaps, the construction of which eluded researchers for about two decades.

An interesting fact about the PCP theorem, that might be relevant for sociologists of science and of the information age, is that it is a collective achievement. The number of people that authored or co-authored papers containing substantial or crucial steps toward the complete solution is more than a dozen. Scientific progress has always relied on constant interaction among people, but the rate of exchange and personal contacts now made possible by the Pentagon-supported information and aviation revolution confronts us with a seemingly new sociological phenomenon in the realm of science. (See for instance, Babai (1990), E-mail and the unexpected power of interaction, *Proc. 5th IEEE Symp. on Structure in Complexity Theory*, 30–44.) In what follows, we do not attempt to sort out the different contributions. Rather, we list those papers that seem to us more directly relevant for their consequences to approximation.

A. Condon (1993). The complexity of the max-word problem and the power of one-way interactive proof systems. *Computational Complexity* 3, 292–305.

This is the first paper to show a connection between interactive proofs and approximation, but the turning point was a paper that appeared a few months later.

U. Feige, S. Goldwasser, L. Lovász, S. Safra, M. Szegedy (1991). Approximating clique is almost NP-complete. *Proc. 32nd Annual IEEE Symp. Found. Comput. Sci.*, 2–12.

The connection between interactive proofs and approximation is here clearly shown by exhibiting a crisp reduction between any $(r(n), q(n))$-verifier and the clique problem. The reduction establishes a bijection between databases for which the verifier accepts on k different random strings and cliques of size k. The reduction is so clean that it can be explained in a few lines. See for example the surveys of [Johnson 1992], [Håstad 1994], and [Shmoys 1995] cited in §1.2.

L. Babai, L. Fortnow, C. Lund (1991). Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity* 1, 3–40. See also FOCS '90: 16–25.

This paper contains the beautiful result which motivated the Feige et al. paper. Namely, $\mathrm{PCP}(p(n), p(n)) = \textsc{Nexptime}$, where $p(n)$ is a polynomial. The Feige et al. paper "scales down" this construction from $\textsc{Nexptime}$ to NP to get $\mathrm{PCP}(c(n), c(n)) \supseteq \mathrm{NP}$, with $c(n) = \log n \log\log n$. Together with the clique reduction, this implies that clique is not approximable within any constant unless $\mathrm{NP} \subset \textsc{Dtime}[n^{\log\log n}]$, the class of languages that can be recognized in time $O(n^{\log\log n})$ by a deterministic Turing machine, a fact that appears unlikely and is conjectured to be false.

These papers do not explicitly frame their result in terms of PCP. This class was formally introduced in a subsequent paper:

S. Arora, S. Safra (1992). Probabilistic checking of proofs; a new characterization of NP. *Proc. 33rd Annual IEEE Symp. Found. Comput. Sci.*, 2–13.

This breakthrough paper introduces the class PCP and shows that $\mathrm{PCP}(\log n, \log n) = \mathrm{NP}$, thereby solving a 20 year old open problem: approximating clique within any constant is NP-hard. Like many papers in the area, it is a technical *tour-de-force.* Its main technical contribution is probably a sophisticated recursive procedure to check polynomial identities very quickly and with high confidence.

Another important contribution of the Arora and Safra paper is the use of *expander graphs* to boost the hardness result for clique. Using later improvements, this technique eventually yields the following: there exists an ϵ such that approximating clique within n^ϵ is NP-hard.

Expanders seem to play an important role in the derivation of hardness results and have already proven to be extremely useful in other areas of computer science such as parallel architectures (a celebrated result of Ajtai–Komlós–Szemerédi showing the existence of small depth sorting networks is based on the existence of such graphs, see for instance, chapter 15 in the Handbook of Theoretical Computer Science, volume A, Elsevier, Amsterdam). A nice introduction to expanders is contained in the book by Alon and Spencer (1991), The probabilistic method, Wiley & Sons., New York. As far as hardness results are concerned, expanders were used in the seminal paper of Papadimitriou and Yannakakis (see §3) to show MAX SNP–hardness results of several problems, including MAX 2-SAT, MAX CUT and MAX 3-SAT-B, the variant of MAX SAT where each variable appears at most B times. Two more interesting results using expanders are:

E. Amaldi, V. Kann (1995). The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoret. Comput. Sci.* 147, 181–210.
P. Crescenzi, R. Silvestri, L. Trevisan (1996). To weight or not to weight: Where is the question? *Proc. 4th Israel Symp. Theory Comput. and Syst.* IEEE, 68–77.

In the first of the two, expanders are used to prove an n^ϵ hardness bound for approximating the feasibility problem for systems of linear equations (given a set of n linear equations, what is the largest subset of feasible equations?). The second paper shows that, for a reasonably large class of optimization problems, the approximation threshold for the weighted and unweighted case is the same. Roughly speaking, the class in question consists of those problems for which the feasibility of a solution does not depend on its weight and includes, among others, MAX CUT and MAX SAT.

After the $\mathrm{PCP}(\log n, \log n) = \mathrm{NP}$ result by Arora and Safra, it was observed that a gap preserving reduction exists between any $(r(n), q(n))$-verifier and 3-SAT formulas. Given a verifier V and an input x, the reduction produces a boolean formula $\varphi_{V,x}$ of size $O(p(n)Q(n))$, where $|x| = n$, $p(n)$ is a polynomial and $Q(n) = O(2^{2^{q(n)}})$, such that: (a) if $x \in L$ then all clauses of $\varphi_{V,x}$ are satisfiable; and (b) if $x \in L$ then at most a $1 - 1/Q(n)$ fraction of the clauses of $\varphi_{V,x}$ is satisfiable (The reduction is not hard and can be found in the cited [Shmoys 1995] and [Johnson 1992], see §1.2).

This suggests a bizarre but exciting possibility: if one could bring the number of queries $q(n)$ down to a constant, a (roughly) $(1 + 1/Q(n))$-hardness result for MAX 3-SAT would follow because $Q(n)$ would be huge, but constant. Sure enough, after a few months the result was announced in this paper:

S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy (1992). Proof verification and hardness of approximation problems. *Proc. 33rd Annual IEEE Symp. Found. Comput. Sci.*, 14–23.

This paper puts the last nail in the coffin of the PCP theorem, which states that:

$$\mathrm{PCP}(\log n, 1) = \mathrm{NP}$$

The consequences of this characterization of NP are momentous. Recall that after Papadimitriou and Yannakakis introduced the class MAX SNP dozens of problems, including MAX 3-SAT, were shown MAX SNP-complete w.r.t. L-reductions. The PCP theorem shows that MAX 3-SAT is not in PTAS (unless P = NP) which implies that no MAX SNP-hard problem is in PTAS. In particular, now we have a very effective and easy–to–use tool to show dozens of hardness results: exhibit an approximation preserving reduction from a problem known to be MAX SNP-hard to the problem at hand. This task often turns out to be just slightly more difficult than an average NP-completeness proof.

The complete proof of the PCP theorem can be found in this important PhD thesis of Sanjeev Arora:

S. Arora (1994). *Probabilistic Checking of Proofs and Hardness of Approximation Problems*, Technical Report CS-TR-476-94, Department of Comput. Sci., Princeton University. Revised version available as ECCC Book 1995, `http://www.eccc.uni-trier.de/eccc-local/ECCC-Books/sanjeev_book_readme.html`.

This thesis contains a clear and detailed exposition of the proof of the PCP theorem, written by one of its foremost contributors. Because of its thoroughness, this is the best reference for those interested in studying this long and difficult proof.

An important ingredient in the proof of the PCP theorem is the use of techniques for checking polynomial identities developed in the context of the theory of self-testing and correcting. The original motivation of this theory was to give a satisfactory answer, at least partially, to one of the most fundamental problems of computer science: how can we effectively and feasibly check that a given program computes what it is supposed to compute?

A very nice discussion of the relationship between self-testing and correcting and the PCP theorem can be found in this wonderful PhD thesis:

M. Sudan (1995). *Efficient checking of polynomials and proofs and the hardness of approximation problems*, Springer-Verlag, Berlin. Lecture Notes Comput. Sci. 1001.

This thesis contains a description and the proof of the PCP theorem from a different angle. It is, however, not as complete as that contained in the thesis of Arora. This thesis is also a good starting point for those who want to venture into the theory of self-testing and correcting.

S. Hougardy, H. J. Prömel, A. Steger (1994). Probabilistically checkable proofs and their consequences for approximation algorithms. *Disc. Math.* 136, 175–223.

This is a survey containing a proof of the PCP theorem and a discussion of some of its applications to approximation. It is important to stress that the proof contained in the paper is not the authors' (so far there is a single known proof of the PCP theorem). This survey was intended to publicize these important results within the discrete mathematics community by presenting in an organized and compact fashion what at the time was only available in scattered and fragmented sources. Since the authors had to rely on incomplete materials such as drafts and conference abstracts, their work can be considered as an independent verification of this difficult proof. In this sense it represents a valuable service to the community.

After the Arora et al. paper, problems that had been open for years continued to fall one after the other.

C. Lund, M. Yannakakis (1994). On the hardness of approximating minimization problems. *J. ACM* 41, 960–981. See also STOC '93: 286–293.

In this beautiful paper two long standing open problems are solved. The first result is a gap preserving reduction between clique and chromatic number which shows that there exists a $\delta > 0$ such that $\chi(G)$ is not approximable within $O(n^\delta)$ unless P = NP—quite a dramatic improvement from the previous best lower bound of 2 (see §2).

The second result is a proof that MIN SET COVERING cannot be approximated better than $(1/4)\log n$, unless NP $\subseteq$ $\textsc{Dtime}[2^{\mathrm{polylog}(n)}]$. This result is important in many respects. First, quite magically, it matches the upper bound of $O(\log n)$ given by the greedy algorithm (see for instance Chvatál (1979), A greedy heuristic for the set covering problem, *Math. Oper. Res.* 4, 233–235). Second, rather than the PCP theorem, it uses a different characterization of NP due to Feige and Lovász in terms of so-called multiprover interactive proof systems, another construct developed in the context of cryptography.

S. Khanna, N. Linial, S. Safra (1993). On the hardness of approximating the chromatic number. *Proc. 2nd Israel Symp. Theory Comput. and Syst.* IEEE, 250–260.

Here, a very elegant and much simpler proof of the chromatic number lower bound of [Lund and Yannakakis 1994] can be found.

M. Bellare (1993). Interactive proofs and approximation: reductions from two provers in one round. *Proc. 2nd Israel Symp. Theory Comput. and Syst.* IEEE, 266–274.

Although this paper does not contain any big improvements, it was perhaps the first to notice a fact fully exploited in [Lund and Yannakakis 1994], namely that some interactive proof characterizations of NP can be better suited than others for the problem at hand.

The first hardness results using the new methods had an existential flavour. For instance, the result for MAX CLIQUE stated that: there exists *some* $\delta > 0$ such that $\omega(G)$ is not approximable within n^δ. The value of δ was small and hard to compute. The next step, quite naturally, was to look into the PCP technology and try to make some parts of this complex machinery more efficient.

M. Bellare, S. Goldwasser, C. Lund, A. Russell (1993). Efficient probabilistically checkable proofs and applications to approximation. *Proc. Twenty fifth Annual ACM Symp. Theory of Comput.*, 294–304.

This paper initiates what is to this day the current trend. By looking at PCP parameters like average query complexity, query size, answer size, etc. Bellare et al. were able to prove hardness results for quite reasonable constants. For instance, for MAX CLIQUE they get $n^{1/30}$.

A stream of highly technical papers introduced and put to fruition more sophisticated parameters such as free bits, average free bits and the like, obtaining better and better constants.

M. Bellare, O. Goldreich, M. Sudan (1995). Free bits, PCPs and non-approximability – towards tight results. *Proc. 36th Annual IEEE Symp. Found. Comput. Sci.*, 422–431. Full version available as ECCC Report TR95-024,

ftp://ftp.eccc.uni-trier.de/pub/eccc/reports/1995/TR95-024/index.html.

This treatise systematizes and improves upon the most recent developments and gives stronger hardness bounds than ever before for several problems. It also contains a detailed and enjoyable history of the various contributions.

After this stream of results we are now in a position unimaginable a few years back: the best known lower and upper bounds for some fundamental combinatorial problems are not too far apart. For instance, for the MAX 3-SAT problem the best algorithm delivers a solution whose value is about 80.0 % of the optimum whereas no algorithm can deliver more than 87.5 % of the optimum (unless P = NP). For MAX CUT the bounds are, respectively, 87.8 % and 94.2 %.

One feels hesitant in stating these results because the pace at which improvements are taking place can make these statements obsolete. For instance, Feige has recently announced a $(1-\epsilon)\ln n$ lower bound for MIN SET COVERING, for any $\epsilon > 0$, under the improved (weaker) complexity theoretic assumption NP $\not\subseteq$ DTIME$[n^{O(\log\log n)}]$. For MAX CLIQUE, the lower bound increased rapidly, going from n^{ϵ} to $n^{1/30}, n^{1/6}, n^{1/5}, n^{1/4}, n^{1/3}$ until Håstad announced recently a $n^{1/2-o(1)}$ lower bound under the hypothesis NP $\not\subseteq$ coRP (i.e. NP-complete problem cannot be solved by randomized polynomial time algorithms). By the time this paper was published– it was presented at STOC '96 together with Feige's set covering result– it was already obsolete!

J. Håstad (1996). Clique is hard to approximate within $n^{1-\epsilon}$. *Proc. 37th Annual IEEE Symp. Found. Comput. Sci.*, 627–636.

U. Feige, J. Kilian (1996). Zero knowledge and the chromatic number. *Proc. 11th Annual IEEE Conf. Comput. Complexity*, 278–287.

Here Håstad announced the stunning tight hardness result of $n^{1-o(1)}$ for the MAX CLIQUE problem under the assumption NP $\not\subseteq$ coRP. In turn, this was immediately used by Feige and Kilian to give a tight $n^{1-o(1)}$ lower bound for chromatic number.

As for 1997, we can look forward to this result:

J. Håstad (1997). Some optimal inapproximability results. *Proc. 29th Annual ACM Symp. Theory of Comput.* To appear.

Here matching lower bounds for MAX Ek-SAT (the variant of MAX SAT where each clause has exactly k literals) are presented together with improved lower bounds for, among others, MAX CUT and MIN VERTEX COVER.

And more is yet to come...

Acknowledgements

We would like to thank the following persons for useful comments on the text: Gunnar Andersson, Pierluigi Crescenzi, Devdatt Dubhashi, Thomas Emden-Weinert, Lars Engebretsen, Johan Håstad, Stefan Hougardy, Sanjeev Khanna, Hans Jurgen Prömel, David Shmoys, Aravind Srinivasan, Madhu Sudan, Luca Trevisan.