Max NP-completeness Made Easy

Pierluigi Crescenzi* Luca Trevisan†

December 13, 1996

Abstract

We introduce a simple technique to obtain reductions between optimization constraint satisfaction problems. The technique uses the probabilistic method to reduce the size of disjunctions. As a first application, we prove the Max NP-completeness of MAX 3SAT without using the PCP theorem (thus solving an open question posed in Khanna et al. (1994)). Successively, we show that the "planar" restrictions of several optimization constraint satisfaction problems admit linear-time approximation schemes (thus improving the results of Khanna and Motwani (1996)).

1 Introduction

Comparing the complexity of different combinatorial optimization problems has been an extremely active research area during the last 25 years. Roughly speaking, the question is: given two optimization problems A and B, is solving A more (less, equally) difficult than solving B? Starting from the deep background in computability theory, researchers first attacked this question by shifting their attention from the original optimization problems to their corresponding decision versions: the NP-completeness theory eventually arose and hundreds and hundreds of combinatorial problems were shown to be equivalently hard to be solved [GJ79]. Almost at the same time, however, it was noticed that, even though all known NP-complete problems are polynomial-time isomorphic, their corresponding optimization problems may behave in a drastically different way when dealing with approximate solutions [Joh74]. As a result, the development of approximation algorithms (that is, algorithms yielding a solution whose cost is within a multiplicative constant factor from the optimum) immediately revealed itself as a useful tool to cope with the NP-hardness of a combinatorial optimization problem.

The original question was then refined: given two optimization problems A and B, is *approximately* solving A more (less, equally) difficult than *approximately* solving B? In order to answer this question, basically two interleaving approaches have been followed. On the one hand, researchers have studied which kind of reducibility is suitable to compare the approximation properties of optimization problems [ADP80, PM81, OM87, Sim89, CP91, PY91, KMSV94, CT94, CKST95]. It was clear, indeed, that the many-to-one polynomial-time reducibility could not be used since, not only a function mapping instances into instances is necessary, but also a function mapping solutions into solutions. On the other hand, more and more sophisticated techniques to obtain reductions

*Dipartimento di Scienze dell'Informazione, Università di Roma "La Sapienza", Via Salaria 113, 00198 Roma, Italy. E-mail: `piluc@dsi.uniroma.it`.

†Centre Universitaire d'Informatique, Université de Genève, Rue Général-Dufour 24, CH-1211, Genève, Switzerland. E-mail: `trevisan@cui.unige.ch`. Work done at the Università di Roma "La Sapienza".

were developed passing through the use of expander graphs [PY91] and arriving at the extremely complicated and powerful toolkit of probabilistically checkable proofs [KMSV94].

Among the different proposals of approximation preserving reducibilities, the L-reducibility [PY91] can certainly be considered the most popular since, in a certain sense, it is the most simple and natural. However, more sophisticated definitions are necessary to obtain completeness results in natural approximation classes [KMSV94, CT94]. Moreover, the simplicity of the L-reducibility forces the "reducer" to use complicated tools. A complicated reduction between two optimization problems has several disdvantages. First, (needless to say) it is difficult to be explained and to be checked. Second, it usually hides the relationship between the combinatorial structure of the two problems. Third, it rarely can be used on a positive side, that is, in order to obtain improved algorithmic results.

The aim of this paper is to introduce a simple technique to obtain reductions between optimization constraint satisfaction problems. The simplicity of the technique will allow us both to clarify already known structural results and to obtain new algorithmic ones. Before stating our results, we recall some previous ones about completeness in approximation classes.

The formal definition of approximation algorithm and performance ratio is due to [Joh74]. Shortly after, several approximation classes were defined, including APX (that is, the class of problems that are approximable within *some* constant factor) and PTAS (that is, the class of problems that are approximable within *any* constant factor). In a few years the field developed rapidly on the algorithmic side but only a few, unsatisfying, hardness results came up, until a novel approach was introduced in [PY91]. In this paper, the authors, instead of searching for complete problems in natural (that is, computationally defined) approximation classes, focused on natural problems and found a reasonable class where they are complete. Their starting point was the logical characterization of NP [Fag74]. They obtained a class of APX optimization problems by introducing a notion of optimization in this characterization, called this class Max NP, and showed that several problems, including MAX SAT, belong to it. However, they could not find any complete problem. A restriction in the definition yields the subclass Max SNP, that still contains interesting problems such as MAX CUT and MAX 3SAT, and that turns out to have natural complete problems (including MAX CUT and MAX 3SAT themselves). The success of this approach was immediate and widespread. Other Max SNP-hard problems were soon discovered and other approximation classes with natural complete problems were introduced [PR93, KT94, KT95]. In [KMSV94] the logical approach was finally reconciled with the computational one. In fact, by making use of the PCP theorem [ALM+92], the authors proved that MAX 3SAT is APX-complete. As a corollary of this result, it follows that MAX 3SAT is Max NP-complete thus answering a question posed in [PY91].

Our Results

In this paper, we introduce a reduction technique based on the *probabilistic method* that allows to directly prove the Max NP-completeness of MAX 3SAT. In order to obtain this result we make use of a *more powerful reducibility* than L-reducibility, called PTAS-reducibility [CT94]. Indeed, our technique further exploits such additional power and does not seem to work when restricted to the L-reducibility. As expected, more powerful reducibilities may allow simpler reductions.

The relevance of our new Max NP-completeness proof is due to the fact that *it does not use the* PCP *machinery* (thus solving an open question posed in [KMSV94]). Moreover it has both structural and algorithmic consequences.

From a *structural* point of view, the Max NP-completeness of MAX 3SAT means that Max NP problems are not harder to approximate than Max SNP ones. The definition of Max NP involves a richer logical structure, that is, one more quantifier than Max SNP. The reduction based on the PCP theorem does not clearly explain how comes that it is possible to get rid of this additional quantifier. This is due to the fact that this reduction is *global* in a very strong sense, and, while reducing a logical problem to another, it is not clear which variables are mapped to which and which constraint is mapped to which: in a few words, there is no clear relation between the structure of the source problem and the structure of the target problem. In contrast, our reduction gives a very simple explanation of the relation between Max NP and Max SNP. Recall that, roughly speaking, the only difference between the two classes is that in the logical definition of a Max NP problem arbitrarily long disjunctions are allowed [PY91]. However, long disjunctions are easy to satisfy (in a probabilistic sense) and thus they cannot make the problem harder.

From an *algorithmic* point of view, an approximation-preserving reduction from a problem A to a problem B can yield an *approximation scheme* in the following case. Assume that problem B admits an approximation scheme when restricted to some subset $\mathcal{I}_B$ of its instances, and that it is possible to find a set $\mathcal{I}_A$ of instances of A such that any such instance is mapped by the reduction into an instance of $\mathcal{I}_B$. Then, the reduction together with the approximation scheme gives an approximation scheme for the instances of $\mathcal{I}_A$. Our reduction gives indeed *an approximation scheme for* planar *restrictions of* Max Sat *and* Max GSAT-B *that works in linear time* (as opposed to a previous scheme running in time $n^{O(1/\epsilon)}$ [KM96]).

Comparison with PCP-based Reductions. Our result would have been impossible to obtain by using PCP-based reductions for at least two reasons:

1. It is very difficult to characterize the outcome of a PCP-based reduction, and so to understand which set of instances will be mapped into which.
2. PCP-based reductions always generate instances that are hard to approximate, whatsoever was the simple structure of the source instance.

The above considerations suggest that a PCP-free proof that Max 3Sat is APX-complete would have very interesting structural and algorithmic consequences. We show that, unfortunately, at least a weaker version of the PCP theorem is necessary in order to prove the APX-completeness of Max 3Sat.

Comparison with Local-Replacement Reductions. The most common technique to reduce a constraint satisfaction problem to another is the *local-replacement* one [GJ79]. In particular, the best known non-approximability results for several problems, including Max 3Sat and Max Cut, are derived using reductions of this kind [BGS95, TSSW96, Hås96]. Deriving more "efficient" reductions would imply stronger (possibly tight) non-approximability results. Unfortunately, lower bounds for the efficiency of local-replacement reductions have been found [TSSW96]. In particular the reductions yielding the non-approximability results for Max 3Sat and Max Cut are optimal among local-replacement ones. We show that, in some significant cases, our reduction beats the known lower bounds and is thus *provably* better than any local-replacement reduction.

2 Preliminaries

We assume familiarity with the basic concepts of computational complexity theory. For the definitions of most of the complexity classes used in this paper we refer the reader to one of the books on the subject (see, for example, [GJ79, BaG88, BC93, Pap94]).

An *optimization problem* A consists of: (1) the set I of instances, (2) for any instance $x \in I$, a set $\mathsf{sol}(x)$ of solutions, and (3) for any instance $x \in I$ and for any solution $y \in \mathsf{sol}(x)$, a measure $\mathsf{m}(x,y)$. Solving an optimization problem means to find an optimum solution y for a given an instance x, that is, a solution whose measure is maximum or minimum depending on whether the problem is a maximization or a minimization one. In the following opt will denote the function that maps an instance x into the measure of an optimum solution. The optimization problems we deal with in this paper are defined in Section 2.1.

Definition 1 (Performance Ratio) *Let A be an optimization problem. For any instance x and for any solution $y \in \mathsf{sol}(x)$, the* performance ratio *of y with respect to x is defined as*

$$R(x,y) = \max\left\{\frac{\mathsf{opt}(x)}{\mathsf{m}(x,y)}, \frac{\mathsf{m}(x,y)}{\mathsf{opt}(x)}\right\} .$$

For a constant $r > 1$, we say that an algorithm T is *r-approximate* for an optimization problem A if, for any instance x, the performance ratio of the feasible solution $T(x)$ with respect to x is at most r. If a problem A admits a polynomial-time r-approximate algorithm for some constant $r > 1$, then we will say that A belongs to the *class* APX. An optimization problem A belongs to the *class* PTAS if a *polynomial-time approximation scheme* for A exists, that is, an algorithm T such that, for any fixed rational $r > 1$, $T(\cdot, r)$ is a polynomial-time r-approximate algorithm for A.

We refer to [PY91] (see also [KT94]) for a formal definition of Max NP and Max SNP. We here give informal but equivalent definitions. We say that A is a *subproblem* of B if $I_A \subseteq I_B$, and, for any $x \in I_A$, it holds $\mathsf{sol}_A(x) = \mathsf{sol}_B(x)$ and $\mathsf{m}_A(x,\cdot) = \mathsf{m}_B(x,\cdot)$.

Definition 2 (Max SNP and Max NP) *A maximization problem is in the class* Max SNP *if a constant k exists such that A can be expressed as a subproblem of* Max k-CSP. *A maximization problem is in the class* Max NP *if a constant B exists such that A can be expressed as a subproblem of* Max GSAT-B.

Definition 3 (PTAS-reducibility [CT94]) *Let A and B be two optimization problems. A is said to be* PTAS-reducible *to B, in symbols $A \leq_{\text{PTAS}} B$, if three functions f, g, and c exist such that:*

1. *For any rational $r > 1$, and for any $x \in I_A$, $f(x,r) \in I_B$ is computable in time $t_f(|x|, r)$.*
2. *For any rational $r > 1$, for any $x \in I_A$, and for any $y \in \mathsf{sol}(f(x,r))$, $g(x,y,r) \in \mathsf{sol}(x)$ is computable in time $t_g(|x|, |y|, r)$.*
3. *For any fixed r, both $t_f(\cdot, r)$ and $t_g(\cdot,\cdot,r)$ are bounded by a polynomial.*
4. *$c : (1,\infty) \cap Q^+ \to (1,\infty) \cap Q^+$ is a computable function.*
5. *For any rational $r > 1$, for any $x \in I_A$, and for any $y \in \mathsf{sol}_B(f(x,r))$,*

$$R_B(f(x,r),y) \leq c(r) \text{ implies } R_A(x, g(x,y,r)) \leq r .$$

The triple (f,g,c) is said to be a PTAS*-reduction from A to B.*

In [CT94] it is shown that if $A \leq_{\mathrm{PTAS}} B$ and $B \in \mathsf{PTAS}$, then $A \in \mathsf{PTAS}$.

Finally, we summarize the main definitions from the theory of probabilistically checkable proofs.

A *promise problem* is a pair (Y, N) of disjoint sets of strings from some fixed alphabet Σ (we can assume without loss of generality $\Sigma = \{0,1\}$). An algorithm *solves* a promise problem (Y, N) if, for any input string $x \in Y$, the algorithm accepts, and, for any input string $x \in N$, the algorithm rejects. The behaviour of the algorithm can be arbitrary when it receives in input a string from $\Sigma^* - (Y \cup N)$. A language L can be seen as the promise problem $(L, \Sigma^* - L)$.

A *verifier* is an oracle probabilistic polynomial-time Turing machine V. During its computation, V tosses random coins, reads its input and has oracle access to a string π called *proof*. Let now x be an input and π be a proof. We denote by $\mathbf{ACC}[V^{\pi}(x)]$ the probability over its random tosses that V accepts x using π as an oracle. We also denote by $\mathbf{ACC}[V(x)]$ the maximum of $\mathbf{ACC}[V^{\pi}(x)]$ over all proofs π.

Definition 4 (PCP Classes) *Let (Y, N) be a promise problem, let $0 < s < c \leq 1$ be two constants, let q be a positive integer and $r : \mathcal{Z}^+ \to \mathcal{Z}^+$. Then we say that $(Y, N) \in \mathsf{PCP}_{c,s}[r, q]$ if a verifier V exists for (Y, N) such that*

- *V uses $O(r(n))$ random bits, that is, for any input x and for any proof π, V tosses at most $O(r(|x|))$ random coins;*
- *V has query complexity q, that is, for any input x, V reads at most q bits from the proof;*
- *V has soundness s, that is, for any $x \in N$, $\mathbf{ACC}[V(x)] \leq s$;*
- *V has completeness c, that is, for any $x \in Y$, $\mathbf{ACC}[V(x)] \geq c$.*

Using the above notation, we can state the PCP Theorem as follows.

Theorem 5 (PCP Theorem [AS92b, ALM+92]) *A constant q exists such that*

$$\mathsf{NP} \subseteq \mathsf{PCP}_{1,1/2}[\log, q] \; .$$

We shall also use the short-hand notation $\mathsf{PCP}(r, 1) \stackrel{\text{def}}{=} \bigcup_{k \geq 1} \mathsf{PCP}_{2/3,1/3}[r, k]$. Note that the constants $2/3$ and $1/3$ are arbitrarily chosen, that is, for any fixed c and s with $0 < s < c < 1$, it holds $\mathsf{PCP}(r, 1) = \bigcup_{k \geq 1} \mathsf{PCP}_{c,s}[r, k]$.

2.1 Definition of the Problems

Recall that a *(k-ary) constraint function* is a Boolean function $f : \{0,1\}^k \to \{0,1\}$. A *constraint family* $\mathcal{F}$ is a finite collection of constraint functions. The *arity* of $\mathcal{F}$ is the maximum arity of the functions in $\mathcal{F}$. A *constraint* C over a variable set $\{x_1, \ldots, x_n\}$ is a pair $C = (f, (i_1, \ldots, i_k))$ where $f : \{0,1\}^k \to \{0,1\}$ is a constraint function and $i_j \in \{1, \ldots, n\}$ for $j = 1, \ldots, k$. The constraint C is said to be *satisfied* by an assignment $(a_1, \ldots, a_n)$ to $(x_1, \ldots, x_n)$ if $C(a_1, \ldots, a_n) \stackrel{\text{def}}{=} f(a_{i_1}, \ldots, a_{i_k}) = 1$. We say that constraint C is *from* $\mathcal{F}$ if $f \in \mathcal{F}$.

For a constraint family $\mathcal{F}$, the *constraint satisfaction problem* Max $\mathcal{F}$ is the maximization problem defined as follows:

Instance: A collection $\phi = \{C_1, \ldots, C_m\}$ of constraints from $\mathcal{F}$ over a variable set $X = \{x_1, \ldots, x_n\}$.

SOLUTION: A truth assignment τ for the variables in X.

MEASURE: Number of constraints satisfied by τ.

We now give a list of the constraint families used in this paper.

- For any $k \geq 1$, kSAT $= \{f : \{0,1\}^h \to \{0,1\} : |\{\vec{a} : f(\vec{a}) = 0\}| = 1 \wedge h \leq k\}$, that is, kSAT is the set of at-most-k-ary *disjunctive* constraints.
- SAT $= \bigcup_{k \geq 1} k$SAT
- For any $B \geq 1$, GSAT-B consists of all Boolean functions that admit a disjunctive normal form (DNF) representation where all the conjunctions have no more than B terms.
- For any $k \geq 1$, k-CSP consists of all h-ary Boolean functions, for $h \leq k$, that is, k-CSP $= \{f : \{0,1\}^h \to \{0,1\} \ : \ h \leq k\}$.

For a polynomial-time computable function $k : \mathcal{Z}^+ \to \mathcal{Z}^+$ we let MAX $k(n)$SAT be the restriction of MAX SAT to instances with constraints of arity at most $k(n)$ (where n is the number of variables). Finally, we will also make use of the MIN BIN PACKING problem defined as follows:

INSTANCE: Finite set U of items, and a size $s(u) \in \mathcal{Q}^+ \cap (0,1]$ for each $u \in U$.

SOLUTION: A partition of U into disjoint sets $U_1, U_2, \ldots, U_m$ such that the sum of the sizes of the items in each U_i is at most 1.

MEASURE: The number of used bins, that is, the number m of disjoint sets.

3 The Disjunction Shrinking Technique

In this section we introduce the technique to obtain approximation preserving reductions between constraint satisfaction problems. The basic idea is better explained in the special case of the reduction from MAX SAT to MAX 3SAT. Standard reduction techniques based on local replacement fail to reduce MAX SAT to MAX 3SAT due to the possible presence of large clauses. Large clauses, however, are easy to satisfy using a simple randomized algorithm (that, in turn, performs poorly on small clauses). We then *combine* (in a probabilistic sense) a solution based on standard reductions and one given by the randomized algorithm, and this mixed solution will be good for any combination of small and large clauses. The idea of probabilistically combining different solutions has been used in the design of *approximation algorithms* (e.g. in [GW94]) but we use it for the first time to develop an *approximation-preserving reduction.*

Theorem 6 (Disjunction Shrinking Theorem) *Let $p \in (0,1/2)$, $B, k \in \mathcal{Z}^+$, $r, r' \geq 1$ be such that*

$$r \geq \max\left\{\frac{1}{1-(1-p^B)^{k+1}} \, , \, \frac{r'}{(1-p)^B}\right\} \, .$$

Then r-approximating MAX GSAT-B *is reducible to r'-approximating* MAX kB-CSP.

PROOF: Let ϕ be an instance of MAX GSAT-B with m constraints, m_k be the number of constraints of ϕ with k or less B-ary conjuncts, and ϕ_k be the instance of MAX kB-CSP containing these m_k constraints. Define $m_l = m - m_k$ and $\phi_l = \phi - \phi_k$. Let τ_k be an r'-approximate solution for ϕ_k, and a be the number of constraints satisfied by τ_k. It is immediate to verify that

$$\mathsf{opt}(\phi) \leq r'a + m_l \ . \tag{1}$$

Indeed, any assignment cannot satisfy more than $r'a$ constraints in ϕ_k (otherwise τ_k would not be r'-approximate) and more then m_l constraints in ϕ_l. We now define a random assignment τ_R over the variables of ϕ with the following distribution:

- If a variable x occurs in ϕ_k, then

$$\mathbf{Pr}[\tau_R(x) = \tau_k(x)] = 1 - p \ .$$

- If a variable x occurs in ϕ but not in ϕ_k, then

$$\mathbf{Pr}[\tau_R(x) = \mathsf{true}] = 1/2 \ .$$

Let us now estimate the average number of constraints of ϕ that are satisfied by τ_R. Any literal is true with probability at least p (since $p < 1/2$, $1 - p > p$), thus the probability that a constraint in ϕ_l is contradicted is at most $(1 - p^B)^{k+1}$. On the other hand, if a constraint is satisfied by τ_k, then there is a probability at least $(1 - p)^B$ that it is still satisfied by τ_R. We can thus infer a lower bound on the average number of constraints of ϕ that are satisfied by τ_R:

$$\mathbf{E}[\mathsf{m}(\phi, \tau_R)] \geq (1 - p)^B a + (1 - (1 - p^B)^{k+1})m_l \geq (1/r)\mathsf{opt}(\phi)$$

where the last inequality is due to (1) and to the hypothesis on p, B, k, r, and r'. Using the method of conditional expectation [AS92a] (see also [Yan94]) we can find in linear time an assignment τ such that $\mathsf{m}(\phi, \tau) \geq \mathbf{E}[\mathsf{m}(\phi, \tau_R)]$. That is, the performance ratio of τ is at most r. □

From the proof of the above theorem we have the following result.

Theorem 7 *Let $p \in (0, 1/2)$, $k \in \mathcal{Z}^+$, $r, r' \geq 1$ be such that*

$$r \geq \max\{1/(1 - (1 - p)^{k+1}) \ , \ r'/(1 - p)\} \ .$$

Then r-approximating MAX SAT *is reducible to r'-approximating* MAX kSAT.

4 The Max NP-completeness Result

As a first application of the technique of the previous section we will now give a PCP-free proof of the Max NP-completeness of MAX 3SAT. Observe that from our definition of Max NP it is sufficient to prove that, for any $B \in \mathcal{Z}^+$, a PTAS-reduction from MAX GSAT-B to MAX 3SAT exists.

Theorem 8 MAX 3SAT *is* Max NP-*complete under* PTAS-*reductions.*

PROOF: The reduction will be the composition of the reduction from MAX GSAT-B to MAX kB-CSP given in Theorem 6 and of the standard reduction from MAX kB-CSP to MAX 3SAT [PY91]. For any h, the MAX h-CSP problem is in Max SNP, and thus a PTAS-reduction (f_h, g_h, c_h) exists from MAX h-CSP to MAX 3SAT. Let ϕ be an instance of MAX GSAT-B, and let $r > 1$ be fixed. We compute values k, r', p such that

$$r \geq \max\left\{ \frac{1}{1-(1-p^B)^{k+1}} , \frac{r'}{(1-p)^B} \right\} .$$

We then compute ϕ_k (the subset of ϕ containing constraints that are disjunctions of at most k conjuncts) that is an instance of MAX kB-CSP. From an r'-approximate solution for ϕ_k we are able to reconstruct an r-approximate solution for ϕ. The former problem can in turn be reduced to find a $c_{kB}(r')$-approximate solution for the instance $f_{kB}(\phi_k, r')$ of MAX 3SAT. $\square$

Remark 9 *We note that, in the proof of the above theorem, the "intermediate" problem* MAX kB-CSP *is not fixed since k depends on the approximation factor that we want to preserve. This is not in contradiction with the definition of* PTAS*-reduction. Indeed, we don't see how to use our technique in combination with other known reducibilities.*

We will now see that, in order to prove the APX-completeness of MAX 3SAT, at least a weak version of the PCP theorem is necessary. In the following we say that an algorithm T is a *non-constructive* PTAS for a maximization problem A if the following properties hold:

1. For any instance x of A and for any rational $r > 1$, $a = T(x, r)$ is a real number with the property that $\mathsf{opt}(x)/r \leq a \leq \mathsf{opt}(x)$.

2. For any fixed $r > 1$, the running time of $T(\cdot, r)$ is polynomial.

Non-constructive approximation for minimization problems is defined similarly. Note that if a problem is in PTAS then it admits a non-constructive PTAS, but the converse is not necessarily true (see [CT94]).

Lemma 10 $\mathsf{PCP}(\log n, 1) = \mathsf{P}$ *if and only if* MAX 3SAT *admits a non-constructive* PTAS.

PROOF: The "if" part is a restating of the standard reduction from PCP verifiers to MAX 3SAT (see [ALM+92]).

The other direction is more interesting. We use ideas from [CT94]. For any $1/2 \leq s < c \leq 1$, let GAP $3\mathrm{SAT}_{c,s}$ be the following promise problem: given a 3SAT formula ϕ with m clauses, reject if $\mathsf{opt}(\phi) \leq sm$ and accept if $\mathsf{opt}(\phi) \geq cm$. This problem is in $\mathsf{PCP}_{c,s}[\log, 3]$ (a proof is an assignment, the verifier picks a random clause and accepts if and only if the clause is satisfied by the assignment) and thus in $\mathsf{PCP}(\log n, 1) = \mathsf{P}$. Then a polynomial-time Turing machine $T_{c,s}$ exists such that, on input ϕ, $T_{c,s}$ accepts whenever $\mathsf{opt}(\phi) \geq cm$, rejects whenever $\mathsf{opt}(\phi) \leq sm$, and whose behaviour is undefined otherwise. Let $r > 1$ be fixed: we now describe a non-constructive r-approximate algorithm for MAX 3SAT. Let ϕ be a 3SAT formula with m clauses; its optimum lies somewhere between $m/2$ and m. We divide this interval into $k = \lceil 2/\log r \rceil$ subintervals, where the ith interval (for $i = 0, \ldots, k-1$) is $[m\alpha_i, m\alpha_{i+1})$ with $\alpha_i = .5r^{i/2}$. For any i, we run $T_{\alpha_{i+1},\alpha_i}(\phi)$. Let j be the largest index such that $T_{\alpha_{j+1},\alpha_j}(\phi)$ accepts. From the definition of $T_{c,s}$ it follows that $\mathsf{opt}(\phi) \geq m\alpha_j$ (since $T_{\alpha_{j+1},\alpha_j}(\phi)$ accepts) and that $\mathsf{opt}(\phi) \leq m\alpha_{j+2}$ (since $T_{\alpha_{j+2},\alpha_{j+1}}(\phi)$ rejects). It follows that α_j is a non-constructive r-approximate solution for ϕ. $\square$

The following result states that APX-complete problems are unlikely to have non-constructive PTAS's. This result has already been proved in [CT94] using the PCP theorem: the novelty of the proof that we give here is that it is PCP-free.

Lemma 11 *If an* APX-*complete problem* A *admits a non-constructive* PTAS, *then* NP = co-NP.

PROOF: Let (f, g, c) be a PTAS-reduction from MIN BIN PACKING to A; let $\bar{r} = c(1.4)$. Let I be an instance of MIN BIN PACKING such that the total size of the elements is 2; it is easy to see that the optimum packing of such instance will use either 2 or 3 bins. Distinguishing between these two cases is NP-hard (it is a restatement of the PARTITION[1] problem). Let $x = f(I, 1.4)$. From the definition of PTAS-reduction it follows that if y is an $\bar{r}$-approximate solution for x, then $g(I, y, 1.4)$ is a 1.4-approximate solution for I (indeed, an optimum solution). An NP algorithm can compute a non-constructive $\bar{r}$-approximate solution v for x, then guess a solution y of measure $\mathsf{m}(x, y) \geq v$, compute the solution $P = g(I, y, 1.4)$ and accept if and only if $\mathsf{m}(I, P) = 3$. It is easy to see that this is an NP algorithm for the complement of the PARTITION problem. □

The above two lemmas imply the following theorem.

Theorem 12 *If* MAX 3SAT *is* APX-*complete under* PTAS-*reductions, then* NP $\neq$ co-NP *implies* $\mathsf{PCP}(\log n, 1) \not\subseteq \mathsf{P}$.

This result essentially states that any proof of the APX-completeness of MAX 3SAT "contains" already a proof of the fact that NP $\neq$ *co*-NP implies $\mathsf{PCP}(\log n, 1) \not\subseteq \mathsf{P}$. This latter fact is weaker than the PCP Theorem in two ways: first, the verifier has not completeness 1 (i.e. it has *two-sided* error) and, second, the existence of intractable problems in $\mathsf{PCP}(\log n, 1)$ assumes NP $\neq$ *co*-NP instead that P $\neq$ NP. None of these variations appear to make the PCP theorem easier to be proved.

5 Linear-time Approximation Schemes

In this section we will consider the *planar* restriction of MAX GSAT-B and we will use our reduction technique to develop linear-time approximation schemes for this problem (in particular, for MAX SAT). The approximation schemes we describe are a composition of the reductions of Section 3 and of the linear time PTAS's for MAX kSAT and MAX k-CSP that are implied by the techniques of [Bak94, HMR+94, KM96].

To begin, we recall the definition of *planar instance* of a constraint satisfaction problem.

Definition 13 (Incidence Graph) *Let* $\mathcal{F}$ *be a (possibly infinite) constraint family. Let* ϕ *be an instance of* MAX $\mathcal{F}$ *over variable set* X. *The* incidence graph *of* ϕ, *denoted* $G_\phi = (V, E)$ *is defined as follows:*

- *V has a v-vertex for each variable $x \in X$ and an f-vertex for any constraint C of ϕ.*

[1]Recall that the NP-complete problem PARTITION is defined as follows: given a set U of items and a size function $s : U \to Q \cap (0, 1]$, does there exists a subset $U' \subseteq U$ such that

$$\sum_{u \in U'} s(u) = \sum_{u \notin U'} s(u)?$$

- *For each constraint C of ϕ and each variable x occurring in C there is an edge between the vertex for C and the vertex for x.*

For a constraint family $\mathcal{F}$, Max Planar $\mathcal{F}$ is the restriction of Max $\mathcal{F}$ to instances whose incidence graph is planar. We will focus on Max Planar k-CSP, Max Planar Sat, and Max Planar GSAT-B.

Khanna and Motwani [KM96] prove that the latter two problems have a PTAS that computes $(1+\epsilon)$-approximate solutions in time $n^{O(1/\epsilon)}$. We will improve the running time to $O(n)$ (the constant hidden in the O-notation will depend on ϵ and, in the Max Planar GSAT-B case, also on B). We use the fact that the techniques of Khanna and Motwani yield linear-time PTAS's when specialized to Max Planar kSat and Max Planar k-CSP (we give a sketch of this latter result). We first give some definitions.

Definition 14 (Tree Decomposition) *A* tree decomposition *of a graph $G = (V, E)$ is a tree $T = (I, F)$, where each node $i \in I$ is labelled by a subset $X(i)$ of V and such that:*

- $\bigcup_{i \in I} X(i) = V$.
- *For any $(u, v) \in E$, an $i \in I$ exists such that $u, v \in X(i)$,*
- *For any $v \in V$, the set $\{i \in I : v \in X(i)\}$ induces a subtree of T.*

Definition 15 (Treewidth) *The* width *of a tree-decomposition of a graph G is* $\max\{|X(i)| - 1 : i \in I\}$. *The* treewidth *of a graph is the minimum width over all its tree-decompositions.*

Theorem 16 ([Kha96]) *For fixed k and h,* Max k-CSP *restricted to instances whose incidence graph has treewidth at most h can be solved in linear time.*

Proof:[Sketch] Given an instance ϕ of Max k-CSP with m constraints over n variables whose incidence graph G has treewidth at most h we first find an optimum tree-decomposition of G. This can be done in linear time [Bod93]. Then we apply divide and conquer: the root vertex of the tree-decomposition is a set of $h + 1$ nodes of G that disconnect G into two components. If some of them are f-nodes than we replace them with the v-nodes corresponding to the variables occurring in them. This gives a separator S with at most $k(h + 1)$ v-nodes. We try all the possible assignments to the variables of S, for any such assignment we delete the variables of S from the incidence graph (thus disconnecting it) and then we recurse on the connected components of the incidence graph. Removing the root from the tree-decomposition of the incidence graph of ϕ gives tree-decompositions for all the connected components. The running time is given by the recursion

$$T(m, n) = 2^{k(h+1)}(T(m', n') + T(m'', n'')) \qquad T(1, 1) = O(1) ,$$

where $n = n' + n''$ and $m = m' + m''$. The recursion solves as $T(m, n) = O(2^{k(h+1)}(m + n)) = O(m + n)$. □

Theorem 17 *For fixed $\epsilon > 0$,* Max Planar k-CSP *admits an $(1 + \epsilon)$-approximate linear time algorithm,*

Proof: It suffices to observe that from the proof of [KM96, Theorem 1] (see also [Bak94, Theorem 1]) and from a result of [Bod88] it follows that $(1 + \epsilon)$-approximating Max Planar k-CSP reduces to optimally solve Max k-CSP restricted to instances whose incidence graph has treewidth $O(1/\epsilon)$. From Theorem 16 the claim thus follows. □

Theorem 18 *For any fixed $\epsilon > 0$ and $B \in \mathcal{Z}^+$,* Max Planar GSAT-B *admits a linear time $(1+\epsilon)$-approximate algorithm.*

Proof: Let k, p, ϵ' be constants (depending only on ϵ and B) such that $p \in (0, 1/2)$ and

$$1+\epsilon \geq \max\left\{\frac{1}{1-(1-p^B)^{k+1}} \, , \, \frac{1+\epsilon'}{(1-p)^B}\right\} .$$

Let ϕ be a planar instance of Max GSAT-B. The construction of Theorem 6 reduces the problem of $(1+\epsilon)$-approximating ϕ to the problem of $(1+\epsilon')$-approximating an instance ϕ' of Max kB-CSP obtained from ϕ by removing all the constraints with more than k conjuncts. The reduction runs in linear time, it doesn't increase the size of the instance and preserves planarity of the incidence graph. A $(1+\epsilon')$-approximate solution for ϕ' can be found in linear time using Theorem 17. $\square$

Remark 19 (NC Approximation Schemes) *Hunt et al. [HMR$^+$94] proved the following related result: for any fixed $\epsilon > 0$,* Max Planar k-CSP *admits an $(1+\epsilon)$-approximate* NC *algorithm. Since the reduction in the proof of Theorem 6 can be done in* NC *using k-wise independent distribution to do the derandomization, it follows that* Max Planar GSAT-B *admits a* NC *approximation scheme. This solves an open question in [HMR$^+$94].*

6 Comparison with Local-Replacement Reductions

Local-replacement reductions have played a fundamental role in proving NP-completeness results (see [GJ79]). In context of constraint satisfaction problems, a local-replacement reduction maps each constraint of the original problem into one or more constraints of the target, possibly introducing auxiliary variables.

Unfortunately, in [TSSW96] it has been shown this kind of reductions have inherent limitations. For example, it is shown that it is not possible to use local-replacement techniques to give an approximation-preserving reduction from Max $k(n)$SAT to Max $l(n)$SAT if $\lim_n k(n)/l(n) = \infty$. As a consequence, no local-replacement can show that these two problems have the same approximation threshold[2].

The next result shows that this latter fact is indeed true: its proof uses our disjunction-shrinking reduction technique (which is not a local-replacement one).

Theorem 20 *The approximation thresholds of* Max Sat *and* Max $k(n)$SAT *are equal, provided $k(n)$ is a monotone non-decreasing unbounded function.*

Proof: We prove that for any r and any $r' < r$, the existence of an r'-approximate algorithm for Max $k(n)$SAT implies the existence of an r-approximate algorithm for Max Sat.

There exist p and h such that $p \in (0, 1/2)$ and

$$r \geq \max\{1/(1-(1-p)^{h+1}) \, , \, r'/(1-p)\} .$$

Since $k(n)$ is non-decreasing and unbounded, there is some constant n_0 (depending only on h) such that $k(n) \geq h$ for any $n \geq n_0$. The approximation algorithm for Max Sat follows from

[2]The *approximation threshold* of an optimization problem $A \in$ APX is a real number $r_A \geq 1$ such that, for any $\epsilon > 0$, A admits an $(r_A+\epsilon)$-approximate polynomial-time algorithm but A does not admit an $(r_A-\epsilon)$-approximate polynomial-time algorithm.

Theorem 7 and from an r'-approximation algorithm for MAX hSAT. It remains to be seen that MAX hSAT is r'-approximable. Indeed, the only instances of MAX hSAT that are not instances of MAX $k(n)$SAT have at most $n_0 = O(1)$ variables. For these instances an optimum solution can be found in linear time. □

We conclude that our technique for shrinking disjunctions produces reductions that beat the local-replacement technique, that is, that are *provably better* than any possible reduction by local replacement that fits the framework of [TSSW96]. This encourages to look for simple techniques to obtain reductions to MAX CUT and MAX 2SAT beating the ones in [BGS95, TSSW96]: a positive answer would imply improved non-approximability results for these problems.

References

[ADP80] G. Ausiello, A. D'Atri, and M. Protasi. Structure preserving reductions among convex optimization problems. *Journal of Computer and System Sciences*, 21:136–153, 1980.

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science*, pages 14–23, 1992.

[AS92a] N. Alon and J. Spencer. *The Probabilistic Method.* Wiley Interscience, 1992.

[AS92b] S. Arora and S. Safra. Probabilistic checking of proofs; a new characterization of NP. In *Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science*, pages 2–13, 1992.

[BaG88] J.L. Balcázar, J. Díaz, and J Gabarró. *Structural Complexity I.* Springer-Verlag, 1988.

[Bak94] B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994.

[BC93] D.P. Bovet and P. Crescenzi. *Introduction to the Theory of Complexity.* Prentice Hall, 1993.

[BGS95] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCP's and non-approximability – towards tight results (3rd version). Technical Report TR95-24, Electronic Colloquium on Computational Complexity, 1995. Preliminary version in *Proc. of FOCS'95.*

[Bod88] H.L. Bodlaender. Some classes of graphs with bounded treewidth. *Bulletin of the EATCS*, 36:116–126, 1988.

[Bod93] H.L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. In *Proceedings of the 25th ACM Symposium on Theory of Computing*, pages 226–234, 1993.

[CKST95] P. Crescenzi, V. Kann, R. Silvestri, and L. Trevisan. Structure in approximation classes. In *Proceedings of the 1st Combinatorics and Computing Conference*, pages 539–548. LNCS 959, Springer Verlag, 1995.

[CP91] P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, 93:241–262, 1991. Preliminary version in *Proc. of FCT'89.*

[CT94] P. Crescenzi and L. Trevisan. On approximation scheme preserving reducibility and its applications. In *Proceedings of 14th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 330–341. LNCS 880, Springer Verlag, 1994.

[Fag74] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In *SIAM-AMS Proceedings*, pages 43–73, 1974.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to the Theory of NP-Completeness.* Freeman, 1979.

[GW94] M. Goemans and D. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics*, 7(4):656–666, 1994. Preliminary version in *Proc. of IPCO'93.*

[Hås96] J. Håstad. Getting optimal in-approximability results. Manuscript, September 1996.

[HMR$^+$94] H. B. Hunt III, M.V. Marathe, V. Radhakrishnan, S.S. Ravi, D.J. Rosenkrantz, and R.E. Stearns. Approximation schemes using L-reductions. In *Proceedings of 14th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 342–353. LNCS 880, Springer Verlag, 1994.

[Joh74] D.S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[Kha96] S. Khanna. Personal Communication, 1996.

[KM96] S. Khanna and R. Motwani. Towards a syntactic characterization of PTAS. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, 1996.

[KMSV94] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 819–830, 1994.

[KT94] P.G. Kolaitis and M.N. Thakur. Logical definability of NP optimization problems. *Information and Computation*, 115(2):321–353, 1994.

[KT95] P.G. Kolaitis and M.N. Thakur. Approximation properties of NP minimization classes. *Journal of Computer and System Sciences*, 50:391–411, 1995. Preliminary version in *Proc. of Structures91.*

[OM87] P. Orponen and H. Mannila. On approximation preserving reductions: complete problems and robust measures. Technical Report C-1987-28, Department of Computer Science, University of Helsinki, 1987.

[Pap94] C.H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[PM81] A. Paz and S. Moran. Non deterministic polynomial optimization problems and their approximation. *Theoretical Computer Science*, 15:251–277, 1981.

[PR93] A. Panconesi and D. Ranjan. Quantifiers and approximations. *Theoretical Computer Science*, 107:145–163, 1993. Preliminary version in *Proc. of STOC'90.*

[PY91] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991. Preliminary version in *Proc. of STOC'88.*

[Sim89] H.U. Simon. Continous reductions among combinatorial optimization problems. *Acta Informatica*, 26:771–785, 1989.

[TSSW96] L. Trevisan, G.B. Sorkin, M. Sudan, and D.P. Williamson. Gadgets, approximation, and linear programming. In *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science*, 1996.

[Yan94] M. Yannakakis. On the approximation of maximum satisfiability. *Journal of Algorithms*, 17:475–502, 1994. Preliminary version in *Proc. of SODA'92.*