ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 332 (2005) 123–139
Theoretical Computer Science
www.elsevier.com/locate/tcs

The approximability of non-Boolean satisfiability problems and restricted integer programming ☆

Maria Serna[a,*,1], Luca Trevisan[b], Fatos Xhafa[a,1]

[a] Department of LSI, UPC, Campus Nord - C6, Jordi Girona Salgado, 1-3, 08034-Barcelona, Spain
[b] University of California at Berkeley Computer Science Division 615 Soda Hall, Berkeley, CA 94720-1776, USA

Received 14 July 2003; received in revised form 15 July 2004; accepted 6 October 2004
Communicated by O. Watanabe

Abstract

In this paper we present improved approximation algorithms for two classes of maximization problems defined in Barland et al. (J. Comput. System Sci. 57(2) (1998) 144). Our factors of approximation substantially improve the previous known results and are close to the best possible. On the other hand, we show that the approximation results in the framework of Barland et al. hold also in the parallel setting, and thus we have a new common framework for both computational settings. We prove almost tight non-approximability results, thus solving a main open question of Barland et al.

We obtain the results through the constraint satisfaction problem over multi-valued domains, for which we develop approximation algorithms and show non-approximability results. Our parallel approximation algorithms are based on linear programming and random rounding; they are better than previously known sequential algorithms. The non-approximability results are based on new recent progress in the fields of probabilistically checkable proofs and multi-prover one-round proof systems.

Keywords: Parallel approximation algorithms; Non-approximability; Positive linear programming; Randomized rounding; Non-Boolean constraint satisfaction; Maximum capacity representatives

☆ Preliminary version appeared in Proceedings of 15th Annual Symposium on Theoretical Aspects of Computer Science, STACS'98.

* Corresponding author.

E-mail addresses: mjserna@lsi.upc.es (M. Serna), luca@eecs.berkeley.edu (L. Trevisan), fatos@lsi.upc.es (F. Xhafa).

[1] This research was partially supported by the ALCOM FT Research Project No. IST-1999-14186 and CICYT Project TRACER TIC2002-04498-C05-03.

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2004.10.014

1. Introduction

Expressing combinatorial optimization problems as integer linear programs (ILP) has several applications. In particular, several approximation algorithms start from the linear programming relaxation of the ILP formulation, and then use randomized rounding [25,11], primal-dual methods [12], or more sophisticated methods [21,10].

An interesting new *structural* use of Integer Linear Programming has been taken in a recent paper of Barland et al. [2], where *syntactic classes* of maximization problems are introduced. A problem belongs to one such class if it can be expressed by an ILP with a certain restricted format. The approximability properties of the problem in a class are then implied by the approximability of the respective prototypical ILP. The main goal of [2] was to overcome some limitations of the standard way of defining syntactic classes, namely the approach of logical definability [24,23,18,19]. The latter approach, indeed, fails to explain why problems with similar logical definability, such as MAX k–DIMENSIONAL MATCHING and MAX CLIQUE have very different approximability properties. Furthermore, using ILP, classes are defined in terms of a single parameter that determines the hardness of the problems. This parameter is either the maximum number of occurrences of any variable or the maximum size of the domain of the variables. The latter kind of restriction gives rise to a family of classes that Barland et al. call Max FSBLIP (for-maximum feasible subsystem of bounded layered integer program). Essentially, these classes consist of linear integer programs with *syntactic* restrictions on the range of the variables, the number of occurrences of a variable (e.g. the variables of the objective function can appear a bounded number of times in the program) and the *dominance* condition on the constraints—a syntactic criterion that try to capture the arithmetic nature of a constraint. Letting the variables to take values in a constant, logarithmic, or polynomial range allowed Barland et al. to capture syntactic maximization classes that are constant-approximable, polylog-approximable and poly-approximable, respectively. An interesting question is whether these three classes form a proper hierarchy. Barland et al. did not completely resolve this point and left improved non-approximability results as an open question.

In this paper our interest is twofold. In one hand, we use the integer programming as a framework for parallel approximability, aiming to obtain improved parallel approximation results. It is known that all the problems contained in logically defined syntactic classes that are constant-factor approximable, are also constant-factor approximable [2] in NC. This feature of logically defined syntactic classes is desirable for at least two reasons: it reduces the study of sequential and parallel approximability to the same framework, and is in accordance with the fact that almost all the constant-factor approximation algorithms that are known also admit a parallel version with a comparable approximation ratio (see, e.g. [29]). The issue of parallel approximability is not raised in the paper of Barland et al. Our parallel results state that in the new framework of integer programming the sequential results hold as well as in the parallel setting; thus, again we have a common framework for both computational settings. Having this outcome, the second question that we consider is

[2] In this paper we use NC to denote the class of problems that can be solved by an algorithm that runs in poly-logarithmic time on a parallel shared-memory machine with a polynomial number of processors. See, e.g. [8].

what are the limits of parallel, as well as sequential, approximability for these problems. We show that our approximation factors are nearly the best possible by providing some new non-approximability results (the non-approximability results will also hold for sequential algorithms.) In both cases, our main results will be expressed in terms of the multi-valued constraint satisfaction problem, and then translated, by means of reductions, in terms of the model of Barland et al.

Our results and comparison to previous results. In the following, we state our results and we discuss their relation with previously known ones.

In this paper, a crucial role is played by the constraint satisfaction problem over multi-valued domains. In an instance of this problem, we are given a set of constraints of arity at most k over multi-values variables where a constraint is a boolean valued function over $\{0, 1, \ldots, d-1\}^k$ and is given a positive weight. We can think of a k-ary domain-d constraint as a set of k-tuples values (i.e. a relation over $\{0, 1, \ldots, d-1\}^k$) and say that an assignment satisfies the constraint if the corresponding values to the variables of the constraint form a k-tuple belonging to the relation. The goal is to find an assignment to the variables that maximizes the total weight of satisfied constraints. This problem is a common generalization of several known and well-studied problems. To begin with, it is a natural generalization of the boolean constraint satisfaction problem MAX kCSP, introduced by Khanna et al. [16] and then studied in [7,28,17] (in the boolean case, the domain is $\{0, 1\}$, that is, $d = 2$.) It also generalizes multi-prover one-round proof systems and the MAX CAPACITY REPRESENTATIVES problem (introduced by Bellare [3] and further considered by Barland et al.). The version over multi-valued domain has been studied in the restricted case of binary constraints [20] and that of “planar instances” [15]. In this paper we address, for the first time, the approximability of the problem in its full generality. We present a parallel approximation, based on linear programming and random rounding, that achieves an approximation factor $1/d^{k-1}$. The algorithm can be efficiently parallelized and de-randomized. Our major contribution here is the definition and the analysis of an appropriate random rounding scheme. The parallelization mimics a similar proof in [28], but is not entirely straightforward. For the special case of binary constraint ($k = 2$), our approximation guarantee is twice better than the $1/2d$-approximate algorithm of [20].

We also prove several non-approximability results under different complexity assumptions. Such results, follow from recent advances in the fields of probabilistically checkable proofs [13] and of multi-prover one-round proof systems [27,26,1] and from the fact that multi-valued constraint satisfaction problems generalize both models.

In a recent paper, Engebretsen [9] considered the MAX kCSP-G problem -the generalization of the MAX kCSP over a finite abelian group $G-$ and showed that MAX kCSP-G cannot be approximated within $|G|^{k-O(\sqrt{k})-\varepsilon}$, for any constant ε, unless P = NP. This lower bound matches with our upper bound $|G|^{k-1}$ for the problem.

We use reductions from the multi-valued constraint satisfaction problem to derive negative approximation results for the rest of the problems of interest. In terms of the class FSBLIP, our result states that the classes Max FSBLIP(2), Max FSBLIP(log) and Max FSBLIP(poly) form a proper hierarchy (the separation of the two last classes derives from a result of Bellare [3] stating that MAX CAPACITY REPRESENTATIVES which belongs to Max FSBLIP(poly) is not log-approximable; we separate the first two classes by proving that MAX CAPACITY REPRESENTATIVES(log) is not constant-approximable.)

We also consider the class of integer programs Max FMIP (for maximum feasible majority-integer program) for which MAX MAJORITY SAT is a canonical problem. Barland et al. [2] showed that this class contains only constant-approximable problems. For the general Max FMIP problem, we present a slight improvement and simplification over their approximation result. The latter result does not depend on the constraint satisfaction problem. We also prove an almost tight non-approximability result for the problems of this class by reducing from the boolean constraint satisfaction problem.

Organization of the paper. The paper is organized as follows. In Section 2 we give formal definitions of the problems and of the classes of maximization problems we study. Some of the definitions are accompanied by examples so to facilitate the reading. Section 3 contains reductions from the multi-valued constraint satisfaction problem, which enable us to infer approximability and non-approximability results to the rest of problems. The main results of the paper are given in Sections 4 and 5 where we give, respectively, approximability and non-approximability results. We conclude with some remarks in Section 6.

2. Preliminaries

For an integer n, we denote by $[n]$ the set $\{0, \ldots, n-1\}$. A combinatorial optimization problem is characterized by the set of *instances*, by the finite set of *feasible solutions* associated to any instance, and by a *measure* function that associates a non-negative *cost* to any feasible solution of a given instance. We refer, e.g. to [6] for the formal definition of NP Optimization problem.

Definition 1 (MAX CAPACITY REPRESENTATIVES-d). For a function d defined over positive integers, $d : \mathcal{Z}^+ \to \mathcal{Z}^+$, MAX CAPACITY REPRESENTATIVES-$(d(n))$ problem is defined as follows:

Instance: A partition of $\{1, \ldots, n\}$ into sets $S_1, \ldots, S_m$, each, of cardinality at most d; and weights $w_{i,j} \geqslant 0$ for any two elements belonging to different sets of the partition.

Solution: The choice of a representative in any set.

Measure: The sum of the weights $w_{i,j}$ for any i and j that are representatives in different sets of the partition.

Note that a feasible solution to the problem consists of exactly one “representative” element from each set, also called system of representatives, and we want to maximize the edge weight of the m-clique induced by the representatives. This problem was introduced by Bellare [3] who showed that 2P1R (two prover, one round proof systems) reduces to MAX CAPACITY REPRESENTATIVES-d and consequently the problem cannot be approximated in polynomial time within $2^{\Theta(\log^{1/c} n)}$, unless $\mathsf{NP} \subseteq \cup_{d>0}\mathsf{DTIME}(n^{d \log^c n})$. In particular the problem is not log-approximable modulo this assumption.

Definition 2 (*k-ary domain-d constraint*). A k-ary domain-d constraint over variables $x_1, \ldots, x_n$ is a pair $(f, (i_1, \ldots, i_k))$ where $f : [d]^k \to \{0, 1\}$ and $i_j \in \{1, \ldots, n\}$ for $j = 1, \ldots, k$. A constraint $C = (f, (i_1, \ldots, i_k))$ is satisfied by an assignment $\mathbf{a} = a_1, \ldots, a_n$ to $x_1, \ldots, x_n$ if $C(\mathbf{a}) \stackrel{\text{def}}{=} f(a_{i1}, \ldots, a_{ik}) = 1$.

We say that a function $f : [d]^k \to \{0, 1\}$ is *conjunctive* if it can be expressed as a conjunction of equations, i.e. there are values $v_1, \ldots, v_k \in [d]$,

$$f(x_1, \ldots, x_k) = 1 \text{ if and only if } [x_1 = v_1] \wedge \ldots \wedge [x_k = v_k].$$

When this will not cause confusion, we will sometimes blur the important difference between a constraint $(f, (i_1, \ldots, i_k))$ and the function f. For example we say that a constraint $(f, (i_1, \ldots, i_k))$ is conjunctive if function f is, and so on.

Definition 3 (MAX kCSP-d *and* MAX kCONJ-d). For any integer $k \geqslant 1$ and function $d = d(n)$, the MAX kCSP-d is defined as follows:
Instance: A set $\{C_1, \ldots, C_m\}$ of domain-d constraints of arity at most k over $x_1, \ldots, x_n$, and associated non-negative weights $w_1, \ldots, w_m$.
Solution: An assignment $\mathbf{a} = (a_1, \ldots, a_n) \in [d]^n$ to the variables $x_1, \ldots, x_n$.
Measure: The total weight of satisfied constraints.
MAX kCONJ-d is the restriction of MAX kCSP-d to instances where all the constraints are conjunctive.

Some special cases of the problem are as follows. For the case of $k = 2$, we have binary constraints (e.g. MAX 2CONJ-d), and for $d = 2$ we have constraints over boolean variables (e.g. MAX kCONJ-2). The general MAX kCSP-d for $d = 2$ is the standard constraint satisfaction problem, denoted MAX kCSP.

Definition 4 (*Integer linear programming*). The *ILP* is as follows:
Instance: A matrix $A \in \mathcal{Z}^{m \times n}$ and two vectors $\mathbf{c} \in \mathcal{Z}^n$ and $\mathbf{b} \in \mathcal{Z}^m$.
Solution: A vector $\mathbf{x} \in \mathcal{Z}^n$ satisfying $A\mathbf{x} \leqslant \mathbf{b}$.
Measure: $\mathbf{c} \cdot \mathbf{x}$.

Note that in this formulation, the goal is to maximize the measure $\mathbf{c} \cdot \mathbf{x}$. The variables appearing (with non-zero coefficients) in the objective function are called *objective variables* and those appearing only in the linear constraints are *program variables*. The *width* of a constraint is equal to the number of its variables. Moreover, we will assume that program variables take integer values from the interval $[0, d(n))$ for some function d.

Definition 5 (*Constraint dominance*). Given a linear constraint $\mathcal{C}$ of the form $\gamma(1 - t) + \mathbf{a} \cdot \mathbf{q} \geqslant b$, where t is 0/1 variable and $\gamma > 0$, it is said that t dominates the constraint if

$$\gamma \geqslant \left(\sum_{a_j < 0} (d - 1)|a_j| \right) + b.$$

It should be observed that the constraint dominance can be stated as: "t dominates the constraint $\mathcal{C}$ iff for $t = 0$ the constraint is satisfied whatever is the assignment to the rest of variables". Obviously, if an assignment satisfies $\mathbf{a} \cdot \mathbf{q} \geqslant b$, then the constraint is satisfied for any value of t.

Definition 6 (Max FSBLIP($d(n)$) *(Barland et al. [2])*). For a given function $d(n)$, the class Max FSBLIP($d(n)$) contains all the optimization problems Π for which there are positive integer constants l, m, o' (that only depend on Π) such that every instance of Π can be expressed as an ILP with the following structure:

- The program variables can take values in $\{0, 1, \ldots, d(n) - 1\}$.
- Each objective variable t_i takes either 0 or 1 and occurs only in constraints of the form $(1 - t_i) + q_{i,1} + \cdots + q_{i,z} \geqslant 1$ dominated by t_i, where $z \in \mathbf{N}$ can be polynomial in n, and each $q_{i,j}$, $1 \leqslant j \leqslant z$ is a 0/1 program variable associated with the objective variable t_i. These constraints are referred to as objective constraints.
- Each variable $q_{i,j}$ appearing in an objective constraint occurs in at most l other constraints and dominates each of them.
- All constraints that are not objective ones have width m and are dominated by some $q_{i,j}$ associated with some objective variable t_i.
- Each objective variable t_i appears in at most o objective constraints.

For a flavor of how the problems of this class are, let us consider the ILP for Max SAT problem which belongs to the class Max FS-BLIP(2). Given an instance $\mathcal{C}$ of Max SAT consisting of m clauses $C_1, \ldots, C_m$ on n variables $x_1, \ldots, x_n$, we let t_i be a 0/1 variable corresponding to whether the ith clause is false/true; to the ith clause there are associated n variables $q_{i,1}, \ldots, q_{i,n}$. Further, to any variable x_j of the formula, there is associated a 0/1 variable depending whether the variable is assigned to false or true, respectively. For any clause C_i and any variable x_j is introduced a 0/1 constant $p_{i,j}$ assigned to 1 if x_j appears positively in clause C_i and 0 otherwise. Similarly the constants $n_{i,j}$ are defined, that is, $n_{i,j}$ is assigned to 1 if x_j appears negatively in clause C_i and 0 otherwise. The ILP [2] is as follows:

$$
\begin{array}{rl}
\max & t_1 + t_2 + \cdots + t_m \\
\text{s.t.} & \\
& (1 - t_i) + q_{i,1} + q_{i,2} + \cdots + q_{i,n} \geqslant 1, \\
& (1 - q_{i,j}) + p_{i,j} + n_{i,j} \geqslant 1, \\
& (1 - q_{i,j}) + (1 - p_{i,j}) + v_j \geqslant 1, \\
& (1 - q_{i,j}) + (1 - n_{i,j}) + (1 - v_j) \geqslant 1, \\
& t_i, v_j, q_{i,j} \in \{0, 1\}, \\
& 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n.
\end{array}
\tag{1}
$$

Notice that for this program we have $d = 2$, each t_i dominates the constraint (1) and $q_{i,j}$ dominates the rest of constraints. For any i and j, $q_{i,j}$ appears in four constraints in the entire integer program; the constraints which are not objective ones have width 3 and, finally, for any i, t_i appears in only one objective constraint.

The second class is that of Max FMIP (maximum feasible majority IP) for which Max Majority SAT is a canonical problem.

Definition 7 (Max FMIP *(Barland et al. [2])*). An optimization problem Π belongs to the class Max Feasible Majority IP (in short, Max FMIP) if there exist positive constants k, σ and a polynomial p such that for any instance I of Π we can find a set of linear inequalities over the integers

$$A\mathbf{x} \geqslant b,$$
$$\mathbf{x} \in \{-k, -k+1, \ldots, k-1, k\}^n,$$

where $b_j \leqslant \sigma$, the entries of A are integers of absolute value at most $p(n)$, and the optimum of I is precisely the maximum number of inequalities that are simultaneously satisfiable. Variables are allowed to take integer values from the interval $[-k, k]$ or $[-k, k] - \{0\}$.

As an example, let us consider the problem MAX MAJORITY SAT. We recall that an instance of MAX MAJORITY SAT is an instance of SAT with the additional condition that a clause is satisfied if at least half of its literals are true. Now we match the above definition as follows. For any boolean variable, introduce a (numeric) variable taking values in $\{-1, 1\}$. To any clause there is associated an inequality linear restriction. The left-hand side of the inequality is the sum of the variables corresponding to the variables of the clause with coefficients ± 1 depending whether they appear positively or negated in the clause; the right-hand side is 0. For example, to the instance $(x, \neg y, \neg z), (x, y, \neg z)$ corresponds the following program:

$$x - y - z \geqslant 0,$$

$$x + y - z \geqslant 0,$$

$$x, y, z \in \{-1, 1\}$$

and we want to find values to x, y, z that maximize the number of satisfied inequalities.

We will make use of a version of linear programming that is efficiently approximable in NC.

Definition 8 (*Positive linear programming (Luby and Nisan [22])*). A maximization linear program is said to be an instance of *positive linear programming* (PLP for short) if it is written as $\max\{\mathbf{c}^{\mathrm{T}}\mathbf{x} : A\mathbf{x} \leqslant \mathbf{b}, \mathbf{x} \geqslant 0\}$ where all the entries of A, $\mathbf{b}$ and $\mathbf{c}$ are non-negative.

Maximization positive linear programs are also called *fractional packing* problems. Luby and Nisan developed a very efficient algorithm for approximating positive linear programming problems.

Theorem 1 (*Luby and Nisan [22]*). *There exists a parallel algorithm that given in input a maximization instance P of PLP and a rational $\varepsilon > 0$ returns a feasible solution for P whose cost is at least $(1-\varepsilon)$ times the optimum. Furthermore, the algorithm runs in time polynomial in $1/\varepsilon$ and $\log N$ using $\mathrm{O}(N)$ processors, where N is the number of non-zero entries in P.*

3. Reductions among problems

Theorem 2. *For any constant k and function $d(n)$,* MAX kCONJ-$d(n)$ *belongs to* Max FSBLIP$(d(n))$.

Proof. Our formulation is similar to that of Max Capacity Representatives given in [2, Section 3]. Let $\{C_1, \ldots, C_m\}$ be a set of k-ary domain-d conjunctive constraints over $x_1, \ldots, x_n$, and $w_1, \ldots, w_m$ be associated non-negative weights. We use two $0/1$ variables t_j and f_j for any constraint, and we use a d-valued variable y_i for any variable x_i. The ILP is

$$\begin{array}{lll} \max & \sum_j w_j t_i & \\ \text{s.t.} & & \\ & (1-t_j)+f_j \geqslant 1 & \forall j=1,\ldots,m, \\ & d(1-f_j)+y_i \geqslant v & \forall j=1,\ldots,m \ \forall [x_i=v] \in C_j, \\ & d(1-f_j)-y_i \geqslant -v & \forall j=1,\ldots,m \ \forall [x_i=v] \in C_j. \end{array}$$

Notice that each objective variable t_j appears in a unique objective constraint, each variable f_j in an objective constraints occurs in at most $2k$ other constraints dominating each of them, and, finally, any constraints has width 2. $\square$

Theorem 3. *If* Max kConj-d *is r-approximate* (*in* NC) *and* $d^k = \text{poly}(n)$, *then* Max kCSP-d *is r-approximable* (*in* NC).

Proof. For any constraint C_j of weight w_j, let s be the number of satisfying assignments to its variables (note that $s \leqslant d^k$). Then we can express C_j as the disjunction of s conjunctive constraints $K_j^1, \ldots, K_j^s$, each one enforcing one of the satisfying assignments of C_j. Observe that any (global) assignment, satisfies at most one of the K_j^i constraints and satisfies one if and only if satisfies C_j. Let us substitute C_j with the $K_j^1, \ldots, K_j^s$ constraints, and give weight w_j to all of them. We repeat the same substitution for any constraint. The new instance is equivalent to the former, in the sense that they share the same set of feasible solutions, and the cost of each solution is always the same. Observe that the substitution process can be done also in parallel for all the constraints. $\square$

Theorem 4. Max 2Conj-d *is r-approximable* (*in* NC) *if and only if* Max Capacity Representatives-d *is r-approximable* (*in* NC).

Proof. It is easy to see that the two problems are equal. Without loss of generality we can assume that any set in a Max Capacity Representatives-d instance has exactly d elements (add dummy elements and give weight zero to the pairs corresponding to such elements) and that in a Max 2Conj-d instance with n variables there are all the possible $\binom{n}{2}d^2$ conjunctive constraints (add the missing constraints with weight zero). Now, the equivalence is immediate: every set S_i in Max Capacity Representatives-d corresponds to a d-valued variable $s_i = a$, $a = 0, 1, \ldots, d-1$, meaning that the representative of set S_i is a; the choice of a representative corresponds to the value assigned to the variable; to a pair of representatives in different sets S_i, S_j corresponds a conjunctive constraint $s_i = a \wedge s_j = b$; the weight of a constraint is that of the edge from which it was derived. Clearly, starting from an instance of Max Capacity Representatives-d we construct (in NC) an instance of Max 2Conj-d such that its feasible solutions are also feasible solutions of the same cost for MAX CAPACITY REPRESENTATIVES-d and vice versa. The theorem thus readily follows. □

Theorem 5. MAX kCONJ-2 *can be expressed as a* Max FMIP *problem with* $p(n) = 1$, $k' = 1$ *and* $\sigma = k$.

Proof. Let φ be an instance of MAX kCONJ-2. We have a variable $y_i \in \{-1, 1\}$ for any variable x_i of φ. For any constraint C_j, let P_j (resp. N_j) be the set of indices of variables that are assigned to 1 (resp. 0) in C_j. Let k_j be the arity of C_j. Then C_j is expressible as

$$\bigwedge_{i \in P_j} [x_i = 1] \wedge \bigwedge_{i \in N_j} [x_i = 0].$$

We translate C_j into the constraint $\sum_{i \in P_j} y_i + \sum_{i \in N_j} -y_i \geqslant k_j$. Under the understanding that $\{-1, 1\}$ assignments to y_i should be mapped to $\{0, 1\}$ assignments for x_i (i.e. $x_i = (1 + y_i)/2$), the two constraints are equivalent. We repeat the translation for any constraint, and the theorem thus follows. □

4. Positive results: algorithms

In this section we give approximation algorithms for the problems of the classes Max FSBLIP$(d(n))$ and Max FMIP. The approximation results for the problems of the first class are obtained through the approximability of MAX kCONJ-d while for the second class the method we use is straightforward.

4.1. *The approximability of MAX kCONJ-d Problem*

We now consider a linear programming relaxation of MAX kCONJ-d. We have a variable z_j for any constraint C_j, with the intended meaning that $z_j = 1$ when C_j is satisfied and $z_j = 0$ otherwise. We also have a variable $t_{i,v}$ for any variable x_i and any value $v \in [d]$, meaning that $t_{i,v} = 1$ if $x_i = v$ and $t_{i,v} = 0$ otherwise.

$$\begin{aligned} \max \quad & \sum_j w_j z_j \\ \text{s.t.} \quad & \\ & z_j \leqslant t_{i,v} && \forall\, i, v, [x_i = v] \in C_j, \\ & \sum_{v \in [d]} t_{i,v} = 1, && \\ & 0 \leqslant t_{i,v} \leqslant 1 && \forall\, i \in [n],\ \forall v \in [d]. \end{aligned} \qquad \text{(CONJ)}$$

Lemma 1. *The linear program* (CONJ) *is* $(1 - o(1))$*-approximable in* NC.

Proof. The proof is a generalization of a result of [28]. We reformulate (CONJ) in a slightly different way.

$$
\begin{aligned}
\max\quad & \sum_j w_j z_j \\
\text{s.t.}\quad & \\
& z_j + \sum_{u \neq v} t_{i,u} \leqslant 1 \qquad \forall\, i, v, [x_i = v] \in C_j, \\
& \sum_{v \in [d]} t_{i,v} = 1, \\
& 0 \leqslant t_{i,v} \leqslant 1 \qquad \forall\, i \in [n],\ \forall v \in [d].
\end{aligned}
\tag{CONJ1}
$$

We just used the fact that

$$
t_{i,v} = 1 - \sum_{u \neq v} t_{i,u}.
$$

Observe that (CONJ1) would be in PLP form if it had no equality constraints. For any i, let occ_i be the total weights of the constraints j where a variable $t_{i,v}$ occurs together with z_j, that is $occ_i = \sum_{j,v:x_i=v \text{ occurs in } c_j} w_j$. Let also define $occ = \sum_i occ_i$; we observe that $occ \leqslant k \sum_j w_j$, and that the optimum of (CONJ) and (CONJ1) is at least $\sum_j w_j/d \geqslant occ/kd$. Let us consider the PLP

$$
\begin{aligned}
\max\quad & \sum_j w_j z_j + \sum_i occ_i \sum_v t_{i,v} \\
\text{s.t.}\quad & \\
& z_j + \sum_{u \neq v} t_{i,u} \leqslant 1 \qquad \forall\, i, v, [x_i = v] \in C_j, \\
& \sum_{v \in [d]} t_{i,v} \leqslant 1, \\
& 0 \leqslant t_{i,v} \leqslant 1 \qquad \forall\, i \in [n],\ \forall v \in [d].
\end{aligned}
\tag{CONJ2}
$$

Claim 6. *If* $(\mathbf{z}, \mathbf{t})$ *is feasible for* (*CONJ*1) *and has cost* c, *then it is also feasible for* (*CONJ*2) *and has cost* $c + occ$.

Claim 7. *Given* $(\mathbf{z}, \mathbf{t})$ *feasible for* (*CONJ*2) *of cost* $c + occ$, *then we can find in* NC *a solution* $(\mathbf{z}', \mathbf{t}')$ *of cost* c *that is feasible for* (*CONJ*1).

Proof (Of the Claim)**.** We define $t'_{i,0} = 1 - \sum_{v \neq 0} t_{i,v}$ and $t'_{i,v} = t_{i,v}$ for $v \neq 0$; and, furthermore, we define

$$
z'_j = \min\{z_j, \min_{[x_i = v] \in C_j} t_{i,v}\}.
$$

The solution $(\mathbf{z}', \mathbf{t}')$ is clearly computable in NC given $(\mathbf{z}, \mathbf{t})$, and is feasible for (CONJ1) and (CONJ). To prove the claim, it remains to show

$$\sum_j w_j z'_j \geqslant \sum_j w_j z_j + \sum_i \Big(occ_i \sum_v t_{i,v} \Big) - occ. \tag{2}$$

Let J be the set of indices j such that $z'_j < z_j$. For any $j \in J$ we clearly have $z'_j = \min_{[x_i=v]\in C_j} t_{i,v}$. We call $i(j)$ and $v(j)$ the index and the value such that $z'_j = t_{i(j),v(j)}$. By the feasibility of $(\mathbf{z}, \mathbf{t})$, we have $z_j \leqslant 1 - \sum_{u \neq v} t_{i,u} \leqslant 1 - \sum_{u \neq v(j)} t_{i(j),u}$. We prove Eq. (2) in two steps.

$$\begin{aligned}
\sum_j w_j (z_j - z'_j) &= \sum_{j \in J} w_j (z_j - z'_j) \\
&\leqslant \sum_{j \in J} w_j \Big(1 - \Big(\sum_{u \neq v(j)} t_{i(j),u} \Big) - t_{i(j),v(j))} \Big) \\
&= \sum_{j \in J} w_j \Big(1 - \sum_v t_{i(j),v} \Big).
\end{aligned}$$

On the other hand,

$$\begin{aligned}
occ - \sum_i \Big(occ_i \sum_v t_{i,v} \Big) &= \sum_i occ_i \Big(1 - \sum_v t_{i,v} \Big) \\
&= \sum_i \sum_{j:x_i \text{ occurs in } c_j} w_j \Big(1 - \sum_v t_{i,v} \Big) \\
&= \sum_i \sum_{i:x_i \text{ occurs in } c_j} w_j \Big(1 - \sum_v t_{i,v} \Big) \\
&\geqslant \sum_{j \in J} w_j \Big(1 - \sum_v t_{i(j),v} \Big).
\end{aligned}$$

We have therefore established

$$occ - \sum_i \Big(occ_i \sum_v t_{i,v} \Big) \geqslant \sum_j w_j (z_j - z'_j),$$

which is equivalent to (2). □

Thus, the optimum of (CONJ2) is *occ* plus the optimum of (CONJl), i.e. it is at most $(kd + 1)$ times the optimum of (CONJl). As in [28], the lemma now follows by finding a $(1 - \mathrm{o}(1))$-approximate solution for (CONJ2), converting it into a solution for (CONJl) (and thus (CONJ)) and observing that it is still $(1 - \mathrm{o}(1))$-approximate. □

Remark 1. The above lemma shows that the linear programming relaxations derived from LP's of Max FSBLIP problems are approximable in parallel within any constant. Since these linear programming relaxations are instances of Positive Linear Programming, we have that this class of LP relaxations can be seen as an extension of positive linear programming.

Lemma 2 (*Random rounding for* MAX kCONJ-d). *Let* $(\mathbf{z}, \mathbf{t})$ *be a feasible solution for* (CONJ). *Consider the random assignment obtained by setting, for any* i, v

$$\mathbf{Pr}[x_i = v] = (k-1)/dk + t_{i,v}/k.$$

Then such an assignment has an average cost at least $\frac{1}{d^{k-1}} \sum_j w_j z_j$. *The analysis only assumes that the distribution is k-wise independent*.

Proof. It is sufficient to prove that any constraint C_j is satisfied with probability at least $\frac{1}{d^{k-1}} z_j$; the lemma will then follow by the linearity of expectation. Observe that if the atom $[x_i = v]$ occurs in C_j then $z_j \leqslant t_{i,v}$. Then

$$\Pr[C_j \text{ is satisfied}] \geqslant \left(\frac{k-1}{dk} + \frac{1}{k} z_j\right)^k \geqslant \frac{1}{d^{k-1}} z_j. \tag{3}$$

For the last inequality, we consider the function

$$f(z) = \frac{\left(\frac{k-1}{dk} + \frac{1}{k} z\right)^k}{z}$$

in the interval $0 \leqslant z \leqslant 1$, compute its first derivative, and show that f has a minimum in $z = 1/d$, that is $f(z) \geqslant f(1/d) = 1/d^{k-1}$, $\forall z, 0 \leqslant z \leqslant 1$. In the first inequality of Eq. (3) we have assumed that the random variables induced by the clause C_j are independent. □

Remark 2. The above analysis is tight and establishes that the integrality gap of (CONJ) is d^{k-1}. The bound is achieved, e.g. by the instance consisting of clauses $C_1, C_2, \ldots, C_{d^k}$ that are all possible size k (domain-d) conjunctions of $\{x_1, \ldots, x_k\}$.

Theorem 8. *For any* $d = d(n)$ *and* $k = k(n)$ *such that* $d^k = n^{\mathrm{O}(1)}$, *there is an* NC $(1/d^{k-1} - \mathrm{o}(1))$-*approximate algorithm for* MAX kCSP-d. *In particular, there is a* $(1/d - \mathrm{o}(1))$-*approximate* NC *algorithm for* MAX CAPACITY REPRESENTATIVES-d.

Proof. The first statement of the theorem follows from Theorem 3 and the Lemma 2, and the second statement follows from the Theorem 4. □

4.2. *The* Max FSBLIP *problems*

Borland et al. [2] have shown the following approximation result for the problems of the class Max FSBLIP$(d(n))$.

Theorem 9 (*Barland et al.* [2]). *For every problem in the class* Max FSBLIP$(d(n))$, *there is a constant p such that the problem is* $1/d(n)^p$-*approximable*.

Their proof is a generalization of the greedy technique used for the approximation of the Max NP problems in [24]. Note that the constant p in Theorem 9 is $p = l \cdot m$, where m is the width of a constraint and l is the maximum number of occurrences of any objective variable in the rest of program constraints (recall the definition of Max FSBLIP).

Now, if we take a MAX kCSP-d problem, and we want to express it as a Max FSBLIP(d), and then we use the FSBLIP approximation algorithm of Barland et al., what we get is

– in the translation from MAX kCSP-d to Max FSBLIP(d) we have $l = 2k$, $m = 2$, $o = 1$ (Theorem 2).
– the Barland et al. algorithm will have approximation ratio $1/d^{lm}$ which is $1/d^{4k}$ which is worse than our approximation ratio $1/d^{k-1}$.

So, given a Max kCSP-d instance, it is better to use our algorithm than to translate the instance into an instance of Max FSBLIP(d) and then use Barland's algorithm on it. In particular, for MAX CAPACITY REPRESENTATIVES-d we have $m = 2$ and $l = 1$. The sequential factor of [2] is $1/d^2$ while our factor is $(1/d - \varepsilon)$. This improvement is quite natural because the sequential result is obtained via a uniform probability distribution while we have used a distribution obtained from the fractional solution of the linear programming relaxation to the problem. Finally, since Max NP is properly contained in Max FSBLIP(2) [2] we derive also constant parallel approximability of improved factors for the Max NP problems.

4.3. *The* Max FMIP *problems*

A prototypical problem in Max FMIP is MAX MAJORITY SAT, which is the variation of MAX SAT where a clause is satisfied if at least half the literals (rather than at least one) are satisfied. Barland et al. [2] showed that this class contains only constant-approximable problems (using, once more, the syntactic structure of integer programs) and gave a structural explanation of this result.

It is easy to find a 2-approximate solution for MAX MAJORITY SAT. Any clause is either satisfied by the assignment $x_i = 0, \forall i$, or by the assignment $x_i = 1, \forall i$. Thus one of the two assignments satisfies at least half the clauses. [3]

For the Max FMIP problem in its general setting, we present a slight improvement and simplification over the approximation result of Barland et al. [2].

Theorem 10. *Given an instance of a* Max FMIP *problem, the random assignment where each variable is set to* $-k$ *or to* k *with probability* $1/2$ *independently at random satisfies each constraint with probability at least* $1/2^{1+\lceil \sigma/k \rceil}$, *provided that the constraint is satisfiable.*

Proof. Consider a constraint $\sum_i a_i x_i \geqslant b$. If the constraint is satisfiable, then $\sum_i |a_i| k \geqslant b$. Since the a_i are integers, there must be a set J of at most $\lceil b/k \rceil$ indices such that $\sum_{i \in J} |a_i| k \geqslant b$. Under the uniform distribution, with probability at least $1/2^{|J|} \geqslant 1/2^{\lceil b/k \rceil}$ we will have $\sum_{i \in J} a_i x_i \geqslant b$. It is also easy to see that, by symmetry, with probability at least $1/2$ we have $\sum_{i \notin J} a_i x_i \geqslant 0$.

The theorem thus follows since for the whole set of constraints, $b_j \leqslant \sigma, \forall j$. □

The above theorem can be derandomized in NC through the techniques of Karger and Koller [14].

[3] This nice idea is due to Michel Goemans.

Notice that the factor of approximation given by the above theorem improves the constant-factor approximation of Barland et al. Indeed, Barland et al. [2] find an assignment to variables of the program that satisfies any restriction with probability at least $1/(2 \cdot 3^{\sigma})$, while we find a solution that satisfies any restriction with probability at least $1/(2 \cdot 2^{\lceil \sigma/k \rceil})$. Finally, we remark that our proof is quite simple compared to that of [2] (they use again a greedy technique which results a bit more complicated than the case of Max FSBLIP problems, since the constraints have no bounded width anymore).

5. Negative results: hardness of approximation

We first define Probabilistically Checkable Proof Systems and Multi-Prover One-Round Proof Systems. Our notation merges the notations of [4] and [5]. For an integer d, we denote by $[d]^*$ the set of all strings over $[d]$.

Definition 9 (*Verifier*). A verifier V for a language L is a randomized polynomial time oracle Turing machine. V receives in input a string x and has oracle access to a string π that is an alleged proof that $x \in L$.

Definition 10 (PCP *and* MIP). Let $c, s, r, q, d : \mathcal{Z}^+ \to \mathcal{Z}^+$ such that $0 \leqslant s(n) < c(n) \leqslant 1$ for any n; we say that a language L belongs to $\mathsf{PCP}_{c,s}[r, q, d]$ if there exists a verifier V such that

(1) For any input string x and oracle proof $\pi \in [d(n)]^*$, V queries at most $q(n)$ entries of π and uses at most $\mathrm{O}(r(n))$ random bits;
(2) For any $x \in L$, there exists a $\pi \in [d(n)]^*$ such that the probability that V accepts x with oracle π is at least $c(n)$;
(3) For any $x \notin L$, for any $\pi \in [d(n)]^*$; the probability that V accepts x with oracle π is at most $s(n)$.

The class $\mathsf{MIP}_{c,s}[r, q, d]$ is similar, with the only difference that π is presented as a sequence of q strings $\pi_1, \ldots, \pi_q$, where $\pi_i \in [d]^*$, and V has the further restriction that it can read at most one entry of any π_i.

From the above definition it follows that $\mathsf{MIP}_{c,s}[r, q, d] \subseteq \mathsf{PCP}_{c,s}[r, q, d]$ for any choice of the parameters. The following result is folklore.

Theorem 11. *If* MAX kCSP-$(d(n))$ *is* $\rho(n)$*-approximable, then, for any* $c(n)$ *and* $s(n)$ *such that* $s(n)/c(n) < \rho(n^{\mathrm{O}(1)} 2^{\mathrm{O}(r(n))})$, *it holds*

$$\mathsf{PCP}_{c(n),s(n)}[r(n), k(n), d(n)] \subseteq \mathsf{DTIME}(2^{\mathrm{O}(r(n)+k(n)\log d(n))}).$$

We prove several non-approximability results in the following theorem.

Theorem 12. *The following statements hold* (n *is the size of the input*):

(1) *A constant* $c > 0$ *exists such that, for any constant* $d \geqslant 2$, *it is* NP*-hard to approximate* MAX 2CSP-d *within* $1/d^c$. *Furthermore, for any* $\varepsilon > 0$, *it is infeasible to approximate* MAX 2CSP-$(\log n)$ *within* $2^{\log^{1-\varepsilon} n}$ *unless* $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log^{\mathrm{O}(1/\varepsilon)} n})$.

(2) *For any constant d, for any* $k \geqslant 3$, *for any* $\varepsilon > 0$, *it is* NP-*hard to approximate* MAX kCSP-d *within* $1/d^{\lfloor k/3 \rfloor} + \varepsilon$.
(3) *Constants k and c exist such that it is* NP-*hard to approximate* MAX kCSP-$(\log n)$ *within* $1/\log n^c$.
(4) *For any* $k \geqslant 5$, *any* $\varepsilon > 0$, *it is* NP-*hard to approximate* MAX kCSP *within* $2^{\log^{1/3-\varepsilon} n}$.
(5) *For any* $\varepsilon > 0$, *a constant* $k = O(1/\varepsilon)$ *exists such that it is* NP-*hard to approximate* MAX kCSP *within* $2^{\log^{1-\varepsilon} n}$.
(6) *For any* $\varepsilon > 0$, Max FMIP *problems are hard to approximate within* $1/2^{\lfloor \sigma/3 \rfloor} + \varepsilon$.

Proof (Sketch)**.** For (1), Raz [26] has shown that a constant $c' > 0$ exists such that, for any $k : \mathcal{Z}^+ \to \mathcal{Z}^+$, $\mathsf{NP} \subseteq \mathsf{MIP}_{1,2^{-ck(n)}}[k(n)\log n, 2, 3^{k(n)}]$. The first part of the claim follows by setting $k(n) = \lfloor \log_3 d(n) \rfloor$; the second part by setting $k(n) = \log^{O(1/\varepsilon)}(n)$. Next, for (2), Håstad [13] has shown that for any $\varepsilon > 0$, for any fixed prime p, $\mathsf{NP} = \mathsf{PCP}_{1-\varepsilon, 1/p+\varepsilon}[\log, 3, p]$. The claim follows by choosing $p = d$ and repeating the proof $k/3$ times. Further, (3)–(5) are re-statements of the results of Raz and Safra [27], and Arora and Sudan [1] using Theorem 2. Finally, (6) follows from the hardness of MAX kCSP-2 and from Theorem 5. □

From the above theorem we easily deduce a couple of corollaries. First, from part (1) of Theorems 3 and 4 we have

Corollary 13. MAX CAPACITY REPRESENTATIVES$(\log n)$ *is not constant-approximable*.

This result solves an open question of Barland et al. namely whether the problem MAX CAPACITY REPRESENTATIVES$(\log n)$ is constant-approximable.

From Corollary 13 we derive easily the following structural result.

Corollary 14. *The classes* Max FSBLIP(2), Max FSBLIP(log) *and* Max FSBLIP(poly) *form a proper hierarchy*.

Note that we separate the first two classes since MAX CAPACITY REPRESENTATIVES(log) is not constant-approximable, the separation of the two last classes derives from a result of Bellare [3] stating that MAX CAPACITY REPRESENTATIVES which belongs to Max FSBLIP(poly) is not log-approximable.

Finally, it is worth to mention the almost tight non-approximability result for the problems of class Max FMIP.

6. Concluding remarks

We show that the sequential results of Barland et al. obtained in the framework of integer programs of restricted format hold also in the parallel setting and thus we come out with a new common framework for both settings (a previous common framework is that of logical definability of Papadimitriou and Yannakakis). Moreover, we give substantial improvements in two directions. First, our factors of approximation are close to the best possible while the ones in Barland et al. were far from, due to the use of the greedy technique they employed. Secondly, we show tight non-approximability results for NON-BOOLEAN CONSTRAINT SATISFACTION and MAX CAPACITY REPRESENTATIVES. (This is the first time the approximability and non-approximability of both problems is addressed in its full generality.) The non-approximability result of the last problem allow us to establish that the classes Max FSBLIP(2), Max FSBLIP(log) and Max FSBLIP(poly) form a proper hierarchy.

Acknowledgements

We thank an anonymous referee for useful comments that improved the readability of the paper.

References

[1] S. Arora, M. Sudan, Improved low degree testing and its applications, in: Proc. 29th ACM STOC, 1997, pp. 485–495.
[2] I. Barland, P.G. Kolaitis, M.N. Thakur, Integer programming as a framework for optimization and approximability, J. Comput. System Sci. 57 (2) (1998) 144–161.
[3] M. Bellare, Interactive proofs and approximation: reductions from two provers in one round, in: Proc. 2nd Israel Symposium on Theory and Computing Systems, IEEE, Silver Spring, MD, 1993.
[4] M. Bellare, S. Goldwasser, C. Lund, A. Russell, Efficient probabilistically checkable proofs and applications to approximation, in: Proc. 25th ACM STOC, 1993, pp. 294–304; See also the errata sheet in Proc. STOC'94.
[5] M. BeIlare, O. Goldreich, M. Sudan, Free bits, PCP's and non-approximability, SIAM J. Comput. 27 (3) (1998) 804–915.
[6] D.P. Bovet, P. Crescenzi, Introduction to the Theory of Complexity, Prentice-Hall, Englewood Cliffs, NJ, 1993.
[7] N. Creignou, A dichotomy theorem for maximum generalized satisfiability problems, J. Comput. System Sci. 51 (3) (1995) 511–522.
[8] J. Díaz, M. Serna, P. Spirakis, J. Torán, Paradigms for Fast Parallel Approximability, Cambridge University Press, Cambridge, 1997.
[9] L. Engebretsen, Lower bounds for non-Boolean constraint satisfaction programs, Electronic Colloquium on Computational Complexity, Report No. 42, 2000.
[10] G. Even, J. Naor, S. Rao, B. Schieber, Divide-and-conquer approximation algorithms via spreading metrics, in: Proc. 36th IEEE FOCS, 1995, pp. 62–71.
[11] M.X. Goemans, D.P. Williamson, New 3/4-approximation algorithms for the maximum satisfiability problem, SIAM J. Discrete Math. 7 (1994) 656–666.
[12] M.X. Goemans, D.P. Williamson, The primal-dual method for approximation algorithms and its application to the network design problems, in: D. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing Company, MA, 1996.
[13] J. Håstad, Some optimal inapproximability results, J. Assoc. Comput. Mach. 48 (4) (2001) 798–859.
[14] D.R. Karger, D. Koller, (De)randomized construction of small sample spaces in $\mathcal{NC}$, in: Proc. 35th IEEE FOCS, 1994, pp. 252–263.
[15] S. Khanna, R. Motwani, Towards a syntactic characterization of PTAS, in: Proc. 28th ACM STOC, 1996, pp. 329–337.
[16] S. Khanna, R. Motwani, M. Sudan, U. Vazirani, On syntactic versus computational views of approximability, SIAM J. Comput. 30 (6) (2000) 1863–1920.
[17] S. Khanna, M. Sudan, L. Trevisan, D.P. Williamson, The approximability of constraint satisfaction problems, SIAM J. Comput. 30 (6) (2000) 1863–1920.

[18] P.G. Kolaitis, M.N. Thakur, Logical definability of NP optimization problems, Inform. and Comput. 115 (2) (1994) 321–353.
[19] P.G. Kolaitis, M.N. Thakur, Approximation properties of NP minimization classes, J. Comput. System Sci. 50 (1995) 391–411.
[20] H.C. Lau, O. Watanabe, Randomized approximation of the constraint satisfaction problem, Nordic J. Comput. 3 (1996) 405–424.
[21] N. Linial, E. London, Y. Rabinovich, The geometry of graphs and some of its algorithmic applications, Combinatorica 15 (2) (1995) 215–245.
[22] M. Luby, N. Nisan, A parallel approximation algorithm for positive linear programming, in: Proc. 25th ACM STOC, 1993, pp. 448–457.
[23] A. Panconesi, D. Ranjan, Quantifiers and approximations, Theoret. Comput. Sci. 107 (1993) 145–163.
[24] C.H. Papadimitriou, M. Yannakakis, Optimization, approximation, and complexity classes, J. Comput. System Sci. 43 (1991) 425–440.
[25] P. Raghavan, C.D. Thompson, Randomized rounding: a technique for provably good algorithms and algorithmic proofs, Combinatorica 7 (1987) 365–374.
[26] R. Raz, A parallel repetition theorem, SIAM J. Comput. 27 (3) (1998) 763–803.
[27] R. Raz, S. Safra, A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP, in: Proc. 29th ACM STOC, 1997, pp. 475–484.
[28] L. Trevisan, Parallel approximation algorithms by positive linear programming, Algorithmica 21 (1998) 72–88.
[29] F. Xhafa, On parallel versus sequential approximability: complexity and approximation results, Ph.D. Thesis, Universitat Politècnica de Catalunya, 1998.