# Sub-exponential Approximation Schemes for CSPs: From Dense to Almost Sparse*

**Dimitris Fotakis[1], Michael Lampis[2], and Vangelis Th. Paschos[3]**

1 **School of Electrical and Computer Engineering, National Technical University of Athens, Greece**
fotakis@cs.ntua.gr

2 **PSL Research University, Université Paris Dauphine, LAMSADE, CNRS UMR7243, France**
michail.lampis@dauphine.fr

3 **PSL Research University, Université Paris Dauphine, LAMSADE, CNRS UMR7243, France**
paschos@lamsade.dauphine.fr

## Abstract

It has long been known, since the classical work of (Arora, Karger, Karpinski, JCSS 99), that Max-Cut admits a PTAS on dense graphs, and more generally, Max-$k$-CSP admits a PTAS on "dense" instances with $\Omega(n^k)$ constraints. In this paper we extend and generalize their exhaustive sampling approach, presenting a framework for $(1-\varepsilon)$-approximating any Max-$k$-CSP problem in *sub-exponential* time while significantly relaxing the denseness requirement on the input instance.

Specifically, we prove that for any constants $\delta \in (0,1]$ and $\varepsilon > 0$, we can approximate Max-$k$-CSP problems with $\Omega(n^{k-1+\delta})$ constraints within a factor of $(1-\varepsilon)$ in time $2^{O(n^{1-\delta}\ln n/\varepsilon^3)}$. The framework is quite general and includes classical optimization problems, such as Max-Cut, Max-DICUT, Max-$k$-SAT, and (with a slight extension) $k$-Densest Subgraph, as special cases. For Max-Cut in particular (where $k=2$), it gives an approximation scheme that runs in time sub-exponential in $n$ even for "almost-sparse" instances (graphs with $n^{1+\delta}$ edges).

We prove that our results are essentially best possible, assuming the ETH. First, the density requirement cannot be relaxed further: there exists a constant $r < 1$ such that for all $\delta > 0$, Max-$k$-SAT instances with $O(n^{k-1})$ clauses cannot be approximated within a ratio better than $r$ in time $2^{O(n^{1-\delta})}$. Second, the running time of our algorithm is almost tight *for all densities*. Even for Max-Cut there exists $r < 1$ such that for all $\delta' > \delta > 0$, Max-Cut instances with $n^{1+\delta}$ edges cannot be approximated within a ratio better than $r$ in time $2^{n^{1-\delta'}}$.

**1998 ACM Subject Classification** F.1.3 Complexity Measures and Classes, F.2.2 Nonnumerical Algorithms and Problems, G.2.1 Combinatorics, G.2.2 Graph Theory

**Keywords and phrases** polynomial and subexponential approximation, sampling, randomized rounding

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2016.37

* This research was supported by the project AlgoNow, co-financed by the European Union (European Social Fund – ESF) and Greek national funds, through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: THALES, investing in knowledge society through the European Social Fund.



33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016).
Editors: Nicolas Ollinger and Heribert Vollmer; Article No. 37; pp. 37:1–37:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The complexity of Constraint Satisfaction Problems (CSPs) has long played a central role in Theoretical Computer Science and it quickly became evident that almost all interesting CSPs are NP-complete [27]. Thus, since approximation algorithms are one of the standard tools for dealing with NP-hard problems, the question of approximating the corresponding optimization problems (Max-CSP) has attracted significant interest over the years [28]. Unfortunately, most CSPs typically resist this approach: not only are they APX-hard [23], but quite often the best polynomial-time approximation ratio we can hope to achieve for them is that guaranteed by a trivial random assignment [21]. This striking behavior is often called *approximation resistance*.

Approximation resistance and other APX-hardness results were originally formulated in the context of *polynomial-time* approximation. It would therefore seem that one conceivable way for working around such barriers could be to consider approximation algorithms running in super-polynomial time, and indeed super-polynomial approximation for NP-hard problems is a topic that has been gaining more attention in the literature recently [10, 7, 6, 11, 12, 13]. Unfortunately, the existence of quasi-linear PCPs with small soundness error, first given in the work of Moshkovitz and Raz [24], established that approximation resistance is a phenomenon that carries over even to *sub-exponential* time approximation, essentially "killing" this approach for CSPs. For instance, we now know that if, for any $\varepsilon > 0$, there exists an algorithm for Max-3-SAT with ratio $7/8 + \varepsilon$ running in time $2^{n^{1-\varepsilon}}$ this would imply the existence of a sub-exponential *exact* algorithm for 3-SAT, disproving the Exponential Time Hypothesis (ETH). It therefore seems that sub-exponential time does not improve the approximability of CSPs, or put another way, for many CSPs obtaining a very good approximation ratio requires almost as much time as solving the problem exactly.

Despite this grim overall picture, many positive approximation results for CSPs have appeared over the years, by taking advantage of the special structure of various classes of instances. One notable line of research in this vein is the work on the approximability of *dense* CSPs, initiated by Arora, Karger and Karpinski [4] and independently by de la Vega [14]. The theme of this set of results is that the problem of maximizing the number of satisfied constraints in a CSP instance with arity $k$ (Max-$k$-CSP) becomes significantly easier if the instance contains $\Omega(n^k)$ constraints. More precisely, it was shown in [4] that Max-$k$-CSP admits a *polynomial-time approximation scheme* (PTAS) on dense instances, that is, an algorithm which for any constant $\varepsilon > 0$ can in time polynomial in $n$ produce an assignment that satisfies $(1 - \varepsilon)$OPT constraints. Subsequent work produced a stream of positive [16, 5, 2, 9, 8, 20, 3, 19, 22] (and some negative [15, 1]) results on approximating CSPs which are in general APX-hard, showing that dense instances form an island of tractability where many optimization problems which are normally APX-hard admit a PTAS.

**Our contribution.** The main goal of this paper is to use the additional power afforded by sub-exponential time to extend this island of tractability as much as possible. To demonstrate the main result, consider a concrete CSP such as Max-3-SAT. As mentioned, we know that sub-exponential time does not in general help us approximate this problem: the best ratio achievable in, say, $2^{\sqrt{n}}$ time is still $7/8$. On the other hand, this problem admits a PTAS on instances with $\Omega(n^3)$ clauses. This density condition is, however, rather strict, so the question we would like to answer is the following: Can we efficiently approximate a larger (and more sparse) class of instances while using sub-exponential time?

In this paper we provide a positive answer to this question, not just for Max-3-SAT, but also for any Max-$k$-CSP problem. Specifically, we show that for any constants $\delta \in (0, 1]$,

$\varepsilon > 0$ and integer $k \geq 2$, there is an algorithm which achieves a $(1-\varepsilon)$ approximation of Max-$k$-CSP instances with $\Omega(n^{k-1+\delta})$ constraints in time $2^{O(n^{1-\delta} \ln n/\varepsilon^3)}$. A notable special case of this result is for $k = 2$, where the input instance can be described as a graph. For this case, which contains classical problems such as Max-Cut, our algorithm gives an approximation scheme running in time $2^{O(\frac{n}{\Delta} \ln n/\varepsilon^3)}$ for graphs with average degree $\Delta$. In other words, this is an approximation scheme that runs in time *sub-exponential in* $n$ even for almost sparse instances where the average degree is $\Delta = n^{\delta}$ for some small $\delta > 0$. More generally, our algorithm provides a trade-off between the time available and the density of the instances we can handle. For graph problems ($k = 2$) this trade-off covers the whole spectrum from dense to almost sparse instances, while for general Max-$k$-CSP, it covers instances where the number of constraints ranges from $\Theta(n^k)$ to $\Theta(n^{k-1})$.

**Techniques.** The algorithms in this paper are an extension and generalization of the *exhaustive sampling* technique given by Arora, Karger and Karpinski [4], who introduced a framework of smooth polynomial integer programs to give a PTAS for dense Max-$k$-CSP. The basic idea of that work can most simply be summarized for Max-Cut. This problem can be recast as the problem of maximizing a quadratic function over $n$ boolean variables. This is of course a hard problem, but suppose that we could somehow "guess" for each vertex how many of its neighbors belong in each side of the cut. This would make the quadratic problem linear, and thus much easier. The main intuition now is that, if the graph is dense, we can take a sample of $O(\log n)$ vertices and guess their partition in the optimal solution. Because every non-sample vertex will have "many" neighbors in this sample, we can with high confidence say that we can estimate the fraction of neighbors on each side for all vertices. The work of de la Vega [14] uses exactly this algorithm for Max-Cut, greedily deciding the vertices outside the sample. The work of [4] on the other hand pushed this idea to its logical conclusion, showing that it can be applied to degree-$k$ polynomial optimization problems, by recursively turning them into linear programs whose coefficients are estimated from the sample. The linear programs are then relaxed to produce fractional solutions, which can be rounded back into an integer solution to the original problem.

On a very high level, the approach we follow in this paper retraces the steps of [4]: we formulate Max-$k$-CSP as a degree-$k$ polynomial maximization problem; we then recursively decompose the degree-$k$ polynomial problem into lower-degree polynomial optimization problems, estimating the coefficients by using a sample of variables for which we try all assignments; the result of this process is an integer linear program, for which we obtain a fractional solution in polynomial time; we then perform randomized rounding to obtain an integer solution that we can use for the original problem.

The first major difference between our approach and [4] is of course that we need to use a larger sample. This becomes evident if one considers Max-Cut on graphs with average degree $\Delta$. In order to get the sampling scheme to work we must be able to guarantee that each vertex outside the sample has "many" neighbors inside the sample, so we can safely estimate how many of them end up on each side of the cut. For this, we need a sample of size at least $n \log n/\Delta$. Indeed, we use a sample of roughly this size, and exhausting all assignments to the sample is what dominates the running time of our algorithm. As we argue later, not only is the sample size we use essentially tight, but more generally the running time of our algorithm is essentially optimal (under the ETH).

Nevertheless, using a larger sample is not in itself sufficient to extend the scheme of [4] to non-dense instances. As observed in [4] "to achieve a multiplicative approximation for dense instances it suffices to achieve an additive approximation for the nonlinear integer

programming problem". In other words, one of the basic ingredients of the analysis of [4] is that additive approximation errors of the order $\varepsilon n^k$ can be swept under the rug, because we know that in a dense instance the optimal solution has value $\Omega(n^k)$. This is *not* true in our case, and we are therefore forced to give a more refined analysis of the error of our scheme, independently bounding the error introduced in the first step (coefficient estimation) and the last (randomized rounding).

A further, more serious complication arises when considering Max-$k$-CSP for $k > 2$. Indeed, the idea of using a larger sample size to handle non-dense instances of Max-Cut was already considered in [17], which for $k = 2$ gives an algorithm of similar performance to the one we present in this paper. The main obstacle to extending such an algorithm to the general case of $k > 2$ is that the scheme of [4] recursively decomposes dense instances into lower-order polynomials which roughly retain the same "denseness". This property seems much harder to transfer to the non-dense case, because intuitively if we start from a non-dense instance the decomposition could end up producing some dense and some sparse sub-problems. Here we present a scheme that approximates Max-$k$-CSP with $\Omega(n^{k-1+\delta})$ constraints, but does not seem to extend to instances with fewer than $n^{k-1}$ constraints. As we will see, there seems to be a fundamental complexity-theoretic justification explaining exactly why this decomposition method cannot be extended further.

**Hardness.** What makes the results of this paper more interesting is that we can establish that in many ways they are essentially best possible, if one assumes the ETH. In particular, there are at least two ways in which one may try to improve on these results further: one would be to improve the running time of our algorithm, while another would be to extend the algorithm to the range of densities it cannot currently handle. In Section 6 we show that both of these approaches would face significant barriers. Our starting point is the fact that (under ETH) it takes exponential time to approximate Max-Cut arbitrarily well on sparse instances, which is a consequence of the existence of quasi-linear PCPs. By manipulating such Max-Cut instances, we are able to show that for *any* average degree $\Delta = n^\delta$ with $\delta < 1$ the time needed to approximate Max-Cut arbitrarily well almost matches the performance of our algorithm. Furthermore, starting from sparse Max-Cut instances, we can produce instances of Max-$k$-SAT with $O(n^{k-1})$ clauses while preserving hardness of approximation. This gives a complexity-theoretic justification for our difficulties in decomposing Max-$k$-CSP instances with less than $n^{k-1}$ constraints. We note that the hardness results we present here refute (subject to the ETH) a conjecture made in [17] that a better time-density trade-off can be achieved for Max-Cut.

## 2 Notation and Preliminaries

An $n$-variate degree-$d$ polynomial $p(\vec{x})$ is *$\beta$-smooth* [4], for some constant $\beta \geq 1$, if for every $\ell \in \{0, \ldots, d\}$, the absolute value of each coefficient of each degree-$\ell$ monomial in the expansion of $p(\vec{x})$ is at most $\beta n^{d-\ell}$. An $n$-variate degree-$d$ $\beta$-smooth polynomial $p(\vec{x})$ is *$\delta$-bounded*, for some constant $\delta \in (0, 1]$, if for every $\ell$, the sum, over all degree-$\ell$ monomials in $p(\vec{x})$, of the absolute values of their coefficients is $O(\beta n^{d-1+\delta})$. Therefore, for any $n$-variate degree-$d$ $\beta$-smooth $\delta$-bounded polynomial $p(\vec{x})$ and any $\vec{x} \in \{0,1\}^n$, $|p(\vec{x})| = O(d\beta n^{d-1+\delta})$.

Our algorithms for Max-Cut, Max-$k$-SAT, and Max-$k$-CSP are obtained by reducing to the following problem: Given an $n$-variate $d$-degree $\beta$-smooth $\delta$-bounded polynomial $p(\vec{x})$, we seek a binary vector $\vec{x}^* \in \{0,1\}^n$ that maximizes $p$.

As in [4, Lemma 3.1], our general approach is motivated by the fact that any $n$-variate

$d$-degree $\beta$-smooth polynomial $p(\vec{x})$ can be naturally decomposed into a collection of $n$ polynomials $p_j(\vec{x})$. Each of them has degree $d-1$ and at most $n$ variables and is $\beta$-smooth.

▶ **Lemma 1** ([4]). *Let $p(\vec{x})$ be any $n$-variate degree-$d$ $\beta$-smooth polynomial. Then, there exist a constant $c$ and degree-$(d-1)$ $\beta$-smooth polynomials $p_j(\vec{x})$ such that $p(\vec{x}) = c + \sum_{j=1}^{n} x_j p_j(\vec{x})$.*

Let $G(V, E)$ be a (simple) graph with $n$ vertices and $m$ edges. For each vertex $i \in V$, $N(i)$ denotes $i$'s neighborhood in $G$, i.e., $N(i) = \{j \in V : \{i, j\} \in E\}$. We let $\deg(i) = |N(i)|$ be the degree of $i$ in $G$ and $\Delta = 2|E|/n$ denote the average degree of $G$. We say that a graph $G$ is $\delta$-*almost sparse*, for some constant $\delta \in (0, 1]$, if $m = \Omega(n^{1+\delta})$ (and thus, $\Delta = \Omega(n^{\delta})$).

In Max-Cut, we seek a partitioning of the vertices of $G$ into two sets $S_0$ and $S_1$ so that the number of edges with endpoints in $S_0$ and $S_1$ is maximized. If $G$ has $m$ edges, the number of edges in the optimal cut is at least $m/2$.

In $k$-Densest Subgraph, given an undirected graph $G(V, E)$, we seek a subset $C$ of $k$ vertices so that the induced subgraph $G[C]$ has a maximum number of edges.

An instance of (boolean) Max-$k$-CSP with $n$ variables consists of $m$ boolean constraints $f_1, \ldots, f_m$, where each $f_j : \{0, 1\}^k \to \{0, 1\}$ depends on $k$ variables and is satisfiable, i.e., $f_j$ evaluates to 1 for some truth assignment. We seek a truth assignment to the variables that maximizes the number of satisfied constraints. Max-$k$-SAT is a special case of Max-$k$-CSP where each constraint $f_j$ is a disjunction of $k$ literals. An averaging argument implies that the optimal assignment of a Max-$k$-CSP (resp. Max-$k$-SAT) instance with $m$ constraints satisfies at least $2^{-k}m$ (resp. $(1-2^{-k})m$) of them. We say that an instance of Max-$k$-CSP is $\delta$-*almost sparse*, for some constant $\delta \in (0, 1]$, if the number of constraints is $m = \Omega(n^{k-1+\delta})$.

Using standard arithmetization techniques (see e.g., [4, Sec. 4.3]), we can reduce any instance of Max-$k$-CSP with $n$ variables to an $n$-variate degree-$k$ polynomial $p(\vec{x})$ so that the optimal truth assignment for Max-$k$-CSP corresponds to a maximizer $\vec{x}^* \in \{0, 1\}$ of $p(\vec{x})$ and the value of the optimal Max-$k$-CSP solution is equal to $p(\vec{x}^*)$. Such a polynomial $p(\vec{x})$ is $\beta$-smooth, for an appropriate constant $\beta$ that may depend on $k$, and has at least $m$ and at most $4^k m$ monomials. Moreover, if the instance of Max-$k$-CSP has $m = \Theta(n^{k-1+\delta})$ constraints, then $p(\vec{x})$ is $\delta$-bounded and its maximizer $\vec{x}^*$ has $p(\vec{x}^*) = \Omega(n^{k-1+\delta})$.

Since each $k$-tuple of variables can appear in at most $2^k$ different constraints, $p(\vec{x})$ is $\beta$-smooth, for $\beta \in [1, 4^k]$, and has at least $m$ and at most $4^k m$ monomials. Moreover, if the instance of Max-$k$-CSP has $m = \Theta(n^{k-1+\delta})$ constraints, then $p(\vec{x})$ is $\delta$-bounded and its maximizer $\vec{x}^*$ has $p(\vec{x}^*) = \Omega(n^{k-1+\delta})$.

An algorithm has *approximation ratio* $\rho \in (0, 1]$ (or is $\rho$-*approximate*) if for all instances, the value of its solution is at least $\rho$ times the value of the optimal solution. For graphs with $n$ vertices or CSPs with $n$ variables, we say that an event $E$ happens with high probability (or whp.), if $E$ happens with probability at least $1 - 1/n^c$, for some constant $c \geq 1$. For brevity and clarity, we sometimes write $\alpha \in (1 \pm \epsilon_1)\beta \pm \epsilon_2\gamma$, for some constants $\epsilon_1, \epsilon_2 > 0$, to denote that $(1-\epsilon_1)\beta - \epsilon_2\gamma \leq \alpha \leq (1+\epsilon_1)\beta + \epsilon_2\gamma$.

## 3 Approximating Max-CUT in Almost Sparse Graphs

In this section, we apply our approach to Max-Cut, which serves as a convenient example and allows us to present the intuition and the main ideas.

The Max-Cut problem in a graph $G(V, E)$ is equivalent to maximizing, over all binary vectors $\vec{x} \in \{0, 1\}^n$, the following $n$-variate degree-2 2-smooth polynomial

$$p(\vec{x}) = \sum_{\{i,j\} \in E} \left(x_i(1 - x_j) + x_j(1 - x_i)\right).$$

Setting a variable $x_i$ to 0 indicates that the corresponding vertex $i$ is assigned to the left side of the cut, i.e., to $S_0$, and setting $x_i$ to 1 indicates that vertex $i$ is assigned to the right side of the cut, i.e., to $S_1$. We assume that $G$ is $\delta$-almost sparse and thus, has $m = \Omega(n^{1+\delta})$ edges and average degree $\Delta = \Omega(n^{\delta})$. Moreover, if $m = \Theta(n^{1+\delta})$, $p(\vec{x})$ is $\delta$-bounded, since for each edge $\{i,j\} \in E$, the monomial $x_i x_j$ appears with coefficient $-2$ in the expansion of $p$, and for each vertex $i \in V$, the monomial $x_i$ appears with coefficient $\deg(i)$ in the expansion of $p$. Therefore, for $\ell \in \{1,2\}$, the sum of the absolute values of the coefficients of all monomials of degree $\ell$ is at most $2m = O(n^{1+\delta})$.

Next, we extend and generalize the approach of [4] and show how to $(1-\varepsilon)$-approximate the optimal cut, for any constant $\varepsilon > 0$, in time $2^{O(n \ln n/(\Delta\varepsilon^3))}$ (see Theorem 5). The running time is subexponential in $n$, if $G$ is $\delta$-almost sparse.

## 3.1 Outline and Main Ideas

Applying Lemma 1, we can write the smooth polynomial $p(\vec{x})$ as

$$p(\vec{x}) = \sum_{j \in V} x_j(\deg(j) - p_j(\vec{x}))\,, \tag{1}$$

where $p_j(\vec{x}) = \sum_{i \in N(j)} x_i$ is a degree-1 1-smooth polynomial that indicates how many neighbors of vertex $j$ are in $S_1$ in the solution corresponding to $\vec{x}$. The key observation, due to [4], is that if we have a good estimation $\rho_j$ of the value of each $p_j$ at the optimal solution $\vec{x}^*$, then approximate maximization of $p(\vec{x})$ can be reduced to the solution of the following Integer Linear Program:

$$\max \sum_{j \in V} y_j(\deg(j) - \rho_j) \tag{IP}$$

$$\text{s.t.} \quad (1-\epsilon_1)\rho_j - \epsilon_2\Delta \le \sum_{i \in N(j)} y_i \le (1+\epsilon_1)\rho_j + \epsilon_2\Delta \quad \forall j \in V$$

$$y_j \in \{0,1\} \qquad \forall j \in V$$

The constants $\epsilon_1, \epsilon_2 > 0$ and the estimations $\rho_j \ge 0$ are computed so that the optimal solution $\vec{x}^*$ is a feasible solution to (IP). We always assume wlog. that $0 \le \sum_{i \in N(j)} y_i \le \deg(j)$, i.e., we let the lhs of the $j$-th constraint be $\max\{(1-\epsilon_1)\rho_j - \epsilon_2\Delta, 0\}$ and the rhs be $\min\{(1+\epsilon_1)\rho_j + \epsilon_2\Delta, \deg(j)\}$. Clearly, if $\vec{x}^*$ is a feasible solution to (IP), it remains a feasible solution after this modification. We let (LP) denote the Linear Programming relaxation of (IP), where each $y_j \in [0,1]$.

The first important observation is that for any $\epsilon_1, \epsilon_2 > 0$, we can compute estimations $\rho_j$, by exhaustive sampling, so that $\vec{x}^*$ is a feasible solution to (IP) with high probability (see Lemma 2). The second important observation is that the objective value of any feasible solution $\vec{y}$ to (LP) is close to $p(\vec{y})$ (see Lemma 3). Namely, for any feasible solution $\vec{y}$, $\sum_{j \in V} y_j(\deg(j) - \rho_j) \approx p(\vec{y})$.

Based on these observations, the approximation algorithm performs the following steps:

1. We guess a sequence of estimations $\rho_1, \ldots, \rho_n$, by exhaustive sampling, so that $\vec{x}^*$ is a feasible solution to the resulting (IP) (see Section 3.2 for the details).
2. We formulate (IP) and find an optimal fractional solution $\vec{y}^*$ to (LP).
3. We obtain an integral solution $\vec{z}$ by applying randomized rounding to $\vec{y}^*$ (and the method of conditional probabilities, as in [26, 25]).

To see that this procedure indeed provides a good approximation to $p(\vec{x}^*)$, we observe that:

$$p(\vec{z}) \approx \sum_{j \in V} z_j(\deg(j) - \rho_j) \approx \sum_{j \in V} y_j^*(\deg(j) - \rho_j) \geq \sum_{j \in V} x_j^*(\deg(j) - \rho_j) \approx p(\vec{x}^*)\,. \qquad (2)$$

The first approximation holds because $\vec{z}$ is an (almost) feasible solution to (IP) (see Lemma 4), the second approximation holds because the objective value of $\vec{z}$ is a good approximation to the objective value of $\vec{y}^*$, due to randomized rounding, the inequality holds because $\vec{x}^*$ is a feasible solution to (LP) and the final approximation holds because $\vec{x}^*$ is a feasible solution to (IP).

In Sections 3.3 and 3.4, we make the notion of approximation precise so that $p(\vec{z}) \geq (1-\varepsilon)p(\vec{x}^*)$. As for the running time, it is dominated by the time required for the exhaustive-sampling step. Since we do not know $\vec{x}^*$, we need to run the steps (2) and (3) above for every sequence of estimations produced by exhaustive sampling. So, the outcome of the approximation scheme is the best of the integral solutions $\vec{z}$ produced in step (3) over all executions of the algorithm. In Section 3.2, we show that a sample of size $O(n \ln n/\Delta)$ suffices for the computation of estimations $\rho_j$ so that $\vec{x}^*$ is a feasible solution to (IP) with high probability. If $G$ is $\delta$-almost sparse, the sample size is sublinear in $n$ and the running time is subexponential in $n$.

## 3.2 Obtaining Estimations $\rho_j$ by Exhaustive Sampling

To obtain good estimations $\rho_j$ of the values $p_j(\vec{x}^*) = \sum_{i \in N(j)} x_i^*$, i.e., of the number of $j$'s neighbors in $S_1$ in the optimal cut, we take a random sample $R \subseteq V$ of size $\Theta(n \ln n/\Delta)$ and try exhaustively all possible assignments of the vertices in $R$ to $S_0$ and $S_1$. If $\Delta = \Omega(n^\delta)$, we have $2^{O(n \ln n/\Delta)} = 2^{O(n^{1-\delta} \ln n)}$ different assignments. For each assignment, described by a 0/1 vector $\vec{x}$ restricted to $R$, we compute an estimation $\rho_j = (n/|R|) \sum_{i \in N(j) \cap R} x_i$, for each vertex $j \in V$, and run the steps (2) and (3) of the algorithm above. Since we try all possible assignments, one of them agrees with $\vec{x}^*$ on all vertices of $R$. So, for this assignment, the estimations computed are $\rho_j = (n/|R|) \sum_{i \in N(j) \cap R} x_i^*$. The following shows that for these estimations, we have that $p_j(\vec{x}^*) \approx \rho_j$ with high probability.

▶ **Lemma 2.** *Let $\vec{x}$ be any binary vector. For all $\alpha_1, \alpha_2 > 0$, we let $\gamma = \Theta(1/(\alpha_1^2 \alpha_2))$ and let $R$ be a multiset of $r = \gamma n \ln n/\Delta$ vertices chosen uniformly at random with replacement from $V$. For any vertex $j$, if $\rho_j = (n/r) \sum_{i \in N(j) \cap R} x_i$ and $\hat{\rho}_j = \sum_{i \in N(j)} x_i$, with probability at least $1 - 2/n^3$,*

$$(1-\alpha_1)\hat{\rho}_j - (1-\alpha_1)\alpha_2\Delta \leq \rho_j \leq (1+\alpha_1)\hat{\rho}_j + (1+\alpha_1)\alpha_2\Delta\,. \qquad (3)$$

We note that $\rho_j \geq 0$ and always assume that $\rho_j \leq \deg(j)$, since if $\rho_j$ satisfies (3), $\min\{\rho_j, \deg(j)\}$ also satisfies (3). For all $\epsilon_1, \epsilon_2 > 0$, setting $\alpha_1 = \frac{\epsilon_1}{1+\epsilon_1}$ and $\alpha_2 = \epsilon_2$ in Lemma 2, and taking the union bound over all vertices, we obtain that for $\gamma = \Theta(1/(\epsilon_1^2 \epsilon_2))$, with probability at least $1 - 2/n^2$, the following holds for all vertices $j \in V$:

$$(1-\epsilon_1)\rho_j - \epsilon_2\Delta \leq \hat{\rho}_j \leq (1+\epsilon_1)\rho_j + \epsilon_2\Delta\,. \qquad (4)$$

Therefore, with probability at least $1 - 2/n^2$, the optimal cut $\vec{x}^*$ is a feasible solution to (IP) with the estimations $\rho_j$ obtained by restricting $\vec{x}^*$ to the vertices in $R$.

## 3.3 The Cut Value of Feasible Solutions

We next show that the objective value of any feasible solution $\vec{y}$ to (LP) is close to $p(\vec{y})$. Therefore, assuming that $\vec{x}^*$ is feasible, any good approximation to (IP) is a good approximation to the optimal cut.

▶ **Lemma 3.** *Let $\rho_1, \ldots, \rho_n$ be non-negative numbers and $\vec{y}$ be any feasible solution to (LP). Then,*

$$p(\vec{y}) \in \sum_{j \in V} y_j(\deg(j) - \rho_j) \pm 2(\epsilon_1 + \epsilon_2)m\,. \tag{5}$$

## 3.4 Randomized Rounding of the Fractional Optimum

As a last step, we show how to round the fractional optimum $\vec{y}^* = (y_1^*, \ldots, y_n^*)$ of (LP) to an integral solution $\vec{z} = (z_1, \ldots, z_n)$ that almost satisfies the constraints of (IP).

To this end, we use randomized rounding, as in [26]. In particular, we set independently each $z_j$ to 1, with probability $y_j^*$, and to 0, with probability $1 - y_j^*$. By Chernoff bounds [1], we obtain that with probability at least $1 - 2/n^8$, for each vertex $j$,

$$(1-\epsilon_1)\rho_j - \epsilon_2\Delta - 2\sqrt{\deg(j)\ln(n)} \leq \sum_{i \in N(j)} z_i \leq (1+\epsilon_1)\rho_j + \epsilon_2\Delta + 2\sqrt{\deg(j)\ln(n)}\,. \tag{6}$$

Specifically, the inequality above follows from the Chernoff bound in footnote 1, with $k = \deg(j)$ and $t = 2\sqrt{\deg(j)\ln(n)}$, since $\mathbb{E}[\sum_{i \in N(j)} z_j] = \sum_{i \in N(j)} y_j^* \in (1 \pm \epsilon_1)\rho_j \pm \epsilon_2\Delta$. By the union bound, (6) is satisfied with probability at least $1 - 2/n^7$ for all vertices $j$.

By linearity of expectation, $\mathbb{E}[\sum_{j \in V} z_j(\deg(j) - \rho_j)] = \sum_{j \in V} y_j^*(\deg(j) - \rho_j)$. Moreover, since the probability that $\vec{z}$ does not satisfy (6) for some vertex $j$ is at most $2/n^7$ and since the objective value of (IP) is at most $n^2$, the expected value of a rounded solution $\vec{z}$ that satisfies (6) for all vertices $j$ is least $\sum_{j \in V} y_j^*(\deg(j) - \rho_j) - 1$ (assuming that $n \geq 2$). Using the method of conditional expectations, as in [25], we can find in (deterministic) polynomial time an integral solution $\vec{z}$ that satisfies (6) for all vertices $j$ and has $\sum_{j \in V} z_j(\deg(j) - \rho_j) \geq \sum_{j \in V} y_j^*(\deg(j) - \rho_j) - 1$. Next, we sometimes abuse the notation and refer to such an integral solution $\vec{z}$ (computed deterministically) as the integral solution obtained from $\vec{y}^*$ by randomized rounding.

The following is similar to Lemma 3 and shows that the objective value $p(\vec{z})$ of the rounded solution $\vec{z}$ is close to the optimal value of (LP).

▶ **Lemma 4.** *Let $\vec{y}^*$ be the optimal solution of (LP) and let $\vec{z}$ be the integral solution obtained from $\vec{y}^*$ by randomized rounding (and the method of conditional expectations). Then,*

$$p(\vec{z}) \in \sum_{j \in V} y_j^*(\deg(j) - \rho_j) \pm 3(\epsilon_1 + \epsilon_2)m\,. \tag{7}$$

## 3.5 Putting Everything Together

Therefore, for any $\varepsilon > 0$, if $G$ is $\delta$-almost sparse and $\Delta = n^\delta$, the algorithm described in Section 3.1, with sample size $\Theta(n \ln n/(\varepsilon^3\Delta))$, computes estimations $\rho_j$ such that the

[1] We use the following standard Chernoff bound (see e.g., [18, Theorem 1.1]): Let $Y_1, \ldots, Y_k$ independent random variables in $[0,1]$ and let $Y = \sum_{j=1}^k Y_j$. Then for all $t > 0$, $\mathbb{P}[|Y - \mathbb{E}[Y]| > t] \leq 2\exp(-2t^2/k)$.

optimal cut $\vec{x}^*$ is a feasible solution to (IP) whp. Hence, by the analysis above, the algorithm approximates the value of the optimal cut $p(\vec{x}^*)$ within an additive term of $O(\varepsilon m)$. Specifically, setting $\epsilon_1 = \epsilon_2 = \varepsilon/16$, the value of the cut $\vec{z}$ produced by the algorithm satisfies the following with probability at least $1 - 2/n^2$:

$$p(\vec{z}) \geq \sum_{j \in V} y_j^*(\deg(j) - \rho_j) - \frac{3\varepsilon m}{8} \geq \sum_{j \in V} x_j^*(\deg(j) - \rho_j) - \frac{3\varepsilon m}{8} \geq p(\vec{x}^*) - \frac{\varepsilon m}{2} \geq (1-\varepsilon)p(\vec{x}^*)\,.$$

The first inequality follows from Lemma 4, the second inequality holds because $\vec{y}^*$ is the optimal solution to (LP) and $\vec{x}^*$ is feasible for (LP), the third inequality follows from Lemma 3 and the fourth inequality holds because the optimal cut has at least $m/2$ edges.

▶ **Theorem 5.** *Let $G(V, E)$ be a $\delta$-almost sparse graph with $n$ vertices. Then, for any $\varepsilon > 0$, we can compute, in time $2^{O(n^{1-\delta} \ln n/\varepsilon^3)}$ and with probability at least $1 - 2/n^2$, a cut $\vec{z}$ of $G$ with value $p(\vec{z}) \geq (1-\varepsilon)p(\vec{x}^*)$, where $\vec{x}^*$ is the optimal cut.*

## 4 Approximate Maximization of Smooth Polynomials

Generalizing the ideas applied to Max-Cut, we arrive at the main algorithmic result of the paper: an algorithm to approximately optimize $\beta$-smooth $\delta$-bounded polynomials $p(\vec{x})$ of degree $d$ over all binary vectors $\vec{x} \in \{0,1\}^n$. The intuition and the main ideas are quite similar to those in Section 3, but the details are significantly more involved because we are forced to recursively decompose degree $d$ polynomials to eventually obtain a linear program.

▶ **Theorem 6.** *Let $p(\vec{x})$ be an $n$-variate degree-$d$ $\beta$-smooth $\delta$-bounded polynomial. Then, for any $\varepsilon > 0$, we can compute, in time $2^{O(d^7\beta^3 n^{1-\delta} \ln n/\varepsilon^3)}$ and with probability at least $1 - 8/n^2$, a binary vector $\vec{z}$ so that $p(\vec{z}) \geq p(\vec{x}^*) - \varepsilon n^{d-1+\delta}$, where $\vec{x}^*$ is the maximizer of $p(\vec{x})$.*

**Max-$k$-CSP.** Using Theorem 6 it is a straightforward observation that for any Max-$k$-CSP problem (for constant $k$) we can obtain an algorithm which, given a Max-$k$-CSP instance with $\Omega(n^{k-1+\delta})$ constraints for some $\delta > 0$, for any $\varepsilon > 0$ returns an assignment that satisfies $(1-\varepsilon)$OPT constraints in time $2^{O(n^{1-\delta} \ln n/\varepsilon^3)}$. This follows from Theorem 6 using two observations: first, the standard arithmetization of Max-$k$-CSP described in Section 2 produces a degree-$k$ $\beta$-smooth $\delta$-bounded polynomial for $\beta$ depending only on $k$. Second, the optimal solution of such an instance satisfies at least $\Omega(n^{k-1+\delta})$ constraints, therefore the additive error given in Theorem 6 is $O(\varepsilon\text{OPT})$. This algorithm for Max-$k$-CSP contains as special cases algorithm for various standard problems such as Max-Cut, Max-DICUT and Max-$k$-SAT.

## 5 Approximating the $k$-Densest Subgraph in Almost Sparse Graphs

In this section, we present an extension of the algorithms we have presented which can be used to approximate $k$-Densest Subgraph in $\delta$-almost sparse graphs. This is a problem also handled in [4], but only for the case where $k = \Omega(n)$. Smaller values of $k$ cannot be handled by the scheme of [4] for dense graphs because when $k = o(n)$ the optimal solution has objective value much smaller than the additive error of $\varepsilon n^2$ inherent in their scheme.

Here we obtain a sub-exponential time approximation scheme that works on graphs with $\Omega(n^{1+\delta})$ edges *for all* $k$ by judiciously combining two approaches: when $k$ is relatively large, we use a sampling approach similar to Max-Cut; when $k$ is small, we can resort to the naïve algorithm that tries all $\binom{n}{k}$ possible solutions. We select (with some foresight) the

threshold between the two algorithms to be $k = \Omega(n^{1-\delta/3})$, so that in the end we obtain an approximation scheme with running time of $2^{O(n^{1-\delta/3}\ln n)}$, that is, slightly slower than the approximation scheme for Max-Cut. It is clear that the brute-force algorithm achieves this running time for $k = O(n^{1-\delta/3})$, so in the remainder we focus on the case of large $k$.

The $k$-Densest Subgraph problem in a graph $G(V, E)$ is equivalent to maximizing, over all vectors $\vec{x} \in \{0,1\}^n$, the $n$-variate degree-2 1-smooth polynomial $p(\vec{x}) = \sum_{\{i,j\}\in E} x_i x_j$, under the linear constraint $\sum_{j\in V} x_j = k$. Setting a variable $x_i$ to 1 indicates that the vertex $i$ is included in the set $C$ that induces a dense subgraph $G[C]$ of $k$ vertices. We assume that $G$ is $\delta$-almost sparse i.e. $m = \Omega(n^{1+\delta})$ edges. As usual, $\vec{x}$ denotes the optimal solution.

The algorithm follows the same general approach and the same basic steps as the algorithm for Max-Cut in Section 3. In the following, we highlight only the differences.

**Obtaining Estimations by Exhaustive Sampling.** We first observe that if $G$ is $\delta$-almost sparse and $k = \Omega(n^{1-\delta/3})$, a random subset of $k$ vertices contains $\Omega(n^{1+\delta/3})$ edges in expectation. We thus assume that the optimal solution induces at least $\Omega(n^{1+\delta/3})$ edges.

Working as in Section 3.2, we use exhaustive sampling and obtain for each vertex $j \in V$, an estimation $\rho_j$ of $j$'s neighbors in the optimal dense subgraph, i.e., $\rho_j$ is an estimation of $\hat{\rho}_j = \sum_{i\in N(j)} x^*_i$. For the analysis, we apply Lemma 2 with $n^{\delta/3}$, instead of $\Delta$, or in other words, we use a sample of size $\Theta(n^{1-\delta/3}\ln n)$. The reason is that we can only tolerate an additive error of $\varepsilon n^{1+\delta/3}$, by the lower bound on the optimal solution observed in the previous paragraph. Then, the running time due to exhaustive sampling is $2^{O(n^{1-\delta/3}\ln n)}$.

By Lemma 2 and the discussion following it in Section 3.2, we obtain that for all $\epsilon_1, \epsilon_2 > 0$, if we use a sample of the size $\Theta(n^{1-\delta/3}\ln n/(\epsilon_1^2\epsilon_2))$, with probability at least $1 - 2/n^2$, the following holds for all estimations $\rho_j$ and all vertices $j \in V$:

$$(1-\epsilon_1)\rho_j - \epsilon_2 n^{\delta/3} \le \hat{\rho}_j \le (1+\epsilon_1)\rho_j + \epsilon_2 n^{\delta/3} \tag{8}$$

**Linearizing the Polynomial.** Applying Lemma 1, we can write the polynomial $p(\vec{x})$ as $p(\vec{x}) = \sum_{j\in V} x_j p_j(\vec{x})$, where $p_j(\vec{x}) = \sum_{i\in N(j)} x_i$ is a degree-1 1-smooth polynomial that indicates how many neighbors of vertex $j$ are in $C$ in the solution corresponding to $\vec{x}$. Then, using the estimations $\rho_j$ of $\sum_{i\in N(j)} x^*_i$, obtained by exhaustive sampling, we have that approximate maximization of $p(\vec{x})$ can be reduced to the solution of the following ILP:

$$\begin{aligned} &\max \sum_{j\in V} y_j\rho_j && \text{(IP}'\text{)}\\ \text{s.t.}\quad &(1-\epsilon_1)\rho_j - \epsilon_2 n^{\delta/3} \le \sum_{i\in N(j)} y_i \le (1+\epsilon_1)\rho_j + \epsilon_2 n^{\delta/3} \quad \forall j \in V\\ &\sum_{i\in V} y_i = k \end{aligned}$$

By (8), if the sample size is $|R| = \Theta(n^{1-\delta/3}\ln n/(\epsilon_1^2\epsilon_2))$, with probability at least $1-2/n^2$, the densest subgraph $\vec{x}^*$ is a feasible solution to (IP$'$) with the estimations $\rho_j$ obtained by restricting $\vec{x}^*$ to the vertices in $R$. In the following, we let (LP$'$) denote the Linear Programming relaxation of (IP$'$), where each $y_j \in [0,1]$.

**The Number of Edges in Feasible Solutions.** We next show that the objective value of any feasible solution $\vec{y}$ to (LP$'$) is close to $p(\vec{y})$. Therefore, assuming that $\vec{x}^*$ is feasible, any good approximation to (IP$'$) is a good approximation to the densest subgraph.

▶ **Lemma 7.** *Let $\rho_1, \ldots, \rho_n$ be non-negative numbers and $\vec{y}$ be any feasible solution to (LP′). Then,*

$$p(\vec{y}) \in (1 \pm \epsilon_1) \sum_{j \in V} y_j \rho_j \pm \epsilon_2 n^{1+\delta/3} . \tag{9}$$

**Randomized Rounding of the Fractional Optimum.** As a last step, we show how to round the fractional optimum $\vec{y}^* = (y_1^*, \ldots, y_n^*)$ of (LP′) to an integral solution $\vec{z} = (z_1, \ldots, z_n)$ that almost satisfies the constraints of (IP′). We use randomized rounding, as for Max-CUT.

▶ **Lemma 8.** *Let $\vec{y}^*$ be the optimal solution of (LP′) and let $\vec{z}$ be the integral solution obtained from $\vec{y}^*$ by randomized rounding (and the method of conditional expectations). Then,*

$$p(\vec{z}) \in (1 \pm \epsilon_1)^2 \sum_{j \in V} y_j^* \rho_j \pm 3\epsilon_2 n^{1+\delta/3} . \tag{10}$$

We thus arrive to the main theorem of this section.

▶ **Theorem 9.** *Let $G(V, E)$ be a $\delta$-almost sparse graph with $n$ vertices. Then, for any integer $k \geq 1$ and for any $\varepsilon > 0$, we can compute, in time $2^{O(n^{1-\delta/3} \ln n/\varepsilon^3)}$ and with probability at least $1 - 2/n^2$, an induced subgraph $\vec{z}$ of $G$ with $k$ vertices whose number of edges satisfies $p(\vec{z}) \geq (1-\varepsilon)p(\vec{x}^*)$, where $\vec{x}^*$ is the number of edges in the $k$-Densest Subgraph of $G$.*

## 6 Lower Bounds

We now give some lower bound arguments showing that the schemes we have presented are, in some senses, likely to be almost optimal. Our complexity assumption will be the ETH, which states that no algorithm can solve instances of 3-SAT of size $n$ in time $2^{o(n)}$.

There are two natural ways in which one may hope to improve or extend the algorithms we have presented so far: relaxing the density requirement or decreasing the running time. First, recall that the algorithm we have given for Max-$k$-CSP works in the density range between $n^k$ and $n^{k-1}$. Here, we give a reduction establishing that it's unlikely that this can be improved. Our starting point is the following (known) inapproximability result.

▶ **Theorem 10.** *There exist $c, s \in (0,1)$ with $c > s$ such that for all $\epsilon > 0$ we have: if there exists an algorithm which, given an $n$-vertex 5-regular instance of Max-CUT, can distinguish between the case where a solution cuts at least a $c$ fraction of the edges and the case where all solutions cut at most an $s$ fraction of the edges in time $2^{n^{1-\epsilon}}$ then the ETH fails.*

▶ **Theorem 11.** *There exists $r > 1$ such that for all $\epsilon > 0$ and all (fixed) integers $k \geq 3$ we have the following: if there exists an algorithm which $r$-approximates Max-$k$-SAT on instances with $\Omega(n^{k-1})$ clauses in time $2^{n^{1-\epsilon}}$ then the ETH fails.*

**Proof.** We reduce a Max-CUT instance from Theorem 10 to Max-2-SAT: the set of variables is the set of vertices; for each edge $(u, v)$ we include the clauses $(u \vee v)$ and $(\neg u \vee \neg v)$. The new instance has $n$ variables and $5n$ clauses and there exist constants $c, s$ such that either some assignment satisfies $5cn$ clauses or all assignments satisfy at most $5sn$ of them.

Fix $k$ and add to the instance $(k-2)n$ new variables $x_{(i,j)}$, $i \in \{1, \ldots, k-2\}$, $j \in \{1, \ldots, n\}$. We perform the following transformation: for each clause $(l_1 \vee l_2)$ and for each tuple $(i_1, i_2, \ldots, i_{k-2}) \in \{1, \ldots, n\}^{k-2}$ we construct $2^{k-2}$ new clauses of size $k$. The first two literals of these clauses are $l_1, l_2$. The rest consist of the variables $x_{(1,i_1)}, x_{(2,i_2)}, \ldots, x_{(k,i_{k-2})}$, but in each clause a different set of variables is negated. In other words, to construct a clause

of the new instance we select a clause of the original instance, one variable from each of the groups of $n$ new variables, and a subset of these variables to be negated.

First, observe that the new instance has $5n^{k-1}2^k$ clauses and $(k-1)n$ variables, which satisfies the density conditions. Consider an assignment of the original formula. Any satisfied clause has now been replaced by $n^{k-2}2^k$ satisfied clauses, while for an unsatisfied clause any assignment to the new variables satisfies exactly $n^{k-2}(2^k-1)$ clauses. Thus, for fixed $k$, there exist constants $s', c'$ such that either a $c'$ fraction of the clauses of the new instance is satisfiable or at most a $s'$ fraction is. If we had an approximation algorithm with ratio better than $c'/s'$ running in time $2^{N^{1-\epsilon}}$, where $N$ is the number of variables of the new instance, we could use it to decide the original instance in time that would disprove the ETH. ◀

A second possible avenue for improvement may be to consider potential speedups of our algorithms. We give an almost tight answer to such questions via the following theorem.

▶ **Theorem 12.** *There exists $r > 1$ such that for all $\epsilon > 0$ we have the following: if there exists an algorithm which, for some $\Delta = o(n)$, $r$-approximates* Max-*CUT on $n$-vertex $\Delta$-regular graphs in time $2^{(n/\Delta)^{1-\epsilon}}$ then the ETH fails.*

**Proof.** Without loss of generality we prove the theorem for the case when the degree is a multiple of 10. Consider an instance $G(V, E)$ of Max-CUT as given by Theorem 10. Let $n = |V|$ and suppose that the desired degree is $d = 10\Delta$, where $\Delta$ is a function of $n$. We construct a graph $G'$ as follows: for each vertex $u \in V$ we introduce $\Delta$ new vertices $u_1, \ldots, u_\Delta$ as well as $5\Delta$ "consistency" vertices $c_1^u, \ldots, c_{5\Delta}^u$. For every edge $(u, v) \in E$ we add all edges $(u_i, v_j)$ for $i, j \in \{1, \ldots, \Delta\}$. Also, for every $u \in V$ we add all edges $(u_i, c_j^u)$, for $i \in \{1, \ldots, \Delta\}$ and $j \in \{1, \ldots, 5\Delta\}$. This completes the construction.

The graph we have constructed is $10\Delta$-regular and is made up of $6\Delta n$ vertices. Consider an optimal cut and observe that, for a given $u \in V$ all the vertices $c_i^u$ can be assumed to be on the same side of the cut, since they all have the same neighbors. Furthermore, for a given $u \in V$, all vertices $u_i$ can be assumed to be on the same side of the cut, namely on the side opposite that of $c_i^u$, since the vertices $c_i^u$ are a majority of the neighborhood of each $u_i$. With this observation it is easy to construct a one-to-one correspondence between cuts in $G$ and locally optimal cuts in $G'$.

Consider a cut that cuts $c|E|$ edges of $G$. If we set all $u_i$ of $G'$ on the same side as $u$ is in $G$ we cut $c|E|\Delta^2$ edges of the form $(u_i, v_j)$. Furthermore, by placing the $c_i^u$ on the opposite side of $u_i$ we cut $5\Delta^2|V|$ edges. Thus the max cut of $G'$ is at least $c|E|\Delta^2 + 5\Delta^2|V|$. Using the observations on locally optimal cuts of $G'$ we can conclude that if $G'$ has a cut with $s|E|\Delta^2 + 5\Delta^2|V|$ edges, then $G$ has a cut with $s|E|$ edges. Having $2|E| = 5|V|$ (since $G$ is 5-regular) we get a constant ratio $r$ between the size of the cut of $G'$ in the two cases.

Suppose now that we have an approximation algorithm with ratio better than $r$ which, given an $N$-vertex $d$-regular graph runs in time $2^{(N/d)^{1-\epsilon}}$. Giving our constructed instance as input to this algorithm would allow to decide the original instance in time $2^{n^{1-\epsilon}}$. ◀

## References


1 Nir Ailon and Noga Alon. Hardness of fully dense problems. *Inf. Comput.*, 205(8):1117–1129, 2007. doi:10.1016/j.ic.2007.02.006.

2 Noga Alon, Wenceslas Fernandez de la Vega, Ravi Kannan, and Marek Karpinski. Random sampling and approximation of max-csps. *J. Comput. Syst. Sci.*, 67(2):212–243, 2003. doi:10.1016/S0022-0000(03)00008-4.

**3** A. Arora, A. Frieze, and H. Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Mathematical Programming, Series A*, 92:1–36, 2002.

**4** A. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. *Journal of Computer and System Sciences*, 58:193–210, 1999.

**5** Cristina Bazgan, Wenceslas Fernandez de la Vega, and Marek Karpinski. Polynomial time approximation schemes for dense instances of minimum constraint satisfaction. *Random Struct. Algorithms*, 23(1):73–91, 2003. doi:10.1002/rsa.10072.

**6** Nicolas Bourgeois, Federico Della Croce, Bruno Escoffier, and Vangelis Th. Paschos. Fast algorithms for min independent dominating set. *Discrete Applied Mathematics*, 161(4-5):558–572, 2013. doi:10.1016/j.dam.2012.01.003.

**7** Nicolas Bourgeois, Bruno Escoffier, and Vangelis Th. Paschos. Approximation of max independent set, min vertex cover and related problems by moderately exponential algorithms. *Discrete Applied Mathematics*, 159(17):1954–1970, 2011. doi:10.1016/j.dam.2011.07.009.

**8** Jean Cardinal, Marek Karpinski, Richard Schmied, and Claus Viehmann. Approximating subdense instances of covering problems. *Electronic Notes in Discrete Mathematics*, 37:297–302, 2011. doi:10.1016/j.endm.2011.05.051.

**9** Jean Cardinal, Marek Karpinski, Richard Schmied, and Claus Viehmann. Approximating vertex cover in dense hypergraphs. *J. Discrete Algorithms*, 13:67–77, 2012. doi:10.1016/j.jda.2012.01.003.

**10** Parinya Chalermsook, Bundit Laekhanukit, and Danupon Nanongkai. Independent set, induced matching, and pricing: Connections and tight (subexponential time) approximation hardnesses. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 370–379. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.47.

**11** Marek Cygan, Lukasz Kowalik, and Mateusz Wykurz. Exponential-time approximation of weighted set cover. *Inf. Process. Lett.*, 109(16):957–961, 2009. doi:10.1016/j.ipl.2009.05.003.

**12** Marek Cygan and Marcin Pilipczuk. Exact and approximate bandwidth. *Theor. Comput. Sci.*, 411(40-42):3701–3713, 2010. doi:10.1016/j.tcs.2010.06.018.

**13** Marek Cygan, Marcin Pilipczuk, and Jakub Onufry Wojtaszczyk. Capacitated domination faster than $O(2^n)$. *Inf. Process. Lett.*, 111(23-24):1099–1103, 2011. doi:10.1016/j.ipl.2011.09.004.

**14** Wenceslas Fernandez de la Vega. MAX-CUT has a randomized approximation scheme in dense graphs. *Random Struct. Algorithms*, 8(3):187–198, 1996. doi:10.1002/(SICI)1098-2418(199605)8:3<187::AID-RSA3>3.0.CO;2-U.

**15** Wenceslas Fernandez de la Vega and Marek Karpinski. On the approximation hardness of dense TSP and other path problems. *Inf. Process. Lett.*, 70(2):53–55, 1999. doi:10.1016/S0020-0190(99)00048-4.

**16** Wenceslas Fernandez de la Vega and Marek Karpinski. Polynomial time approximation of dense weighted instances of MAX-CUT. *Random Struct. Algorithms*, 16(4):314–332, 2000. doi:10.1002/1098-2418(200007)16:4<314::AID-RSA2>3.0.CO;2-E.

**17** Wenceslas Fernandez de la Vega and Marek Karpinski. Approximation complexity of nondense instances of MAX-CUT. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(101), 2006. URL: http://eccc.hpi-web.de/eccc-reports/2006/TR06-101/index.html.

**18** D.P. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

**19** Martin E. Dyer, Alan M. Frieze, and Mark Jerrum. Approximately counting hamilton paths and cycles in dense graphs. *SIAM J. Comput.*, 27(5):1262–1272, 1998. doi:10.1137/S009753979426112X.

**20** Alan M. Frieze and Ravi Kannan. The regularity lemma and approximation schemes for dense problems. In *37th Annual Symposium on Foundations of Computer Science, FOCS'96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 12–20. IEEE Computer Society, 1996. doi:10.1109/SFCS.1996.548459.

**21** Johan Hastad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

**22** Tomokazu Imamura and Kazuo Iwama. Approximating vertex cover on dense graphs. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005, Vancouver, British Columbia, Canada, January 23-25, 2005*, pages 582–589. SIAM, 2005. URL: http://dl.acm.org/citation.cfm?id=1070432.1070513.

**23** Sanjeev Khanna, Madhu Sudan, and David P. Williamson. A complete classification of the approximability of maximization problems derived from boolean constraint satisfaction. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 11–20. ACM, 1997. doi:10.1145/258533.258538.

**24** Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5), 2010. doi:10.1145/1754399.1754402.

**25** P. Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130–143, 1988.

**26** P. Raghavan and C.D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–474, 1987.

**27** Thomas J. Schaefer. The complexity of satisfiability problems. In Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho, editors, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 216–226. ACM, 1978. doi:10.1145/800133.804350.

**28** Luca Trevisan. Inapproximability of combinatorial optimization problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 2004. URL: http://eccc.hpi-web.de/eccc-reports/2004/TR04-065/index.html.