On Syntactic versus Computational Views of Approximability

SANJEEV KHANNA* RAJEEV MOTWANI* MADHU SUDAN† UMESH VAZIRANI‡
Stanford University Stanford University IBM Research U.C. Berkeley

Abstract

We attempt to reconcile the two distinct views of approximation classes: syntactic *and* computational. *Syntactic classes such as* MAX SNP *permit structural results and have natural complete problems, while computational classes such as* APX *allow us to work with classes of problems whose approximability is well-understood. Our results provide a syntactic characterization of computational classes, and give a computational framework for syntactic classes.*

We compare the syntactically defined class MAX SNP *with the computationally defined class* APX*, and show that every problem in* APX *can be "placed" (i.e. has approximation preserving reduction to a problem) in* MAX SNP*. Our methods introduce a general technique for creating approximation-preserving reductions which show that any "well" approximable problem can be reduced in an approximation-preserving manner to a problem which is hard to approximate to corresponding factors. We demonstrate this technique by applying it to the classes* RMAX(2) *and* MIN $\mathrm{F}^{+}\Pi_2(1)$ *which have the clique problem and the set cover problem, respectively, as complete problems.*

We use the syntactic nature of MAX SNP *to define a general paradigm,* non-oblivious local search*, useful for developing simple yet efficient approximation algorithms. We show that such algorithms can find good approximations for all* MAX SNP *problems, yielding approximation ratios comparable to the best-known for a variety of specific* MAX SNP*-hard problems. Non-oblivious local search provably out-performs standard local search in both the degree of approximation achieved and the efficiency of the resulting algorithms.*

*Department of Computer Science, Stanford University, Stanford, CA 94305 ({khanna,motwani}@cs.stanford.edu). Supported by an IBM Faculty Development Award, an OTL grant, and NSF Young Investigator Award CCR-9357849, with matching funds from IBM, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

†IBM T.J. Watson Research Center, Yorktown Heights, NY 10598 (madhu@watson.ibm.com).

‡Computer Science Division, U.C. Berkeley, CA 94720. Supported by NSF Grant CCR-9310214 (vazirani@cs.berkeley.edu).

1 Introduction

The approximability of NP optimization (NPO) problems has been investigated in the past via the definition of two different types of problem classes: syntactically-defined classes such as MAX SNP, and computationally-defined classes such as APX (the class of optimization problems to which a constant factor approximation can be found in polynomial time). The former is useful for obtaining structural results and has natural complete problems, while the latter allows us to work with classes of problems whose approximability is completely determined. We attempt to develop linkages between these two views of approximation problems and thereby obtain new insights about both types of classes. We show that a natural generalization of MAX SNP renders it identical to the class APX. This is an unexpected validation of Papadimitriou and Yannakakis's definition of MAX SNP as an attempt at providing a structural basis to the study of approximability. As a side-effect, we resolve the open problem of identifying complete problems for MAX NP. Our techniques extend to a generic theorem which can be used to create an approximation hierarchy. We also develop a generic algorithmic paradigm which is guaranteed to provide good approximations for MAX SNP problems, and may also have other applications.

1.1 Background and Motivation

A wide variety of classes are defined based directly on the polynomial-time approximability of the problems contained within, e.g., APX (constant-factor approximable problems), PTAS (problems with polynomial-time approximation schemes), and FPTAS (problems with fully-polynomial-time approximation schemes). The advantage of working with classes defined using approximability as the criterion is that it allows us to work with problems whose approximability is well-understood. Crescenzi and Panconesi [9] have recently also been able to exhibit complete problems for such classes, particularly APX. Unfortunately such complete problems seem to be rare and artificial, and do not seem to provide insight into the more natural problems in the class. Research in this direction has to find approximation-preserving reductions from the known complete but artificial problems in such classes to the natural problems therein, with a view to understanding the approximability of the latter.

The second family of classes of NPO problems that have been studied are those defined via syntactic considerations, based on a syntactic characterization of NP due to Fagin [10]. Research in this direction, initiated by Papadimitriou and Yannakakis [21], and followed by Panconesi and Ranjan [20] and Kolaitis and Thakur [18], has led to the identification of approximation classes such as MAX SNP, RMAX(2), and MIN $F^+\Pi_2(1)$. The syntactic prescription in the definition of these classes has proved very useful in the establishment of complete problems. Moreover, the recent results of Arora et al [3] have established the hardness of approximating complete problems for MAX SNP to within (specific) constant factors unless P $=$ NP. It is natural to ask why the hardest problems in this syntactic sub-class of APX should bear any relation to all of NP.

Though the computational view allows us to precisely classify the problems based on their approximability, it does not yield structural insights into natural questions such as why certain problems are easier to approximate than some others, what is the canonical structure of the hardest representative problems of a given approximation class, and so on. Furthermore, intuitively speaking, this view is too abstract to facilitate identification of, and reductions to establish, natural complete problems for a class. The syntactic view, on the other hand, is essentially a structural view. The syntactic prescription gives a natural way of identifying canonical hard problems in the class and performing approximation-preserving reductions to establish complete problems.

Attempts at trying to find a class with both the above mentioned properties, i.e. natural complete problems and capturing all problems of a specified approximability, have not been very successful. Typically the focus has been to relax the syntactic criteria to allow for a wider class of problems to be included in the class. However in all such cases it seems inevitable that these classes cannot be expressive enough to encompass all problems with a given approximability. This is because each of these syntactically defined approximation classes is strictly contained in the class NPO; the strict containment can be shown by syntactic considerations alone. As a result if we could show that any of these classes contains all of P, then we would have separated P from NP. We would expect that every class of this nature would be missing some problems from P, and this has indeed been the case with all current definitions.

We explore a different direction by studying the structure of the syntactically defined classes when we look at their closure under approximation-preserving reductions. The advantage of this is that the closure maintains the complete problems of the set, while managing to include all of P into the closure (for problems in P, the reduction is to simply use a polynomial time algorithm to compute an exact solution). It now becomes interesting, for example, to compare the closure[§] of MAX SNP (denoted $\overline{\text{MAX SNP}}$) with APX. A positive resolution, i.e., $\overline{\text{MAX SNP}} = \text{APX}$, would immediately imply the non-existence of a PTAS for MAX SNP-hard problems, since it is known that PTAS is a strict subset of APX, if P $\neq$ NP. On the other hand, an unconditional negative result would be difficult to obtain, since it would imply P $\neq$ NP.

Here we resolve this question in the affirmative. The exact nature of the result obtained depends upon the precise notion of an approximation preserving reduction used to define the closure of the class MAX SNP. The strictest notion of such reductions available in the literature are the L-reductions due to Papadimitriou and Yannakakis [21]. We introduce a new notion of reductions, called E-reductions, which are a slight extension of L-reductions. Using such reductions to define the class $\overline{\text{MAX SNP}}$ we show that this equals APX-PB, the class of all polynomially bounded NP optimization problems which are approximable to within constant factors. By using slightly looser definitions of approximation preserving reductions (and in particular the PTAS-reductions of Crescenzi et al [8]) this can be extended to include all of APX into $\overline{\text{MAX SNP}}$. We then build upon this result to identify an interesting hierarchy of such approximability classes. An interesting side-effect of our results is the positive answer to the question of Papadimitriou and Yannakakis [21] about whether MAX NP has any complete problems.

The syntactic view seems useful not only in obtaining structural complexity results but also in developing paradigms for designing efficient approximation algorithms. Exploiting the syntactic nature of MAX SNP, we develop a general paradigm for designing good approximation algorithms for problems in that class and thereby provide a more computational view of it. We refer to this paradigm as *non-oblivious local search*, and it is a modification of the standard local search technique [23]. We show that every MAX SNP problem can be approximated to within constant factors by such algorithms. It turns out that the performance of non-oblivious local search is comparable to that of the best-known approximation algorithms for several interesting and representative problems in MAX SNP. An intriguing possibility is that this is not a coincidence, but rather a hint at the universality of the paradigm or some variant thereof.

Our results are related to some extent to those of Ausiello and Protasi [4]. They define a class GLO (for Guaranteed

[§]Papadimitriou and Yannakakis [21] hinted at the definition of $\overline{\text{MAX SNP}}$ by stating that: *minimization problems will be "placed" in the classes through L-reductions to maximization problems.*

Local Optima) of NPO problems which have the property that for all locally optimum solutions, the ratio between the value of the global and the local optimum is bounded by a constant. It follows that GLO is a subset of APX, and it was shown that it is in fact a strict subset. We show that a MAX SNP problem is not contained in GLO, thereby establishing that MAX SNP is not contained in GLO. This contrasts with our notion of non-oblivious local search which is guaranteed to provide constant factor approximations for all problems in MAX SNP. In fact, our results indicate that non-oblivious local search is significantly more powerful than standard local search in that it delivers strictly better constant ratios, and also will provide constant factor approximations to problems not in GLO. Independently of our work, Alimonti [1] has used a similar local search technique for the approximation of a specific problem not contained in GLO or MAX SNP.

1.2 Summary of Results

In Section 2, we present the definitions required to state our results, and in particular the definitions of an E-reduction, APX, APX-PB, MAX SNP and $\overline{\text{MAX SNP}}$. In Section 3, we show that $\overline{\text{MAX SNP}} = \text{APX-PB}$. A generic theorem which allows to equate the closure of syntactic classes to appropriate computational classes is outlined in Section 4; we also develop an approximation hierarchy based on this result.

The notion of non-oblivious local search and NON-OBLIVIOUS GLO is developed in Section 5. In Section 6, we illustrate the power of non-obliviousness by first showing that oblivious local search can achieve at most the performance ratio $3/2$ for MAX 2-SAT, even if it is allowed to search *exponentially* large neighborhoods; in contrast, a very simple non-oblivious local search algorithm achieves a performance ratio of $4/3$. We then establish that this paradigm yields a $2^k/(2^k-1)$ approximation to MAX *k*-SAT. In Section 7, we provide an alternate characterization of MAX SNP via a class of problems called MAX *k*-CSP. It is shown that a simple non-oblivious algorithm achieves the best-known approximation for this problem, thereby providing a *uniform* approximation for all of MAX SNP. In Section 8, we further illustrate the power of this class of algorithms by showing that it can achieve the best-known ratio for a specific MAX SNP problem and for VERTEX COVER (which is not contained in GLO). This implies that MAX SNP is not contained in GLO, and that GLO is strict subset of NON-OBLIVIOUS GLO. In Section 9, we apply it to approximating the traveling salesman problem. Finally, in Section 10, we apply this technique to improving a long-standing approximation bound for maximum independent sets in bounded-degree graphs.

2 Preliminaries and Definitions

Given an NPO problem Π and an instance $\mathcal{I}$ of Π, we use $|\mathcal{I}|$ to denote the length of $\mathcal{I}$ and $OPT(\mathcal{I})$ to denote the optimum value for this instance. For any solution S to $\mathcal{I}$, the value of the solution, denoted by $V(\mathcal{I}, S)$, is assumed to be a polynomial time computable function which takes positive integer values (see [7] for a precise definition of NPO).

Definition 1 (Error) *A solution S to an instance $\mathcal{I}$ of an* NPO *problem Π has* error $\mathcal{E}(\mathcal{I}, S)$ *if*

$$\frac{1}{1+\mathcal{E}(\mathcal{I},S)} \leq \frac{V(\mathcal{I},S)}{OPT(\mathcal{I})} \leq 1+\mathcal{E}(\mathcal{I},S).$$

Notice that the above definition of error applies uniformly to the minimization and maximization problems at all levels of approximability.

Definition 2 (Performance Ratio) *An approximation algorithm A for an optimization problem Π has* performance ratio $\mathcal{R}(n)$ *if, given an instance $\mathcal{I}$ of Π with $|\mathcal{I}| = n$, the solution $A(\mathcal{I})$ satisfies*

$$\max\left\{\frac{V(\mathcal{I},A(\mathcal{I}))}{OPT(\mathcal{I})}, \frac{OPT(\mathcal{I})}{V(\mathcal{I},A(\mathcal{I}))}\right\} \leq \mathcal{R}(n).$$

*A solution of value within a multiplicative factor r of the optimal value is referred to as an r-*approximation*.*

The performance ratio for A is $\mathcal{R}$ if it always computes a solution with error at most $\mathcal{R} - 1$.

2.1 E-reductions

We now describe the precise approximation preserving reduction we will use in this paper. Various other notions of approximation preserving reductions exist in the literature (cf. [16, 2]) but the reduction which we use, referred to as the E-reduction (for *error-preserving reduction*), seems to be the strictest. As we will see, the E-reduction is essentially the same as the L-reduction of Papadimitriou and Yannakakis [21] and differs from it in only one relatively minor aspect.

Definition 3 (E-reduction) *A problem Π E-*reduces *to a problem Π' (denoted $\Pi \propto_E \Pi'$) if there exist polynomial time computable functions f, g and a constant β such that*

- *f maps an instance $\mathcal{I}$ of Π to an instance $\mathcal{I}'$ of Π',*
- *g maps solutions S' of $\mathcal{I}'$ to solutions S of $\mathcal{I}$ such that*

$$\mathcal{E}(\mathcal{I},S) \leq \beta\mathcal{E}(\mathcal{I}',S').$$

Remark 1 *An E-reduction is essentially the strictest possible notion of reduction. It requires that the error for Π be linearly related to the error for Π'. Most other notions of reductions in the literature, for example the F-reductions and P-reductions of Crescenzi and Panconesi* [9], *do not enforce this condition. One important consequence of this constraint is that E-reductions are* sensitive, *i.e., when $\mathcal{I} \in \Pi$ is mapped to $\mathcal{I}' \in \Pi'$ under an E-reduction then a good solution to $\mathcal{I}'$ should provide structural information about a good solution to $\mathcal{I}$. Thus, reductions from real optimization problems to decision problems artificially encoded as optimization problems are implausible.*

Remark 2 *Having $\Pi \propto_E \Pi'$ implies that Π is as well approximable as Π'; in fact, an E-reduction is an* FPTAS-*preserving reduction. An important benefit is that this reduction can applied uniformly at all levels of approximability. This is not the case with the other existing definitions of* FPTAS-*preserving reduction in the literature. For example, the* FPTAS-*preserving reduction (F-reduction) of Crescenzi and Panconesi* [9] *is much more unrestricted in scope and does not share this important property of the E-reduction. Note that Crescenzi and Panconesi* [9] *showed that there exists a problem $\Pi' \in$* PTAS *such that for any problem $\Pi \in$* APX, *$\Pi \propto_F \Pi'$. Thus, there is the undesirable situation that a problem Π with no* PTAS *has a* FPTAS-*preserving reduction to a problem Π' with a* PTAS.

Remark 3 *The L-reduction of Papadimitriou and Yannakakis* [21] *enforces the condition that the optima of an instance $\mathcal{I}$ of Π be linearly related to the optima of the instance $\mathcal{I}'$ of Π' to which it is mapped. This appears to be an unnatural restriction considering that the reduction itself is allowed to be an arbitrary polynomial time computation. This is the only real difference between their L-reduction and our E-reduction, and an E-reduction in which the linearity relation of the optimas is satisfied is an L-reduction. Intuitively, however, in the study of approximability the desirable attribute is simply that the errors in the corresponding solutions are closely (linearly) related. The somewhat artificial requirement of a linear relation between the optimum values precludes reductions between problems which are related to each other by some* scaling factor. *For instance, it seems desirable that two problems whose objective functions are simply related by any fixed polynomial factor should be inter-reducible under any reasonable definition of an approximation-preserving reduction. Our relaxation of the L-reduction constraint is motivated precisely by this consideration.*

Let $\mathcal{C}$ be any class of NPO problems. Using the notion of an E-reduction, we define hardness and completeness of problems with respect $\mathcal{C}$, as well its closure and polynomially-bounded sub-class.

Definition 4 (Hard and Complete Problems) *A problem Π' is said to be $\mathcal{C}$-*hard *if for all problems $\Pi \in \mathcal{C}$, we have $\Pi \propto_E \Pi'$. A $\mathcal{C}$-hard problem Π is said to be $\mathcal{C}$-*complete *if in addition $\Pi \in \mathcal{C}$.*

Definition 5 (Closure) *The* closure *of $\mathcal{C}$, denoted by $\overline{\mathcal{C}}$, is the set of all* NPO *problems Π such that $\Pi \propto_E \Pi'$ for some $\Pi' \in \mathcal{C}$.*

Remark 4 *The closure operation maintains the set of complete problems for a class.*

Definition 6 (Polynomially Bounded Subset) *The* polynomially bounded subset *of $\mathcal{C}$, denoted $\mathcal{C}$-PB, is the set of all problems $\Pi \in \mathcal{C}$ for which there exists a polynomial $p(n)$ such that for all instances $\mathcal{I} \in \Pi$, $OPT(\mathcal{I}) \leq p(|\mathcal{I}|)$.*

2.2 Computational and Syntactic Classes

We first define the basic computational class APX.

Definition 7 (APX) *An* NPO *problem Π is in the class* APX *if there exists a polynomial time computable function A mapping instances of Π to solutions, and a constant $c \geq 1$, such that for all instances $\mathcal{I}$ of Π,*

$$\frac{V(I, A(\mathcal{I}))}{c} \leq OPT(\mathcal{I}) \leq c \times V(I, A(\mathcal{I})).$$

The class APX-PB consists of all polynomially bounded NPO problems which can be approximated within constant factors in polynomial time.

If we let F-APX denote the class of NPO problems which are approximable to a factor F, then we obtain a *hierarchy* of approximation classes. For instance, poly-APX and log-APX are the classes of NPO problems which have polynomial time algorithms with performance ratio bounded polynomially and logarithmically, respectively, in the input length. A more precise form of these definitions are provided in Section 4.

Let us review the definition of some syntactic classes.

Definition 8 (MAX SNP and MAX NP [21]) MAX SNP *is the class of* NPO *problems expressible as finding the structure S which maximizes the objective function*

$$V(\mathcal{I}, S) = |\{\vec{x} \mid \Phi(\mathcal{I}, S, \vec{x})\}|,$$

where $\mathcal{I} = (U; \mathcal{P})$ denotes the input (consisting of a finite universe U and a finite set of bounded arity predicates $\mathcal{P}$), S is a finite structure, and Φ is a quantifier-free first-order formula. The class MAX NP *is defined analogously except the objective function is*

$$V(\mathcal{I}, S) = |\{\vec{x} \mid \exists \vec{y} \Phi(\mathcal{I}, S, \vec{x}, \vec{y})\}|.$$

For the purposes of this abstract, we omit the definition and treatment of the weighted version of this class.

Example 1 (MAX k-SAT) *The* MAX k-SAT *problem is: given a collection of m clauses on n boolean variables where each (possibly weighted) clause is a disjunction of precisely k literals, find a truth assignment satisfying a maximum weight collection of clauses. For any fixed integer k,* MAX k-SAT *belongs to the class* MAX SNP. *The results of Papadimitriou and Yannakakis [21] can be adapted to show that for $k \geq 2$,* MAX k-SAT *is complete under E-reductions for the class* MAX SNP.

Definition 9 (RMAX(k) [20]) RMAX(k) *is the class of* NPO *problems expressible as finding a structure S which maximizes the objective function*

$$V(\mathcal{I}, S) = \begin{cases} |\{\vec{x} : S(\vec{x})\}| & \textit{if } \forall \vec{y}, \Phi(\mathcal{I}, S, \vec{y}) \\ 0 & \textit{otherwise} \end{cases}$$

where S is a single predicate and $\Phi(\mathcal{I}, S, \vec{y})$ is a quantifier-free CNF formula in which S occurs at most k times in each clause and all its occurrences are negative.

The results of Panconesi and Ranjan [20] can be adapted to show that MAX CLIQUE is complete under E-reductions for the class RMAX(2).

Definition 10 (MIN $\mathbf{F}^{+}\boldsymbol{\Pi}_2(\boldsymbol{k})$ [18]) MIN $\mathrm{F}^{+}\Pi_2(k)$ *is the class of* NPO *problems expressible as finding a structure S which minimizes the objective function*

$$V(\mathcal{I}, S) = \begin{cases} |\{\vec{x} : S(\vec{x})\}| & \textit{if } \forall \vec{x}, \exists \vec{y}, \Phi(\mathcal{I}, S, \vec{x}, \vec{y}) \\ 0 & \textit{otherwise} \end{cases}$$

where S is a single predicate, $\Phi(\mathcal{I}, S, \vec{y})$ is a quantifier-free CNF formula in which S occurs at most k times in each clause and all its occurrences are positive.

The results of Kolaitis and Thakur [18] can be adapted to show that SET COVER is complete under E-reductions for the class MIN $\mathrm{F}^{+}\Pi_2(1)$.

3 MAX SNP Closure and APX-PB

In this section, we will establish the following theorem and examine its implications. The proof is based on the results of Arora et al [3] on efficient proof verifications.

Theorem 1 $\overline{\text{MAX SNP}}$ = APX-PB.

Remark 5 *The seeming weakness that* $\overline{\text{MAX SNP}}$ *only captures polynomially bounded* APX *problems can be removed by using more looser forms of approximation preserving reduction in defining the closure. In particular, Crescenzi et al [8] define the notion of a* PTAS*-preserving reduction under which* APX $=$ $\overline{\text{APX-PB}}$*. Using their result in conjunction with the above theorem, it is easily seen that* $\overline{\text{MAX SNP}}$ $=$ APX*. This weaker reduction is necessary to allow for reductions from fine-grained optimization problems to coarser (polynomially-bounded) optimization problems (cf.* [8]*).*

The following is a surprising consequence of Theorem 1.

Theorem 2 $\overline{\text{MAX NP}} = \overline{\text{MAX SNP}}$.

Papadimitriou and Yannakakis [21] (implicitly) introduced both these closure classes but did not conjecture them to be the same. It would be interesting to see if this equality can be shown independent of the result of Arora et al [3]. We also obtain the following resolution to the problem posed by Papadimitriou and Yannakakis [21] of finding complete problems for MAX NP.

Theorem 3 MAX SAT *is complete for* MAX NP.

The following sub-sections sketch the proof of Theorem 1. The idea is to first E-reduce any minimization problem in APX-PB to a maximization problem in therein, and then E-reduce any maximization problem in APX-PB to a specific complete problem for MAX SNP, viz. MAX 3-SAT.

3.1 Reducing Minimization to Maximization

Observe that the fact that Π belongs to APX implies the existence of an approximation algorithm A and a constant c such that

$$\frac{OPT(\mathcal{I})}{c} \leq V(\mathcal{I}, A(\mathcal{I})) \leq c \times OPT(\mathcal{I}).$$

Henceforth, we will use $a(\mathcal{I})$ to denote $V(\mathcal{I}, A(\mathcal{I}))$. We first reduce any minimization problem $\Pi \in$ APX-PB to a maximization problem $\Pi' \in$ APX-PB, where the latter is obtained by merely modifying the objective function for Π, as follows: let Π' have the objective function $V'(\mathcal{I}, S) = (c+1)a(\mathcal{I}) - cV(\mathcal{I}, S)$, for all instances $\mathcal{I}$ and solutions S for Π. It can be verified that the optimum value for any instance $\mathcal{I}$ of Π' always lies between $a(\mathcal{I})$ and $(c+1)a(\mathcal{I})$. Thus, A is a $(c+1)$-approximation algorithm for Π'. If S is a δ-error solution to the optimum of Π', i.e.,

$$V'(\mathcal{I}, S) \geq \frac{OPT'(\mathcal{I})}{1+\delta} \geq (1-\delta) OPT'(\mathcal{I}),$$

where $OPT'(\mathcal{I})$ is the optimal value of V' for $\mathcal{I}$. We obtain that

$$V(\mathcal{I}, S) \quad = \quad \frac{(c+1)a(\mathcal{I}) - V'(\mathcal{I}, S)}{c}$$

$$
\begin{aligned}
&\leq \frac{(c+1)a(\mathcal{I}) - OPT'(\mathcal{I}) + \delta \times OPT'(\mathcal{I})}{c} \\
&\leq \frac{c \times OPT(\mathcal{I}) + \delta \times OPT'(\mathcal{I})}{c} \\
&\leq OPT(\mathcal{I}) + (c+1)\delta\, OPT(\mathcal{I}).
\end{aligned}
$$

Thus a solution s to Π' with error δ is a solution to Π with error at most $(c+1)\delta$, implying an E-reduction with $\beta = c+1$.

3.2 NP Languages and MAX 3-SAT

The following theorem, adapted from a result of Arora et al [3], is critical to our E-reduction of maximization problems to MAX 3-SAT.

Theorem 4 *Given a language $L \in$ NP and an instance $x \in \Sigma^n$, one can compute in polynomial time an instance $\mathcal{F}_x$ of* MAX 3-SAT*, with the following properties:*

1. *$\mathcal{F}_x$ has m clauses, where m depends only on n.*
2. *There exists a constant $\epsilon > 0$ such that $(1-\epsilon)m$ clauses of $\mathcal{F}_x$ are satisfied by some truth assignment.*
3. *$x \in L \Rightarrow \mathcal{F}_x$ is satisfiable (completely).*
4. *$x \notin L \Rightarrow$ no truth assignment satisfies more than $(1-\epsilon)m$ clauses of $\mathcal{F}_x$.*
5. *Given a truth assignment which satisfies more than $(1-\epsilon)m$ clauses of $\mathcal{F}_x$, a truth assignment which satisfies $\mathcal{F}_x$ completely can be constructed in polynomial time.*

Some of the properties above may not be immediately obvious from the construction given by Arora et al [3]. It is easy to verify that they provide a reduction with properties (1), (3) and (4). Property (5) is obtained from the fact that all assignments which satisfy most clauses are actually close (in terms of Hamming distance) to valid codewords from a linear code, and the uniquely error-corrected codeword obtained from this "corrupted code-word" will satisfy all the clauses of $\mathcal{F}_x$.

Property (2) requires a bit more care and we provide a brief sketch of how it may be ensured. The idea is to revert back to the PCP model and redefine the proof verification game. Suppose that the original game had the properties that for $x \in L$ then there exists a proof such that the verifier accepts with probability 1, and for $x \notin L$ the verifier accepts with probability at most $1/2$. We now augment this game by adding to the proof a 0th bit, which the prover uses as follows: if the bit is set to 1, then the prover "chooses" to play the old game, else he is effectively "giving up" on the game. The verifier in turn first looks at the 0th bit of the proof. If this is set, then she performs the usual verification, else she tosses an unbiased coin and accepts if and only if it turns up heads. It is clear that for $x \in L$ there exists a proof on which the verifier always accepts. Also, for $x \notin L$ no proof can cause the verifier to accept with probability greater than $1/2$. Finally, by setting the 0th bit to 0, the prover can create a proof which the verifier accepts with probability exactly $1/2$. This proof system can now be transformed into a 3-CNF formula of the desired form.

3.3 Reducing Maximization to MAX 3-SAT

We have already established that, without loss of generality, we only need to worry about maximization problems $\Pi \in$ APX-PB. Consider such a problem Π, and let A be a polynomial-time algorithm which delivers a c-approximation for Π, where c is some constant. Given any instance $\mathcal{I}$ of Π, let $p = c \times a(\mathcal{I})$ be the bound on the optimum value for $\mathcal{I}$ obtained by running A on input $\mathcal{I}$. Note that this may be a stronger bound than the a priori polynomial bound on the optimum value for any instance of length $|\mathcal{I}|$. An important consequence is that $p \leq c\, OPT(\mathcal{I})$.

We generate a sequence of NP decision problems $L_i = \{\mathcal{I} \,|\, OPT(\mathcal{I}) \geq i\}$ for $1 \leq i \leq p$. Given an instance $\mathcal{I}$, we create p formulas $\mathcal{F}_i$, for $1 \leq i \leq p$, using the reduction from Theorem 4, where ith formula is obtained from the NP language L_i.

Consider now the formula $\mathcal{F} = \bigwedge_{i=1}^{p} \mathcal{F}_i$ that has the following features:

- The number of satisfiable clauses of $\mathcal{F}$ is exactly
$$MAX = (1-\epsilon)mp + \epsilon m\, OPT(\mathcal{I}),$$
where ϵ and m are as guaranteed by Theorem 4.
- Given an assignment which satisfies $(1-\epsilon)mp + \epsilon mj$ clauses of $\mathcal{F}$, we can construct in polynomial time a solution to $\mathcal{I}$ of value at least j. To see this, observe the following: any assignment which so many clauses must satisfy more than $(1-\epsilon)m$ clauses in at least j of the formulas $\mathcal{F}_i$. Let i be the largest index for which this happens; clearly, $i \geq j$. Furthermore, by property (5) of Theorem 4, we can now construct a truth assignment which satisfies $\mathcal{F}_i$ completely. This truth assignment can be used to obtain a solution S such that $V(\mathcal{I}, S) \geq i \geq j$.

In order to complete the proof it remains to be shown that given any truth assignment with error δ, i.e., which satisfies $MAX\,/(1+\delta)$ clauses of $\mathcal{F}$, we can find a solution S for $\mathcal{I}$ with error $\mathcal{E}(\mathcal{I}, S) \leq \beta\delta$ for some constant β. We show that this is possible for $\beta = (c^2 + c\epsilon)/\epsilon$. The main idea behind finding such a solution is to use the second property above to find a "good" solution to $\mathcal{I}$ using a "good" truth assignment for $\mathcal{F}$.

Suppose we are given a solution which satisfies $MAX/(1+\delta)$ clauses. Since $MAX/(1+\delta) \geq (1-\delta)\,MAX$ and $MAX = (1-\epsilon)mp + \epsilon m\,OPT(\mathcal{I})$, we can use the second feature from above to construct a solution S_1 such that

$$\begin{aligned} V(\mathcal{I}, S_1) &\geq \frac{(1-\delta)\,MAX - (1-\epsilon)mp}{\epsilon m} \\ &\geq (1-\delta)\,OPT(\mathcal{I}) - \frac{\delta}{\epsilon}p \\ &\geq \left(1 - \delta\left(1 + \frac{c}{\epsilon}\right)\right) OPT(\mathcal{I}). \end{aligned}$$

Let $\delta^* = \delta(1 + c/\epsilon)$, then it is readily seen that

$$V(\mathcal{I}, s_1) \geq \frac{OPT(\mathcal{I})}{1+\gamma}$$

where $\gamma = \delta^*/(1-\delta^*)$. Assuming $\delta^* \leq (c-1)/c$, we obtain that

$$\gamma \leq \left(\frac{c^2 + c\epsilon}{\epsilon}\right) \times \delta.$$

On the other hand, if $\delta^* \geq (c-1)/c$, then the error in a solution S_2 obtained by running the c-approximation algorithm for Π is given by

$$c - 1 \leq \left(\frac{c^2 + c\epsilon}{\epsilon}\right)\delta.$$

Therefore, choosing $\beta = (c^2 + c\epsilon)/\epsilon$, we immediately obtain that the solution with larger value, among S_1 and S_2, has error at most $\beta\delta$. Thus, this reduction is indeed an E-reduction.

4 Generic Reductions and an Approximation Hierarchy

In this section we describe a fairly generic technique for turning a hardness result into an approximation preserving reduction.

We start by listing the kind of constraints imposed on the hardness reduction, the approximation class and the optimization problem. We will observe at the end that these restrictions are obeyed by all known hardness results and the corresponding approximation classes.

Definition 11 (Additive Problems) *An* NPO *problem* Π *is said to be* additive *if there exists an operator* $+$ *which maps a pair of instances* $\mathcal{I}_1$ *and* $\mathcal{I}_2$ *to an instance* $\mathcal{I}_1 + \mathcal{I}_2$ *such that* $OPT(\mathcal{I}_1 + \mathcal{I}_2) = OPT(\mathcal{I}_1) + OPT(\mathcal{I}_2)$.

Definition 12 (Downward Closed Family) *A family of functions* $F = \{f : \mathcal{Z}^+ \to \mathcal{Z}^+\}$ *is said to be* downward closed *if for all* $g \in F$ *and for all constants* c, $g'(n) \in O(g(n^c))$ *implies that* $g' \in F$. *A function* g *is said to be* hard *for the family* F *if for all* $g' \in F$, *there exists a constant* c *such that* $g'(n) \in O(g(n^c))$; *the function* g *is said to be* complete *for* F *if* g *is hard for* F *and* $g \in F$.

Definition 13 (F-APX) *For a downward closed family* F, *the class* F-APX *consists of all problems approximable to within a ratio of* $g(|\mathcal{I}|)$ *for some function* $g \in F$.

Definition 14 (Canonical Hardness) *An* NPO *problem* Π *is said to be* canonically hard *for the class* F-APX *if there exists a transformation* T, *constants* n_0 *and* c, *and a gap function* g *which is hard for the family* F, *such that given an instance* x *of* 3-SAT *on* $n \geq n_0$ *variables and* $N \geq n^c$, $\mathcal{I} = T(x, N)$ *is an instance of* Π *with the following properties:*

- $x \in$ 3-SAT $\Rightarrow OPT(\mathcal{I}) = N$.
- $x \notin$ 3-SAT $\Rightarrow OPT(\mathcal{I}) = N/g(N)$.
- *Given a solution* S *to* $\mathcal{I}$ *with* $V(\mathcal{I}, S) > N/g(N)$, *a truth assignment* $\vec{a}$ *satisfying* x *can be found in polynomial time.*

We defer the proof of the following theorem to the final version of this abstract, but note that it is based on a generalization of the proof of Theorem 1.

Theorem 5 *If* F *is a downward closed family of functions, and an additive* NPO *problem* Π *is canonically hard for the class* F-APX, *then all problems in* F-APX E*-reduce to* Π.

The following is a consequence ofTheorem 5.

Theorem 6 **a)** $\overline{\text{RMAX(2)}}$ = poly-APX.
b) *If SET COVER is canonically hard to approximate to within* $\Omega(\log n)$ *factor, then* log-APX $= \overline{\text{MIN F}^+\Pi_2(1)}$.

We briefly sketch the proof of this theorem. The hardness reduction for MAX SAT and CLIQUE are canonical [3, 11]. The classes APX-PB, poly-APX, log-APX are expressible as classes F-APX for downward closed function families. The problems MAX SAT, MAX CLIQUE and SET COVER are additive. Thus, we can now apply Theorem 5.

Remark 6 *We would like to point out that almost all known instances of hardness results seem to be shown for problems which are additive. In particular, this is true for all MAX SNP problems, MAX CLIQUE, CHROMATIC NUMBER, and SET COVER. One case where a hardness result does not seem to directly apply to an additive problem is that of LONGEST PATH [17]. However in this case, the closely related LONGEST* s-t *PATH problem is easily seen to be additive and the hardness result essentially stems from this problem. Lastly, the most interesting optimization problems which do not seem to be additive are problems related to GRAPH BISECTION or PARTITION, and these also happen to be notable instances where no hardness of approximation results have been achieved!*

5 Local Search and MAX SNP

In this section we present a formal definition of the paradigm of non-oblivious local search, and describe how it applies to a generic MAX SNP problem. Given a MAX SNP problem Π, recall that the goal is to find a structure S which maximizes the objective function: $V(\mathcal{I}, S) = \sum_{\vec{x}} \Phi(\mathcal{I}, S, \vec{x})$. In the subsequent discussion, we view S as a k-dimensional boolean vector.

5.1 Classical Local Search

We start by reviewing the standard mechanism for constructing a local search algorithm. A δ-local algorithm $\mathcal{A}$ for Π is based on a *distance function* $\mathcal{D}(S_1, S_2)$ which is the Hamming distance between two k-dimensional vectors. The δ*-neighborhood* of a structure S is given by $N(S, \delta) = \{S' \subseteq U^n \mid D(S, S') \leq \delta\}$, where U is the universe. A structure S is called δ*-optimal* if $\forall S' \in N(S, \delta)$, we have $V(\mathcal{I}, S) \geq V(\mathcal{I}, S')$. The algorithm computes a δ-optimum by performing a series of greedy improvements to an initial structure S_0, where each iteration moves from the current structure S_i to some $S_{i+1} \in N(S_i, \delta)$ of better value (if any). For constant δ, a δ-local search algorithm for a polynomially-bounded NPO problem runs in polynomial time because

- each local change is polynomially computable, and
- the number of iterations is polynomially bounded since the value of the objective function improves monotonically by an integral amount with each iteration, and the optimum is polynomially-bounded.

5.2 Non-Oblivious Local Search

A non-oblivious local search algorithm is based on a 3-tuple $\langle S_0, \mathcal{F}, \mathcal{D}\rangle$ where S_0 is the initial solution structure which must be independent of the input, $\mathcal{F}(\mathcal{I}, S)$ is a real-valued function referred to as the *weight function*, and $\mathcal{D}$ is a real-valued *distance function* which returns the distance between two structures in some appropriately chosen metric. The distance function $\mathcal{D}$ is computable in time polynomial in $|U|$. Thus as before, for constant δ, a non-oblivious δ-local algorithm terminates in time polynomial in the input size.

The classical local search paradigm, which we call *oblivious local search*, makes the natural choice for the function $\mathcal{F}(\mathcal{I}, S)$, and the distance function $\mathcal{D}$, i.e. it chooses them to be $V(\mathcal{I}, S)$ and the Hamming distance. However, as we show later, this choice does not always yield a good approximation ratio. We now formalize our notion of this more general type of local search.

Definition 15 (Non-Oblivious Local Search Algorithm) *A* non-oblivious local search *algorithm is a δ-local search algorithm whose weight function is defined to be*

$$\mathcal{F}(\mathcal{I}, S) = \sum_{\vec{x}} \sum_{i=1}^{r} p_i \Phi_i(\mathcal{I}, S, \vec{x}) ,$$

where r is a constant, Φ_i's are quantifier-free first-order formulas, and the profits p_i are real constants. The distance function $\mathcal{D}$ is an arbitrary polynomial-time computable function.

A non-oblivious local search can be implemented in polynomial time in much the same way as oblivious local search. Note that the we are only considering polynomially-bounded weight functions and the profits p_i are fixed independent of the input size. In general, the non-oblivious weight functions do not direct the search in the direction of the actual objective function. In fact, as we will see, this is exactly the reason why they are more powerful and allow for better approximations.

Definition 16 (Non-Oblivious GLO) *The class of problems* NON-OBLIVIOUS GLO *consists of all problems which can be approximated within constant factors by a non-oblivious δ-local search algorithm for some constant δ.*

Remark 7 *We make some observations about the above definition. It would be perfectly reasonable to allow weight functions which are non-linear, but we stay with the above definition for the purposes of this paper. Allowing only a constant number of predicates in the weight functions enables us to prevent the encoding of arbitrarily complicated approximation algorithms. The structure S is a k-dimensional vector, and so a convenient metric for the distance function $\mathcal{D}$ is the Hamming distance. This should be assumed to be the underlying metric unless otherwise specified. However, we have found that it is sometimes useful to modify this, for example by modifying the Hamming distance so that the complement of a vector is considered to be at distance 1 from it. Finally, it is sometimes convenient to assume that the local search makes the best possible move in the bounded neighborhood, rather than an arbitrary move which improves the weight function. We believe that this does not increase the power of non-oblivious local search.*

6 The Power of Non-Oblivious Local Search

We will show that there exists a choice of a non-oblivious weight function for MAX *k*-SAT such that any assignment which is 1-optimal with respect to this weight function, yields a performance ratio of $2^k/(2^k - 1)$ with respect to the optimal. But first, we obtain tight bounds on the performance of oblivious local search for MAX 2-SAT, establishing that its performance is significantly weaker than the best-known result even when allowed to search exponentially large neighborhoods. We use the following notation: for any fixed truth assignment $\vec{Z}$, S_i is the set of clauses in which exactly i literals are true; and, for a set of clauses S, $W(S)$ denotes the total weight of the clauses in S.

We show a strong separation in the performance of oblivious and non-oblivious local search for MAX 2-SAT. Suppose we use a δ-local strategy with the weight function $\mathcal{F}$ being the total weight of the clauses satisfied by the assignment, i.e., $\mathcal{F} = W(S_1) + W(S_2)$. The following theorem shows that for any $\delta = o(n)$, an oblivious δ-local strategy cannot deliver a performance ratio better than $3/2$. This is rather surprising given that we are willing to allow near-exponential time for the oblivious algorithm.

Theorem 7 *The asymptotic performance ratio for an oblivious δ-local search algorithm for* MAX 2-SAT *is* $3/2$ *for any positive* $\delta = o(n)$. *This ratio is still bounded by* $5/4$ *when* δ *may take any value less than* $n/2$.

We now illustrate the power of non-oblivious local search by showing that it achieves a performance ratio of $4/3$ for MAX 2-SAT, using 1-local search with a simple non-oblivious weight function.

Theorem 8 *Non-oblivious 1-local search achieves a performance ratio of* $4/3$ *for* MAX 2-SAT.

Proof: We use the non-oblivious weight function

$$\mathcal{F}(\mathcal{I}, \vec{Z}) = \frac{3}{2}W(S_1) + 2W(S_2).$$

Consider any assignment $\vec{Z}$ which is 1-optimal with respect to this weight function. Without loss of generality, we assume that the variables have been renamed such that each unnegated literal gets assigned the value true. Let $P_{i,j}$ and $N_{i,j}$ respectively denote the total weight of clauses in S_i containing the literals z_j and $\overline{z}_j$, respectively. Since $\vec{Z}$ is a 1-optimal assignment, each variable z_j must satisfy the following equation.

$$-\frac{1}{2}P_{2,j} - \frac{3}{2}P_{1,j} + \frac{1}{2}N_{1,j} + \frac{3}{2}N_{0,j} \leq 0.$$

Summing this inequality over all the variables, and using

$$\begin{aligned} \sum_{j=1}^{n} P_{1,j} = \sum_{j=1}^{n} N_{1,j} &= W(S_1) \\ \sum_{j=1}^{n} P_{2,j} &= 2W(S_2) \\ \sum_{j=1}^{n} N_{0,j} &= 2W(S_0). \end{aligned}$$

we get the following inequality:

$$W(S_2) + W(S_1) \geq 3W(S_0).$$

This immediately implies that the total weight of the unsatisfied clauses at this local optimum is no more than $1/4$ times the total weight of all the clauses. Thus, this algorithm ensures a performance ratio of $4/3$. ■

Remark 8 *The same result can be achieved by using the oblivious weight function, and instead modifying the distance function so that it corresponds to distances in a hypercube augmented by edges between nodes whose addresses are complement of each other.*

The preceding theorem can be extended to MAX *k*-SAT.

Theorem 9 *Non-oblivious 1-local search achieves a performance ratio of* $2^k/(2^k - 1)$ *for* MAX *k*-SAT.

7 Local Search for CSP and MAX SNP

We now introduce a class of constraint satisfaction problems such that the problems in MAX SNP are exactly equivalent to the problems in this class. Furthermore, every problem in this class can be approximated to within a constant factor by a non-oblivious local search algorithm.

The connection between the syntactic description of optimization problems and their approximability through non-oblivious local search is made via a problem called MAX *k*-CSP which captures all the problems in MAX SNP as a special case.

Definition 17 (k-ary Constraint) *Let* $Z = \{z_1, \ldots, z_n\}$ *be a set of boolean variables. A* k*-ary constraint on* Z *is* $C = (V; P)$, *where* V *is a size* k *subset of* Z, *and* $P : \{T, F\}^k \rightarrow \{T, F\}$ *is a* k*-ary boolean predicate.* ■

Definition 18 (MAX k-CSP) *Given a collection* $C_1, \ldots, C_m$ *of weighted* k*-ary constraints over the variables* $Z = \{z_1, \ldots, z_n\}$, *the* MAX *k*-CSP *problem is to find a truth assignment satisfying a maximum weight sub-collection of the constraints.* ■

The following theorem shows that MAX *k*-CSP problem is a "universal" MAX SNP problem, in that it contains as special cases all problems in MAX SNP.

Theorem 10 **a)** *For fixed* k, MAX *k*-CSP $\in$ MAX SNP.
b) *Let* $\Pi \in$ MAX SNP. *Then, for some constant* k, $\Pi \in$ MAX *k*-CSP. *Moreover, the* k-CSP *instance corresponding to any instance of this problem can be computed in polynomial time.*

A suitable generalization of the non-oblivious local search algorithm for MAX *k*-SAT yields the following result.

Theorem 11 *A non-oblivious 1-local search algorithm has performance ratio* 2^k *for* MAX *k*-CSP.

Theorem 12 *Every optimization problem* $\Pi \in$ MAX SNP *can be approximated to within some constant factor by a (uniform) non-oblivious 1-local search algorithm, i.e.,*

$$\text{MAX SNP} \subseteq \text{NON-OBLIVIOUS GLO}.$$

For a problem expressible as k-CSP*, the performance ratio is at most* 2^k.

8 Non-Oblivious versus Oblivious GLO

In this section, we show that there exist problems for which no constant factor approximation can be obtained by any δ-local search algorithm with oblivious weight function, even when we allow δ to grow with the input size. However, a simple 1-local search algorithm using an appropriate non-oblivious weight function can ensure a constant performance ratio.

8.1 MAX 2-CSP

The first problem is an instance of MAX 2-CSP where we are given a collection of monomials such that each monomial is an "and" of precisely two literals. The objective is to find an assignment to maximize the number of monomials satisfied.

We show an instance of this problem such that for every $\delta = o(n)$, there exists an instance one of whose local optima has value that is a vanishingly small fraction of the global optimum.

The input instance consists of a disjoint union of two sets of monomials, say Γ_1 and Γ_2, defined as below:

$$\Gamma_1 = \bigvee_{1 \le i < j \le n} (\overline{z}_i \wedge \overline{z}_j)$$
$$\Gamma_2 = \bigvee_{1 \le i \le \delta} \bigvee_{i < j \le n} (z_i \wedge z_j)$$

Clearly, $|\Gamma_1| = \binom{n}{2}$, and $|\Gamma_2| = n\delta - \binom{\delta+1}{2}$. Consider the truth assignment $\vec{Z} = (1, 1, \ldots, 1)$. It satisfies all monomials in Γ_2 but none of the monomials in Γ_1. We claim that this assignment is δ-optimal with respect to the oblivious weight function. To see this, observe that complementing the value of any $p \le \delta$ variables will unsatisfy at least $\delta p/2$ monomials in Γ_2 for any $\delta = o(n)$. On the other hand, this will satisfy precisely $\binom{p}{2}$ monomials in Γ_1. For any $p \le \delta$, we have $(\delta p)/2 \ge \binom{p}{2}$, and so Z is a δ-local optimum.

The optimal assignment on the other hand, namely $\vec{Z}_{OPT} = (0, 0, \ldots, 0)$, satisfies all monomials in Γ_1. Thus, for $\delta < n/2$, the performance ratio achieved by any δ-local algorithm is no more than $\binom{n}{2}/(n\delta - \binom{\delta+1}{2})$ which asymptotically diverges to infinity for any $\delta = o(n)$. We have already seen in Section 7 that a 1-local non-oblivious algorithm ensures a performance ratio of 4 for this problem. Since this problem is in MAX SNP, we obtain the following theorem.

Theorem 13 *There exist problems in* MAX SNP *such that for* $\delta = o(n)$*, no* δ*-local oblivious algorithm can approximate them to within a constant performance ratio, i.e.,*

$$\text{MAX SNP} \not\subseteq \text{GLO}.$$

8.2 Vertex Cover

Ausiello-Protasi [4] have shown that VERTEX COVER does not belong to the class GLO and, hence, there does not exist any constant δ such that an oblivious δ-local search algorithm can compute a constant factor approximation. In fact, their example can be used to show that for any $\delta = o(n)$, the performance ratio ensured by δ-local search asymptotically diverges to infinity. However, we show that there exists a rather simple non-oblivious weight function which ensures a factor 2 approximation via a 1-local search. In fact, the algorithm simply enforces the behavior of the standard approximation algorithm which iteratively builds a vertex cover by simply including both end-points of any currently uncovered edge.

We assume that the input graph G is given as a structure $(V, \{E\})$ where V is the set of vertices and $E \subseteq V \times V$ encodes the edges of the graph. Our solution is represented by a 2-ary predicate M which is iteratively constructed so as to represent a maximal matching. Clearly, the end-points of any maximal matching constitute a valid vertex cover and such a vertex cover can be at most twice as large as any other vertex cover in the graph. Thus M is an encoding of the vertex cover computed by the algorithm.

The algorithm starts with M initialized to the empty relation and at each iteration, at most one new pair is included in it. The non-oblivious weight function used is as below:

$$\mathcal{F}(\mathcal{I}, M) = \sum_{\langle x,y,z \rangle \in V^3} \frac{1}{2}\Phi_1(x,y,z) - \Phi_2(x,y,z),$$

where

$$\Phi_1(x,y,z) = (M(x,y) \wedge E(x,y) \wedge (x = z)),$$
$$\Phi_2(x,y,z) = (M(x,y) \wedge M(x,z)).$$

It is not difficult to show that if we start with the empty matching, any 1-optimal relation M always encodes a maximal matching in G. We have established the following.

Theorem 14 *A 1-local search algorithm using the above non-oblivious weight function achieves a performance ratio of* 2 *for the VERTEX COVER problem.*

Theorem 15 GLO *is a* strict *subset of* NON-OBLIVIOUS GLO.

9 The Traveling Salesman Problem

The TSP(1,2) problem is the traveling salesman problem restricted to complete graphs where all edge weights are either 1 or 2; clearly, this satisfies the triangle inequality. Papadimitriou and Yannakakis [22] showed that this problem is hard for MAX SNP. The natural weight function for TSP(1,2), that is, the weight of the tour, can be used to show that a 4-local algorithm yields a $\frac{3}{2}$ performance ratio. The algorithm starts with an arbitrary tour and in each iteration, it checks if there exist two disjoint edges (a, b) and (c, d) on the tour such that deleting them and replacing them with the edges (a, c) and (b, d) yields a tour of lesser cost.

Theorem 16 *A 4-local search algorithm using the oblivious weight function achieves a $\frac{3}{2}$ performance ratio for* TSP(1,2)*.*

The above bound can be shown to be tight.

Theorem 17 *There exists a* TSP(1,2) *instance such that the optimal solution has cost $n + O(1)$ and there exists a certain 4-optimal solution for it with cost $\frac{3}{2}n + O(1)$.*

10 Maximum Independent Sets in Bounded Degree Graphs

The input instance to the maximum independent set problem in bounded degree graphs, denoted MIS-B, is a graph G such that the degree of any vertex in G is bounded by a constant Δ. We present an algorithm with performance ratio $(\sqrt{8\Delta^2 + 4\Delta + 1} - 2\Delta + 1)/2$ for this problem when $\Delta \geq 10$.

Our algorithm uses two local search algorithms such that the larger of the two independent sets computed by these algorithms, gives us the above claimed performance ratio. We refer to these two algorithms as $\mathcal{A}_1$ and $\mathcal{A}_2$.

In our framework, the algorithm $\mathcal{A}_1$ can be characterized as a 3-local algorithm. Let an $i \leftrightarrow j$ swap be the process of deleting i vertices from S (the current independent set) and including j vertices from the set $V - S$ to the set S. Then in each iteration, the algorithm $\mathcal{A}_1$ performs either a $0 \leftrightarrow j$ swap where $1 \leq j \leq 3$, or a $1 \leftrightarrow 2$ swap; it terminates when neither operation applies.

Lemma 1 *The algorithm $\mathcal{A}_1$ has performance ratio $(\Delta + 1)/2$ for* MIS-B.

This nearly matches the approximation ratio of $2/\Delta$ due to Hochbaum [15]. It may be noted that the above result holds for a broader class of graphs, namely, k-claw free graphs. A graph is called *k-claw free* if there does not exist an independent set of size k or larger such that all the vertices in the independent set are adjacent to the same vertex. Lemma 1 applies to $(\Delta + 1)$-claw free graphs.

Our next objective is to further improve this ratio by using the algorithm $\mathcal{A}_1$ in combination with the algorithm $\mathcal{A}_2$. The following lemma uses a slightly different counting argument to give an alternative bound on the approximation ratio of the algorithm $\mathcal{A}_1$ when there is a constraint on the size of the optimal solution.

Lemma 2 *For any real number $c < \Delta$, the algorithm $\mathcal{A}_1$ has performance ratio $(\Delta - c)/2$ for* MIS-B *when the optimal value itself is no more than $((\Delta - c)|V|)/(\Delta + c + 4)$.*

The above lemma shows that the algorithm $\mathcal{A}_1$ yields a better approximation ratio when the size of the optimal independent set is relatively small.

The algorithm $\mathcal{A}_2$ is simply the classical greedy algorithm. This algorithm can be conveniently included in our framework if we use directed local search. The following two lemmas characterize the performance of the greedy algorithm.

Lemma 3 *Suppose there exists an independent set $X \subseteq V$ such that the average degree of vertices in X is bounded by α. Then for any $\alpha \geq 1$, the greedy algorithm produces an independent set of size at least $|X|/(1 + \alpha)$.*

Lemma 4 *For $\Delta \geq 10$ and any non-negative real number $c \leq 3\Delta - \sqrt{8\Delta^2 + 4\Delta + 1} - 1$, the algorithm $\mathcal{A}_2$ has performance ratio $(\Delta - c)/2$ for* MIS-B *when the optimal value itself is at least $((\Delta - c)|V|)/(\Delta + c + 4)$.*

Combining the results of Lemmas 2 and 4 and choosing the largest allowable value for c, we get the following result.

Theorem 18 *An approximation algorithm which simply outputs the larger of the two independent sets computed by the algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$, has performance ratio $(\sqrt{8\Delta^2 + 4\Delta + 1} - 2\Delta + 1)/2$ for* MIS-B.

The performance ratio claimed above is essentially $\Delta/2.414$. This improves upon the long-standing approximation ratio of $\Delta/2$ due to Hochbaum [15] when $\Delta \geq 10$. However, very recently, there has been a flurry of new results for this problem. Berman and Furer [6] have given an algorithm with performance ratio $(\Delta+3)/5+\epsilon$ when Δ is even, and $(\Delta+3.25)/5+\epsilon$ for odd Δ, where $\epsilon > 0$ is a fixed constant. Halldorsson and Radhakrishnan [14] have shown that algorithm $\mathcal{A}_1$ when run on k-clique free graphs, yields an independent set of size at least $\frac{2}{\Delta+k}n$. They combine this algorithm with a clique-removal based scheme to achieve a performance ratio of $\Delta/6(1+o(1))$.

Acknowledgements

Many thanks to Phokion Kolaitis for his helpful comments and suggestions. Thanks also to Giorgio Ausiello and Pierluigi Crescenzi for guiding us through the intricacies of approximation preserving reductions and the available literature on it.

References

[1] P. Alimonti. New Local Search Approximation Techniques for Maximum Generalized Satisfiability Problems. In *Proceedings of the 2nd Italian Conference on Algorithms and Complexity* (1994, to appear).

[2] G. Ausiello, P. Crescenzi and M. Protasi. Approximate Solution of NP Optimization Problems. Rapporto di Ricerca, SI/RR-94/03, Dipartimento di Scienze dell'Informazione, Universita degli Studi di Roma (March 1994).

[3] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Hardness of Approximation Problems. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science* (1992), pp. 14–23.

[4] G. Ausiello and M. Protasi. NP Optimization Problems and Local Optima. Unpublished Manuscript (1992).

[5] M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing* (1993), pp. 294–304.

[6] P. Berman and M. Furer. Approximating Maximum Independent Set in Bounded Degree Graphs. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms* (1993), pp. 365–371.

[7] D.P. Bovet and P. Crescenzi. **Introduction to the Theory of Complexity.** Prentice-Hall, New York (1993).

[8] P. Crescenzi and L. Trevisan. Manuscript (1994).

[9] P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, vol. 93 (1991), pp. 241–262.

[10] R. Fagin. Generalized First-Order Spectra and Polynomial-time Recognizable Sets. In Richard Karp (ed.), **Complexity of Computer Computations**, AMS, 1974.

[11] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science* (1991), pp. 2–12.

[12] Michael R. Garey and David S. Johnson. **Computers and Intractability: A Guide to the Theory of NP-Completeness**. W. H. Freeman, 1979.

[13] Michel X. Goemans and David P. Williamson. .878-Approximation Algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the Twenty-Sixth ACM Symposium on Theory of Computing* (1994), pp. 422-431.

[14] M. M. Halldorsson and J. Radhakrishnan. Improved Approximations of Independent Sets in Bounded-Degree Graphs. In *Proceedings of the 4th Scandanavian Workshop on Algorithm Theory* (1994), pp. 194–206.

[15] Dorit S. Hochbaum. Efficient bounds for the stable set, vertex cover, and set packing problems. *Discrete Applied Mathematics*, vol. 6 (1982), pp. 243–254.

[16] Viggo Kann. **On the Approximability of NP-complete Optimization Problems**. Ph.D. Thesis, Department of Numerical Analysis and Computing Science. Royal Institute of Technology, Stockholm, Sweden (May 1992).

[17] D. Karger, R. Motwani and G.D.S. Ramkumar. On approximating the longest path in a graph. In *Proceedings of the Third Workshop on Algorithms and Data Structures* (1993), pp. 421–432.

[18] Phokion G. Kolaitis and Madhukar N. Thakur. Approximation Properties of NP Minimization Classes. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference* (1991), pp. 353–366.

[19] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing* (1993), pp. 286–293.

[20] A. Panconesi and D. Ranjan. Quantifiers and Approximation. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing* (1990), pp. 446–456.

[21] C. H. Papadimitriou and M. Yannakakis. Optimization, Approximation, and Complexity Classes. *Journal of Computer and System Sciences*, vol. 43 (1991), pp. 425–440.

[22] C. H. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, vol. 18 (1993), pp. 1–11.

[23] M. Yannakakis. The analysis of local search problems and their heuristics. In *Proceedings of the 7th Annual Symposium of Theoretical Aspects of Computer Science* (1990), pp. 298–311.