PART I

Paradigmatic Problems

Chapter 1

Optimal Satisfiability

1.1. Introduction

Given a set of constraints defined on Boolean variables, a satisfiability problem, also called a Boolean constraint satisfaction problem, consists of deciding whether there exists an assignment of values to the variables that satisfies all the constraints (and possibly establishing such an assignment). Often, such an assignment does not exist and, in this case, it is natural to seek an assignment that satisfies a maximum number of constraints or minimizes the number of non-satisfied constraints.

An example of a Boolean constraint satisfaction problem is the problem known as SAT, which consists of deciding whether a propositional formula (expressed as a conjunction of disjunctions) is satisfiable or not. SAT was the first problem shown to be *NP*-complete by Cook [COO 71] and Levin [LEV 73] and it has remained a central problem in the study of *NP*-hardness of optimization problems [GAR 79]. The *NP*-completeness of SAT asserts that no algorithm for this problem can be efficient in the worst case, under the hypothesis *P*≠*NP*. Nevertheless, in practice many efficient algorithms exist for solving the SAT problem.

Satisfiability problems have direct applications in various domains such as operations research, artificial intelligence and system architecture. For example, in operations research, the graph-coloring problem can be modeled as an instance of SAT. To decide whether a graph with n vertices can be colored with k colors, we consider $k \times n$ Boolean variables, x_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, k$, where x_{ij} takes the value true if

Chapter written by Cristina BAZGAN.

and only if the vertex i is assigned the color j. Hoos [HOO 98] studied the effectiveness of various modelings of the graph-coloring problem as a satisfiability problem where we apply a specific local search algorithm to the instance of the obtained satisfiability problem. The Steiner tree problem, widely studied in operations research, contributes to network design and routing applications. In [JIA 95], the authors reduced this problem to a problem that consists of finding an assignment that maximizes the number of satisfied constraints. Certain scheduling problems have been solved by using modeling in terms of a satisfiability problem [CRA 94]. Testing various properties of graphs or hypergraphs is also a problem that reduces to a satisfiability problem. In artificial intelligence, an interesting application is the planning problem that can be represented as a set of constraints such that every satisfying assignment corresponds to a valid plan (see [KAU 92] for such a modeling). Other applications in artificial intelligence are: learning from examples, establishing the coherence of a system of rules of a knowledge base, and constructing inferences in a knowledge base. In the design of electrical circuits, we generally wish to construct a circuit with certain functionalities (described by a Boolean function) that satisfy various constraints justified by technological considerations of reliability or availability, such as minimizing the number of gates used, minimizing the depth of the circuit or only using certain types of gates.

Satisfiability problems also have other applications in automatic reasoning, computer vision, databases, robotics, and computer-assisted design. Gu, Purdom, Franco and Wah wrote an overview article [GU 97] that cites many applications of satisfiability problems (about 250 references).

Faced with a satisfiability problem, we can either study it from the theoretical point of view (establish its exact or approximate complexity, construct algorithms that guarantee an exact or approximate solution), or solve it from the practical point of view. Among the most effective methods for the practical solution of satisfiability problems are local search, Tabu search, and simulated annealing. For further details, refer to [GU 97] and [GEN 99], which offer a summary of the majority of practical algorithms for satisfiability problems.

In this chapter, we present the principal results of exact and approximation complexity for satisfiability problems according to the type of Boolean functions that participate in the constraints. Our goal is not to present exhaustively all the results that exist in the literature but rather to identify the most studied problems and to introduce the basic concepts and algorithms. The majority of satisfiability problems are hard. It is therefore advantageous, both from the theoretical and practical points of view, to identify some specific cases that are easier. We have chosen to present the most studied specific cases: planar instances, instances with a bounded number of occurrences of each variable, and dense instances. Several optimization problems can be modeled as a satisfiability problem with an additional global constraint on the set of feasible solutions. In particular, the MIN BISECTION problem, whose approximation complexity

has not yet been established, can be modeled as a satisfiability problem where the set of feasible solutions is the set of the balanced assignments (with as many variables set to 0 as to 1). We also present a few results obtained on satisfiability problems under this global constraint.

Readers who wish to acquire a deeper knowledge of the complexity of satisfiability problems should consult the monograph by Creignou, Khanna and Sudan [CRE 01], where the proofs of the majority of important results in this domain can be found and that cover, besides the results presented here, other aspects such as counting complexity and function representation complexity, as well as other satisfiability problems. Note also that there is an electronic compendium by Crescenzi and Kann [CRE 95b], which regroups known results of approximation complexity for optimization problems, in particular for satisfiability problems.

This chapter is structured as follows. In section 1.2, we introduce the types of Boolean functions that we will use and we define the decision and optimization problems considered. In section 1.3, we study decision problems, and in section 1.4, maximization and minimization problems. We then discuss a few specific instances of satisfiability problems: planar instances (section 1.5.1), dense instances (section 1.5.2), and instances with a bounded number of occurrences of each variable (section 1.5.3). We also present the complexity of satisfiability problems when the set of feasible solutions is restricted to balanced assignments (section 1.6). We close our chapter with a brief conclusion (section 1.7).

1.2. Preliminaries

An instance of a satisfiability problem is a set of m constraints $C_1, \ldots, C_m$ defined on a set of n variables $x_1, \ldots, x_n$. A constraint C_j is the application of a Boolean function $f : \{0,1\}^k \to \{0,1\}$ to a subset of variables $x_{i_1}, \ldots, x_{i_k}$, where $i_1, \ldots, i_k \in \{1, \ldots, n\}$. This constraint is also expressed as $f(x_{i_1}, \ldots, x_{i_k})$. An assignment $x_i = v_i$, for $i = 1, \ldots, n$, where $v_i \in \{0,1\}$, satisfies the constraint $f(x_{i_1}, \ldots, x_{i_k})$ if and only if $f(v_{i_1}, \ldots, v_{i_k}) = 1$.

A literal is a variable x_i (positive literal) or its negation $\bar{x}_i$ (negative literal).

EXAMPLE 1.1.– A few examples of Boolean functions used to define constraints:

– $T(x) = x, F(x) = \bar{x}$;

– $OR_i^k(x_1, \ldots, x_k) = \bar{x}_1 \vee \ldots \vee \bar{x}_i \vee x_{i+1} \vee \ldots \vee x_k$, where $i \leqslant k$ represents the number of negative literals in the disjunction;

– $AND_i^k(x_1, \ldots, x_k) = \bar{x}_1 \wedge \ldots \wedge \bar{x}_i \wedge x_{i+1} \wedge \ldots \wedge x_k$, where $i \leqslant k$ represents the number of negative literals in the conjunction;

– $XOR^k(x_1, \ldots, x_k) = x_1 \oplus \ldots \oplus x_k$, where $\oplus$ represents the "exclusive or" operation $(0 \oplus 0 = 0, 1 \oplus 0 = 1, 0 \oplus 1 = 1, 1 \oplus 1 = 0)$;

– $XNOR^k(x_1, \ldots, x_k) = \overline{x_1 \oplus \ldots \oplus x_k}$.

A constraint can also be represented as a Boolean expression that can be in various forms. An expression is in conjunctive normal form (CNF) if it is in the form $c_1 \wedge \ldots \wedge c_m$, where each c_t is a disjunctive clause, that is in the form $\ell_{t_1} \vee \ldots \vee \ell_{t_p}$, where ℓ_{t_i}, $i = 1, \ldots, p$ are literals. An expression is in disjunctive normal form (DNF) if it is in the form $c_1 \vee \ldots \vee c_m$, where each c_t is a conjunctive clause, that is in the form $\ell_{t_1} \wedge \ldots \wedge \ell_{t_p}$, where ℓ_{t_i}, $i = 1, \ldots, p$ are literals. A kCNF (or kDNF) expression is a CNF (or DNF) expression in which each clause contains at most k literals.

Note that if each constraint of a satisfiability problem is represented by a CNF expression, the set of constraints of the problem can itself be represented by a CNF expression that corresponds to the conjunction of the previous expressions.

We consider various satisfiability problems according to the type of Boolean functions used to define the constraints. Let $\mathcal{F}$ be a finite set of Boolean functions. A *$\mathcal{F}$-set of constraints* is a set of constraints that only use functions that belong to $\mathcal{F}$. An assignment satisfies an $\mathcal{F}$-set of constraints if and only if it satisfies each constraint in the constraint set.

1.2.1. *Constraint satisfaction problems: decision and optimization versions*

In this section we define the classes of problems that we are going to study. This concerns decision and optimization versions of satisfiability problems.

The decision version of a problem consists of establishing whether this problem allows at least one solution; its search version consists of finding a solution if any exist. The optimization version of a problem consists of finding a solution that maximizes or minimizes a suitable function.

DEFINITION 1.1.– *The satisfiability problem* SAT($\mathcal{F}$) *consists of deciding whether there exists an assignment that satisfies an $\mathcal{F}$-set of constraints. The search problem associated with the decision problem* SAT($\mathcal{F}$) *consists of finding an assignment that satisfies an $\mathcal{F}$-set of constraints if such an assignment exists or then returning "no" otherwise.*

In this chapter, we will see that whenever we can solve the decision problem SAT($\mathcal{F}$) in polynomial time, we can also find a solution for the satisfiable instances and therefore solve the associated search problem in polynomial time.

It is normal practice to distinguish certain variants of SAT($\mathcal{F}$) where each function of $\mathcal{F}$ depends on at most (or exactly) k variables. These variants are expressed as kSAT($\mathcal{F}$) (or EkSAT($\mathcal{F}$)).

We now present a few classical decision problems as well as the corresponding satisfiability problem SAT($\mathcal{F}$):

– SAT is the problem that consists of deciding whether a set of disjunctive clauses defined on n Boolean variables is satisfiable. It corresponds to the SAT($\mathcal{F}$) problem, where $\mathcal{F}$ is the set of OR_i^k functions, for $k \leqslant n$.

– CONJ is the problem that consists of deciding whether a set of conjunctive clauses defined on n Boolean variables is satisfiable. It corresponds to the SAT($\mathcal{F}$) problem, where $\mathcal{F}$ is the set of AND_i^k functions, for $k \leqslant n$.

– LIN2 is the problem that consists of deciding whether a set of linear equations defined on n Boolean variables is satisfiable. It corresponds to the SAT($\mathcal{F}$) problem, where $\mathcal{F}$ is the set of XOR^k, $XNOR^k$ functions, for $k \leqslant n$.

– 2SAT is the version of SAT where each disjunctive clause has at most two literals, and it corresponds to 2SAT($\mathcal{F}$), where $\mathcal{F}$ is the set of OR_i^k functions, for $k \leqslant 2$.

– E3SAT is the version of SAT where each disjunctive clause has exactly three literals, and it corresponds to SAT($\{OR_0^3, OR_1^3, OR_2^3, OR_3^3\}$).

DEFINITION 1.2.– *The maximization problem* MAX SAT($\mathcal{F}$) *consists of establishing an assignment that satisfies a maximum number of constraints from an $\mathcal{F}$-set of constraints.*

For example, the MAX CUT problem, which consists of partitioning the set of vertices of a graph into two parts such that the number of edges whose extremities belong to different parts is maximum, can be formulated as a problem of the type MAX SAT($\{XOR^2\}$) as follows. Considering a graph G, an instance of MAX CUT, we associate with each vertex i a variable x_i and with each edge (i, j) of G the constraint $XOR^2(x_i, x_j)$.

DEFINITION 1.3.– *The minimization problem* MIN SAT DELETION($\mathcal{F}$) *consists of establishing an assignment that minimizes the number of non-satisfied constraints from an $\mathcal{F}$-set of constraints, which corresponds with the minimum number of constraints to remove so that the remaining constraints are satisfied.*

MIN SAT DELETION($\mathcal{F}$) allows us to model certain minimization problems naturally. For example, the s-t MIN CUT problem in a non-directed graph, which consists of partitioning the set of vertices of a graph into two parts such that s and t belong to different parts and such that the number of edges whose extremities belong to different parts is minimum, can be formulated as a problem of the type MIN SAT DELETION($\{XNOR^2\} \cup \{T, F\}$) as follows. Considering a graph G, an instance of s-t MIN CUT, we associate with each vertex i a variable x_i and with each edge (i, j) of G the constraint $XNOR^2(x_i, x_j)$. Furthermore, we add the constraints $T(x_s)$ and $F(x_t)$.

COMMENT 1.1.–

1) The problems MAX SAT($\mathcal{F}$) and MIN SAT DELETION($\mathcal{F}$) are clearly related. Indeed, considering an instance I of MAX SAT($\mathcal{F}$) with m constraints, an optimal solution for the instance I of MAX SAT($\mathcal{F}$) of value $opt_{\text{MAX SAT}(\mathcal{F})}(I)$ is also an optimal solution of the instance I of the MIN SAT DELETION($\mathcal{F}$) problem of value $opt_{\text{MIN SAT DELETION}(\mathcal{F})}(I)$= m-$opt_{\text{MAX SAT}(\mathcal{F})}(I)$. Therefore, the exact complexities of MAX SAT($\mathcal{F}$) and MIN SAT DELETION($\mathcal{F}$) coincide. However, the approximation complexities of the two problems can be very different as we will see in what follows.

2) In the literature, we also define the MIN SAT($\mathcal{F}$) problem that consists of establishing an assignment that minimizes the number of satisfied constraints. For example, in the compendium of Crescenzi and Kann [CRE 95b], the MIN SAT problem consists of establishing an assignment that minimizes the number of satisfied clauses from a set of disjunctive clauses. Note that MIN SAT($\mathcal{F}$) is equivalent, from the exact and approximation point of view, to MIN SAT DELETION($\mathcal{F}'$), where $\mathcal{F}'$ is the set of functions that are complementary to the functions of $\mathcal{F}$. For example, finding an assignment that minimizes the number of constraints satisfied among the constraints $x_1 \vee x_2, x_1 \vee \bar{x}_3, x_2 \vee x_3, \bar{x}_1 \vee \bar{x}_2$ is equivalent to finding an assignment that minimizes the number of non-satisfied constraints among the constraints $\bar{x}_1 \wedge \bar{x}_2, \bar{x}_1 \wedge x_3, \bar{x}_2 \wedge \bar{x}_3, x_1 \wedge x_2$. Thus the MIN SAT problem is equivalent to MIN SAT DELETION($\mathcal{F}$) where the constraints are conjunctive clauses (the problem called MIN CONJ DELETION). In what follows, we will consider only the MIN SAT DELETION($\mathcal{F}$) problem.

1.2.2. *Constraint types*

The complexity of SAT($\mathcal{F}$) as well as the exact and approximation complexities of MAX SAT($\mathcal{F}$) and MIN SAT DELETION($\mathcal{F}$) depend on the types of Boolean functions of the set $\mathcal{F}$. In this section, we describe the types of Boolean functions that have been most studied and that we will discuss in the rest of the chapter.

A Boolean function f is:

– *0-valid* if $f(0, \ldots, 0) = 1$;

– *1-valid* if $f(1, \ldots, 1) = 1$;

– *Horn* if it can be expressed as a CNF expression that has at most one positive literal in each clause;

– *anti-Horn* if it can be expressed as a CNF expression that has at most one negative literal in each clause;

– *affine* if it can be expressed as a conjunction of linear equations on the Galois body $GF(2)$, that is as a conjunction of equations of type $x_{i_1} \oplus \ldots \oplus x_{i_p} = 0$ or $x_{j_1} \oplus \ldots \oplus x_{j_q} = 1$;

– *bijunctive* if it can be expressed as a 2CNF expression;

– *2-monotone* if it can be expressed as a DNF expression in the form $x_{i_1} \wedge \ldots \wedge x_{i_p}$ or $\bar{x}_{\ell_1} \wedge \ldots \wedge \bar{x}_{\ell_q}$ or $(x_{i_1} \wedge \ldots \wedge x_{i_p}) \vee (\bar{x}_{\ell_1} \wedge \ldots \wedge \bar{x}_{\ell_q})$. Note that a 2-monotone function is both Horn and anti-Horn;

– *complement-closed* if for every assignment v we have $f(v) = f(\bar{v})$, where $\bar{v}$ is the complementary assignment to v.

We extend the previous notions to a set $\mathcal{F}$ of functions where all the functions of $\mathcal{F}$ have the required property. For example, if each function of $\mathcal{F}$ is Horn then the set $\mathcal{F}$ is Horn and the SAT($\mathcal{F}$) problem is called HORN SAT. Horn expressions play an important part in artificial intelligence for developing expert systems or formalizing knowledge bases. They also represent the base logic of Prolog.

The notation used in the literature for the SAT($\mathcal{F}$) problem when $\mathcal{F}$ is affine is LIN2. k-LIN2 is the kSAT($\mathcal{F}$) problem where $\mathcal{F}$ is affine and Ek-LIN2 is the variant of k-LIN2 where each equation depends on exactly k variables. An instance of LIN2 is *0-homogenous* (or *1-homogenous*) if all its linear equations have their free terms equal to 0 (or 1 respectively).

MONOTONE-SAT and MONOTONE-kSAT are the variants of SAT and kSAT where every clause contains only positive literals or contains only negative literals.

We will consider other variants of SAT($\mathcal{F}$) in this chapter:

– The NAE3SAT problem is SAT($\{f\}$), where f is of arity 3 and $f(x_1, x_2, x_3) = 1$ if and only if the three variables do not have the same value; more exactly, $f(0,0,0) = f(1,1,1) = 0$, otherwise f takes the value 1.

– The 1IN3SAT problem is SAT($\{g\}$), where g is of arity 3 and $g(x_1, x_2, x_3) = 1$ if and only if exactly one of the three variables has the value 1; more exactly, $g(1,0,0) = g(0,1,0) = g(0,0,1) = 1$, otherwise g takes the value 0.

COMMENT 1.2.– For certain variants of the SAT($\mathcal{F}$) problem, the set of constraints can be represented in an equivalent way by an expression that puts these constraints into conjunction. In the associated optimization problems, such as MAX SAT($\mathcal{F}$) and MIN SAT DELETION($\mathcal{F}$), we use only the expression in the form of a set of constraints in order to be able to count the number of satisfied constraints.

We now present a few variants of optimization problems used in the rest of the chapter:

– MAX SAT, given a set of disjunctive clauses defined on n Boolean variables, consists of finding an assignment that maximizes the number of satisfied clauses. MAX SAT therefore corresponds to the MAX SAT($\mathcal{F}$) problem where $\mathcal{F}$ is the set of OR_i^k functions, for $k \leqslant n$.

– MIN SAT DELETION, given a set of disjunctive clauses defined on n Boolean variables, consists of finding an assignment that minimizes the number of non-satisfied clauses. MIN SAT DELETION therefore corresponds to the MIN SAT DELETION($\mathcal{F}$) problem where $\mathcal{F}$ is the set of OR_i^k functions, for $k \leqslant n$.

– MAX CONJ, given a set of conjunctive clauses defined on n Boolean variables, consists of finding an assignment that maximizes the number of satisfied clauses. MAX CONJ therefore corresponds to the MAX SAT($\mathcal{F}$) problem where $\mathcal{F}$ is the set of AND_i^k functions, for $k \leqslant n$.

– MIN CONJ DELETION, given a set of conjunctive clauses defined on n Boolean variables, consists of finding an assignment that minimizes the number of non-satisfied clauses. MIN CONJ DELETION therefore corresponds to the MIN SAT DELETION ($\mathcal{F}$) problem where $\mathcal{F}$ is the set of AND_i^k functions, for $k \leqslant n$.

– MAX LIN2, given a set of linear equations defined on n Boolean variables, consists of finding an assignment that maximizes the number of satisfied equations. MAX LIN2 therefore corresponds to the MAX SAT($\mathcal{F}$) problem where $\mathcal{F}$ is the set of XOR^k, $XNOR^k$ functions, for $k \leqslant n$.

– MIN LIN2 DELETION, given a set of linear equations defined on n Boolean variables, consists of finding an assignment that minimizes the number of non-satisfied equations. MIN LIN2 DELETION therefore corresponds to the MIN SAT DELETION($\mathcal{F}$) problem where $\mathcal{F}$ is the set of XOR^k, $XNOR^k$ functions, for $k \leqslant n$;

– The problems MAX kSAT, MAX EkSAT, MAX kCONJ, MAXEkCONJ, MAX k-LIN2, and MAX Ek-LIN2, as well as the corresponding MIN DELETION versions, are defined in a similar way on clauses or equations of size (at most) k.

1.3. Complexity of decision problems

In this section we study the complexity of SAT($\mathcal{F}$) decision problems according to the type of functions of the set $\mathcal{F}$.

SAT was the first problem to be shown to be *NP*-complete by Cook [COO 71] and Levin [LEV 73]. We can easily reduce SAT to kSAT, $k \geqslant 3$, which implies the *NP*-completeness of kSAT, for $k \geqslant 3$. However, 2SAT is polynomial [COO 71].

THEOREM 1.1.– 2SAT *is solvable in polynomial time.*

Proof. Let I be an instance of 2SAT with m clauses $C_1, \ldots, C_m$ and n variables $x_1, \ldots, x_n$. We will construct a directed graph G_I with $2n$ vertices $v_1, \bar{v}_1, \ldots, v_n, \bar{v}_n$, where v_i (or $\bar{v}_i$ respectively) corresponds to x_i (or $\bar{x}_i$ respectively). For a literal ℓ_i (respectively $\bar{\ell}_i$), let us express by w_i (or $\bar{w}_i$ respectively) the corresponding vertex. In this way, if $\ell_i = x_i$ then $w_i = v_i$ and $\bar{w}_i = \bar{v}_i$, and if $\ell_i = \bar{x}_i$ then $w_i = \bar{v}_i$ and $\bar{w}_i = v_i$. Each clause made up of one single literal ℓ is replaced by the equivalent clause $\ell \vee \ell$. For each clause $\ell_1 \vee \ell_2$, that is equivalent to the logical implications

$\bar{\ell}_1 \Rightarrow \ell_2$ and $\bar{\ell}_2 \Rightarrow \ell_1$, let us introduce into G_I the arcs $(\bar{w}_1, w_2)$ and $(\bar{w}_2, w_1)$. Note that if there exists a path from w_i to w_j in G_I then there also exists a path from $\bar{w}_j$ to $\bar{w}_i$.

Let us consider an assignment of truth values for the vertices of G_I. This assignment corresponds to an assignment of $x_1, \ldots, x_n$ that satisfies I if and only if:

(a) each i, v_i and $\bar{v}_i$ have complementary values;

(b) no arc (w_p, w_q) is such that w_p has the value 1 and w_q has the value 0 (otherwise the logical implication $\ell_p \Rightarrow \ell_q$ would be false).

We will next justify that I is satisfiable if and only if in G_I no vertex v_i is in the same strongly connected component as $\bar{v}_i$.

Let us assume that I is satisfiable and that there exists a vertex v_i that belongs to the same strongly connected component as $\bar{v}_i$. Let there be an assignment for $x_1, \ldots, x_n$ that satisfies I. This assignment induces an assignment of truth values for the vertices of G_I that satisfy (a). Since v_i belongs to the same strongly connected component as $\bar{v}_i$, there exists in G_I a path from v_i to $\bar{v}_i$ and from $\bar{v}_i$ to v_i. One of these two paths obligatorily has as its initial extremity a vertex valued at 1 and as its terminal extremity a vertex valued at 0. It therefore contains an arc (w_p, w_q) such that w_p has the value 1 and w_q has the value 0, which contradicts (b) and therefore the fact that I is satisfiable.

Let us now assume that no vertex v_i is in the same strongly connected component as $\bar{v}_i$. We will construct an assignment over the vertices such that (a) and (b) are satisfied. Let us first determine the strongly connected components of G_I by using Tarjan's linear algorithm [TAR 72]. Let us then construct the reduced graph of G_I, expressed as G_I^r, whose vertices are the strongly connected components and where we create an arc from a component S_1 to a component S_2 if an arc from a vertex S_1 towards a vertex S_2 exists. Let us express as $\bar{S}_i$ the strongly connected component that contains the complementary literals to the literals of S_i. Obviously, if S_1 is a predecessor of S_2 then $\bar{S}_2$ is a predecessor of $\bar{S}_1$. Tarjan's algorithm generates the strongly connected components in inverse topological order; more exactly, if S_1 is generated before S_2 then S_1 cannot be a predecessor of S_2.

We will now define the truth values for the vertices of G_I^r; a vertex of G_I will then have the truth value of the component to which it belongs. We repeat the following algorithm as long as is possible: let us consider the first component S in the inverse topological order which does not have a truth value, and let us assign the value 1 to S and the value 0 to the component $\bar{S}$. Obviously (a) is satisfied. To justify that (b) is satisfied, we must show that no arc from a vertex that corresponds with a literal of value 1 towards a vertex that corresponds with a literal of value 0 exists. Assume there exists an arc from a vertex w_1 of value 1 that belongs to the component S_1

towards a vertex w_2 of value 0 that belongs to the component S_2. Then in G_I^r there exists an arc from S_1 (valued at 1) to S_2 (valued at 0) and from $\bar{S}_2$ (valued at 1) to $\bar{S}_1$ (value at 0). This contradicts the way in which we have assigned the values 1 to the components because in an inverse topological order S_2 is before S_1 and $\bar{S}_1$ is before $\bar{S}_2$, and therefore at least one of the components S_2 or $\bar{S}_1$ should have the value 1. ∎

Testing the satisfiability of a Horn expression has been shown to be polynomial by Jones and Laaser [JON 77], and the complexity of the polynomial algorithm has been improved by Dowling and Gallier [DOW 84], and Minoux [MIN 88].

THEOREM 1.2.– HORN SAT *is solvable in polynomial time.*

Proof. Let us consider an instance I of HORN SAT. If I does not contain a unit clause, each clause contains at least one negative literal and we only need to set all the variables to 0 to obtain a satisfying assignment. If I contains at least one unit clause, we use the unit solution principle which consists of iteratively applying the following two rules:

1) If a clause is made up of one positive literal x_i (or one negative literal $\bar{x}_i$) then set x_i to 1 (or to 0) and remove the clause.

2) While there exists a clause that contains at least one fixed variable then the expression can be reduced in this way:

(a) Remove every clause that contains a positive literal x_i where x_i has been set to 1 (or a negative literal $\bar{x}_i$ where x_i has been set to 0) because this clause will automatically be satisfied independently of the values of the other literals of the clause.

(b) In every clause, remove every positive literal x_i such that x_i has been set to 0 (or every negative literal $\bar{x}_i$ such that x_i has been set to 1) because such a literal will never satisfy this clause.

If by applying (b) we remove all the literals of a clause then the expression is not satisfiable.

After having applied rules 1 and 2, there are three possible cases:

– I is declared non-satisfiable in 2(b).

– I is satisfiable because all its clauses have been removed by applying 1 and 2(a).

– The partial assignment obtained defines a subinstance I' that does not contain a unit clause. I is therefore satisfiable by setting to 0 the variables that have not been set by the partial assignment.

∎

Obviously a similar algorithm to the previous one can be established to decide whether SAT($\mathcal{F}$) is satisfiable when $\mathcal{F}$ is anti-Horn and in the positive case, to find a satisfying assignment. Each of these two algorithms also works when $\mathcal{F}$ is 2-monotone.

When $\mathcal{F}$ is affine, SAT($\mathcal{F}$) is also solvable in polynomial time using Gaussian elimination.

THEOREM 1.3.– LIN2 *is solvable in polynomial time.*

Therefore SAT($\mathcal{F}$) is solvable in polynomial time when each function of $\mathcal{F}$ is a disjunctive clause of size two at most (or more generally when each function of $\mathcal{F}$ is 2CNF), when $\mathcal{F}$ is Horn or anti-Horn and when $\mathcal{F}$ is affine. Do other specific cases exist for which SAT($\mathcal{F}$) is solvable in polynomial time? Schaefer [SCH 78] established a characterization of the complexity of decision problems according to the type of constraints which shows that the only cases where SAT($\mathcal{F}$) is solvable in polynomial time are the previous cases as well as the trivial case where $\mathcal{F}$ is 0 or 1-valid. In this last case, one of the two trivial assignments (the assignment of 0 for each variable or the assignment of 1 for each variable) is a feasible solution. For example, MONOTONE-SAT is solvable in polynomial time because it falls into this last case.

THEOREM 1.4.– *[Dichotomic theorem for* SAT($\mathcal{F}$) *[SCH 78]] Given an $\mathcal{F}$-set of constraints, the* SAT($\mathcal{F}$) *problem is in P if $\mathcal{F}$ satisfies one of the following conditions, and* SAT($\mathcal{F}$) *is NP-complete otherwise:*

– *$\mathcal{F}$ is 0-valid (1-valid);*

– *$\mathcal{F}$ is Horn (anti-Horn);*

– *$\mathcal{F}$ is affine;*

– *$\mathcal{F}$ is bijunctive.*

1.4. Complexity and approximation of optimization problems

In this section, we first present a polynomial algorithm for solving MAX SAT($\mathcal{F}$) when $\mathcal{F}$ is 2-monotone. Next, we highlight a few classical methods that allow us to establish positive approximation results for MAX SAT($\mathcal{F}$). We also cite other positive and negative results that are found in the literature for MAX SAT($\mathcal{F}$) and MIN SAT DELETION($\mathcal{F}$).

1.4.1. *Maximization problems*

If a SAT($\mathcal{F}$) problem is *NP*-hard then the corresponding MAX SAT($\mathcal{F}$) problem is also *NP*-hard. However, maximization problems exist that become hard even if the corresponding decision problems are easy. Thus, MAX 2SAT is *NP*-hard [GAR 74], MAX HORN SAT is *NP*-hard [KOH 94] even if 2SAT and HORN SAT allow polynomial algorithms. Nevertheless, in certain cases, MAX SAT($\mathcal{F}$) is polynomial. A first trivial case is that where $\mathcal{F}$ is 0 or 1-valid, all the constraints then being satisfied.

We have seen in the previous section that SAT($\mathcal{F}$) is polynomial when $\mathcal{F}$ is 2-monotone (using the algorithm for $\mathcal{F}$ Horn or anti-Horn). In fact, we can establish a stronger result that allows us to establish an assignment that maximizes the number of satisfied constraints in polynomial time.

THEOREM 1.5.– *[Creignou [CRE 95a], Khanna, Sudan, Williamson [KHA 97b]]* MAX SAT($\mathcal{F}$) *is polynomial when each function of* $\mathcal{F}$ *is a 2-monotone function.*

Proof. We consider the equivalent problem MIN SAT DELETION($\mathcal{F}$), which we reduce to the s-t MIN CUT problem in a directed graph. Let us consider an instance I of the MAX SAT($\mathcal{F}$) problem over n variables with m constraints, each function of $\mathcal{F}$ being a 2-monotone function of type:

1) $x_{i_1} \wedge \ldots \wedge x_{i_p}$;

2) $\bar{x}_{\ell_1} \wedge \ldots \wedge \bar{x}_{\ell_q}$;

3) $(x_{i_1} \wedge \ldots \wedge x_{i_p}) \vee (\bar{x}_{\ell_1} \wedge \ldots \wedge \bar{x}_{\ell_q})$.

We construct a directed graph $G_I = (V, A)$, where V contains two special vertices F, T, a vertex x_i for each of the n variables x_i and a vertex v_j for a constraint C_j of type 1, a vertex $\bar{v}_j$ for a constraint C_j of type 2, and two vertices v_j and $\bar{v}_j$ for a constraint C_j of type 3. To construct the set of arcs, we proceed as follows:

– For a constraint C_j of type 1, we create an arc of cost ∞ from x_{i_k} to v_j for $k = 1, \ldots, p$, and an arc of cost 1 from v_j to T.

– For a constraint C_j of type 2, we create an arc of cost ∞ from $\bar{v}_j$ to x_{ℓ_k} for $k = 1, \ldots, q$, and an arc of cost 1 from F to $\bar{v}_j$.

– For a constraint C_j of type 3, we create an arc of cost ∞ from x_{i_k} to v_j for $k = 1, \ldots, p$, an arc of cost ∞ from $\bar{v}_j$ to x_{ℓ_k} for $k = 1, \ldots, q$, and an arc of cost 1 from v_j to $\bar{v}_j$.

We now justify that the value of a minimal cut from F to T corresponds to an assignment with a minimum number of non-satisfied constraints. Let us remember that the value of a cut created by a partition (A, B) with $F \in A$ and $T \in B$ is the sum of the costs of the arcs whose initial extremities belong to A and terminal extremities belong to B.

Given a cut C^* of minimal value from F to T, let us consider the assignment that assigns 0 (or 1 respectively) to the variables that are in the same part as F (or T respectively). If an arc of cost 1 from v_j to T, which corresponds to a constraint of type 1, is part of the cut C^* then at least one of the variables $x_{i_1}, \ldots, x_{i_p}$ is set to 0 because otherwise the vertices that correspond to these variables are all in the same part as T in the cut C^*, and so by putting v_j on the side T of the cut, we would obtain a cut of lower value than the value of the cut C^*, which contradicts the fact that C^* is a cut of minimal value. Thus the constraint C_j is not satisfied. In the same way,

we can justify that if an arc of cost 1 from F to $\bar{v}_j$, which corresponds to a constraint of type 2, is part of the cut C^* then the corresponding constraint C_j is not satisfied. Furthermore, if an arc of cost 1 from v_j to $\bar{v}_j$, which corresponds to a constraint of type 3, is part of the cut C^* then at least one of the variables $x_{i_1}, \ldots, x_{i_p}$ is set to 0 and at least one of the variables $x_{\ell_1}, \ldots, x_{\ell_q}$ is set to 1, and therefore the corresponding constraint C_j is not satisfied.

Let us now consider an assignment for $x_1, \ldots, x_n$ that minimizes the number of non-satisfied constraints. The value of the following cut is equal to the number of constraints not satisfied by the previous assignment:

– Place the vertices that correspond to the variables set to 0 (or 1 respectively) in this assignment in the same part as F (or T respectively).

– Place the vertex v_j that corresponds to a constraint C_j of type 1 in the part of T (or F) if C_j is satisfied (or non-satisfied).

– Place the vertex $\bar{v}_j$ that corresponds to a constraint C_j of type 2 in the part of F (or T) if C_j is satisfied (or non-satisfied).

– If C_j is a constraint of type 3, if $x_{i_1} \wedge \ldots \wedge x_{i_p}$ is satisfied, put v_j in the part of T otherwise in the part of F, and if $\bar{x}_{\ell_1} \wedge \ldots \wedge \bar{x}_{\ell_q}$ is satisfied, put $\bar{v}_j$ in the part of F, otherwise in the part of T.

∎

Thus, MAX SAT($\mathcal{F}$) is solvable in polynomial time when each function of $\mathcal{F}$ is a 0-valid, 1-valid or 2-monotone function. The theorem of classification for MAX SAT($\mathcal{F}$) establishes that the previous cases are the only cases for which the problem is easy.

THEOREM 1.6.– *Theorem of classification for* MAX SAT($\mathcal{F}$) *[CRE 95a, KHA 97b]]* MAX SAT($\mathcal{F}$) *is in P if $\mathcal{F}$ is 0-valid or 1-valid or 2-monotone and* MAX SAT($\mathcal{F}$) *is APX-complete otherwise.*

In what follows, we will establish a few approximation algorithms for a hard problem, MAX SAT. A first, very simple approximation algorithm has been proposed by Johnson [JOH 74].

THEOREM 1.7.– *[JOH 74]* MAX SAT *is approximable up to a factor of* $\frac{1}{2}$.

Proof. Let us consider an instance with m clauses $C_1, \ldots, C_m$ over n variables $x_1, \ldots, x_n$, whose optimal value is expressed as opt. The algorithm consists of considering, for each variable x_i, $x_i = 1$ with the probability $\frac{1}{2}$ and $x_i = 0$ with the probability $\frac{1}{2}$. This algorithm provides an approximation up to a factor of $\frac{1}{2}$. Let W

be the random variable that represents the number of satisfied clauses, so the expectation of this random variable is:

$$E(W) = \sum_{j=1}^{m} P(C_j \text{ is satisfied}) = \sum_{j=1}^{m}(1 - \frac{1}{2^{|C_j|}}) \geqslant \frac{m}{2} \geqslant \frac{opt}{2}$$

By using the conditional expectation method proposed by Erdös and Selfridge [ERD 73], we can transform this algorithm into a deterministic algorithm with the same performance guarantee as follows.

We will set values to the variables in the order $x_1, \ldots, x_n$. Let us assume that we have set the values $b_1, \ldots, b_i$ to the variables $x_1, \ldots, x_i$. Let us calculate $E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0)$ and $E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1)$, and let $x_{i+1} = 0$ if $E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0) \geqslant E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1)$, and $x_{i+1} = 1$ otherwise. Since

$$E(W|x_1 = b_1, \ldots, x_i = b_i) = \frac{1}{2}E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1) +$$

$$\frac{1}{2}E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0)$$

then

$$\max\{E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 1), E(W|x_1 = b_1, \ldots, x_i = b_i, x_{i+1} = 0)\}$$

$$\geqslant E(W|x_1 = b_1, \ldots, x_i = b_i)$$

The assignment found at the end $x_1 = b_1, \ldots, x_n = b_n$ has the value equal to

$$E(W|x_1 = b_1, \ldots, x_n = b_n) \geqslant E(W) \geqslant \frac{opt}{2}.$$

∎

Using the random rounding method, Goemans and Williamson [GOE 94] have improved the previous result.

THEOREM 1.8.– *[GOE 94]* MAX SAT *is approximable up to a factor of* $1 - \frac{1}{e} \approx 0.632$.

Proof. Let I be an instance of MAX SAT with m clauses $C_1, \ldots, C_m$ over n variables $x_1, \ldots, x_n$. The algorithm is the following:

1) Formulate MAX SAT as a linear program in 0–1 variables. With each Boolean variable x_i we associate a 0–1 variable y_i, and with each clause C_j a variable z_j such

that z_j will take the value 1 if and only if C_j is satisfied. Let $C_j^+ = \{i : x_i \in C_j\}$ and $C_j^- = \{i : \bar{x}_i \in C_j\}$. Now the linear program associated with MAX SAT is:

$$(Sat)\begin{cases} \max \sum_{j=1}^m z_j \\ \sum_{i\in C_j^+} y_i + \sum_{i\in C_j^-}(1-y_i) \geqslant z_j \qquad (j=1,\ldots,m) \\ y_i \in \{0,1\} \quad (i=1,\ldots,n), \qquad z_j \in \{0,1\} \quad (j=1,\ldots,m) \end{cases}$$

2) Solve the relaxed problem (P):

$$(P)\begin{cases} \max \sum_{j=1}^m z_j \\ \sum_{i\in C_j^+} y_i + \sum_{i\in C_j^-}(1-y_i) \geqslant z_j \qquad (j=1,\ldots,m) \\ y_i \in [0,1] \quad (i=1,\ldots,n), \qquad z_j \in [0,1] \quad (j=1,\ldots,m) \end{cases}$$

Let (y^*, z^*) be the optimal solution found.

3) Let us consider the assignment: $x_i = 1$ with the probability y_i^* and $x_i = 0$ with the probability $1 - y_i^*$.

Let W be the random variable that represents the number of satisfied clauses. So the expectation of this random variable is

$$E(W) = \sum_{j=1}^m P(C_j \text{ is satisfied}) = \sum_{j=1}^m (1 - \Pi_{i\in C_j^+}(1-y_i^*)\Pi_{i\in C_j^-} y_i^*)$$

We will show that $1 - \Pi_{i\in C_j^+}(1-y_i^*)\Pi_{i\in C_j^-} y_i^* \geqslant (1-\frac{1}{e})z_j^*$.

For this, let us first show that for every solution (y, z) of (P) and every clause C_j with k literals, we have

$$1 - \Pi_{i\in C_j^+}(1-y_i)\Pi_{i\in C_j^-} y_i \geqslant c_k z_j$$

where $c_k = 1 - (1-\frac{1}{k})^k$.

In (*Sat*), the inequality that corresponds to C_j is

$$\sum_{i\in C_j^+} y_i + \sum_{i\in C_j^-}(1-y_i) \geqslant z_j \iff$$

$$|C_j^+| + |C_j^-| - \sum_{i\in C_j^+} y_i - \sum_{i\in C_j^-}(1-y_i) \leqslant k - z_j \iff$$

$$\sum_{i\in C_j^+}(1-y_i) + \sum_{i\in C_j^-} y_i \leqslant k - z_j$$

Knowing that we have the classical inequality $\frac{a_1+\ldots+a_k}{k} \geqslant \sqrt[k]{a_1 \ldots a_n}, \forall a_1, \ldots, a_k \geqslant 0$, we have

$$1 - \Pi_{i \in C_j^+}(1 - y_i)\Pi_{i \in C_j^-} y_i \geqslant 1 - \left(\frac{\sum_{i \in C_j^+}(1 - y_i) + \sum_{i \in C_j^-} y_i}{k} \right)^k$$

$$\geqslant 1 - \left(\frac{k - z_j}{k} \right)^k 1 - \left(1 - \frac{z_j}{k} \right)^k$$

Let us consider the function $f(x) = 1 - (1 - \frac{x}{k})^k$. We can easily verify that f is concave, $f(0) = 0$, and $f(1) = 1 - (1 - \frac{1}{k})^k = c_k$. Knowing that f is concave, then to show that $f(x) \geqslant ax + b$, for $x \in [u, v]$, all we need to do is show that $f(u) \geqslant au + b$ and $f(v) \geqslant av + b$; we deduce from this that $f(x) \geqslant c_k x$, for $x \in [0, 1]$.

Thus

$$1 - \Pi_{i \in C_j^+}(1 - y_i)\Pi_{i \in C_j^-} y_i \geqslant 1 - \left(1 - \frac{z_j}{k} \right)^k \geqslant c_k z_j$$

Since $c_1 (= 1) > c_2 (= \frac{3}{4}) > \ldots > c_k > \ldots > 1 - \frac{1}{e}$, we have

$$1 - \Pi_{i \in C_j^+}(1 - y_i^*)\Pi_{i \in C_j^-} y_i^* \geqslant (1 - \frac{1}{e}) z_j^*$$

for every $j = 1, \ldots, m$.

Thus, in the end we obtain

$$E(W) \geqslant \sum_{j=1}^{m} (1 - \frac{1}{e}) z_j^* = (1 - \frac{1}{e}) opt_P \geqslant (1 - \frac{1}{e}) opt_{Sat}$$

By using the conditional expectation method proposed by Erdös and Selfridge [ERD 73], we can transform this algorithm into a deterministic algorithm with the same performance guarantee as in theorem 1.7. ∎

Goemans and Williamson [GOE 94] then improved the previous algorithm for MAX SAT.

THEOREM 1.9.– *[GOE 94]* MAX SAT *is approximable up to a factor of* $\frac{3}{4}$.

Proof. The algorithm consists of assigning the value 0 to a bit b with the probability $\frac{1}{2}$, and the value 1 with the probability $\frac{1}{2}$. If $b = 0$, we apply Johnson's algorithm, and if $b = 1$, we apply the previous random rounding algorithm.

For a clause C_j of size k, let W_j be the random variable that indicates whether the clause is satisfied:

$$E(W_j) = \frac{1}{2}[E(W_j|b=0) + E(W_j|b=1)]$$

$$E(W_j|b=0) = 1 - \frac{1}{2^k} \geqslant (1 - \frac{1}{2^k}) z_j^*$$

$$E(W_j|b=1) = 1 - (1 - \frac{1}{k})^k \geqslant (1 - (1 - \frac{1}{k})^k) z_j^*$$

Therefore, $E(W_j) \geqslant \frac{3}{4} z_j^*$ and $E(W) = \sum_{i=1}^{m} E(W_j) \geqslant \frac{3}{4} opt_{Sat}$.

Using the conditional expectation method allows us to find a deterministic algorithm with the same performance guarantee. ∎

The previous result is not the best known in the literature concerning the approximation of MAX SAT. Asano and Williamson [ASA 00] established an approximation algorithm with a factor of up to 0.784 for MAX SAT. Johnson's algorithm for MAX SAT [JOH 74] also establishes an approximation with a factor of up to $\frac{2^k-1}{2^k}$ for MAX EkSAT, $k \geqslant 2$. Another method that allows us to obtain better approximation results for MAX SAT and its variants consists of modeling the problem as a semi-defined program and using random rounding [GOE 95]. In this way, following this method for the MAX 2SAT version, Feige and Goemans [FEI 95] obtained the best approximation algorithm that gives an approximation of 0.931. On the negative side, Papadimitriou and Yannakakis [PAP 88] have shown that MAX kSAT, $k \geqslant 2$ is *MAX SNP*-hard, which means that it does not have a polynomial time approximation scheme. Håstad [HÅS 97] later showed that even the MAX EkSAT, $k \geqslant 3$, version is not approximable up to a factor of $(\frac{2^k-1}{2^k} - \varepsilon)$, for every $\varepsilon > 0$, and that MAX E2SAT is not approximable up to a factor of $(\frac{21}{22} - \varepsilon)$, for every $\varepsilon > 0$, if *P≠NP*.

By also using the relaxation from integer programming and random rounding, Trevisan [TRE 96] showed that MAX kCONJ, $k \geqslant 2$, is approximable up to a factor of $\frac{1}{2^{k-1}}$. MAX CONJ is as hard to approximate as MAX INDEPENDENT SET [CRE 96], that is it is not approximable up to a factor of $\frac{1}{m^{1-\varepsilon}}$, for every $\varepsilon > 0$, if *NP≠ ZPP*, where m is the number of constraints.

Johnson's algorithm for MAX SAT [JOH 74] can also be applied to MAX LIN2 and MAX kLIN2, $k \geqslant 2$, and provides an approximation up to a factor of $\frac{1}{2}$. Håstad [HÅS 97] showed that even the MAX E3LIN version is not approximable up to a factor of $(\frac{1}{2} - \varepsilon)$, for every $\varepsilon > 0$, and that MAX E2LIN is not approximable up to a factor of $(\frac{11}{12} - \varepsilon)$, for every $\varepsilon > 0$, if *P≠NP*.

1.4.2. *Minimization problems*

Let us consider the MIN SAT DELETION($\mathcal{F}$) problem. Taking into account the equivalence of this problem to MAX SAT($\mathcal{F}$) from the exact complexity point of view, the polynomial cases for MIN SAT DELETION($\mathcal{F}$) are exactly the same as for MAX SAT($\mathcal{F}$), that is when $\mathcal{F}$ is 0-valid, 1-valid and 2-monotone.

Let us now consider approximation complexity. A classification theorem has also been established for MIN SAT DELETION($\mathcal{F}$) by Khanna, Sudan and Trevisan [KHA 97a]. This theorem is much more complex than the classification theorems for SAT($\mathcal{F}$) and MAX SAT($\mathcal{F}$).

Klauck [KLA 96] showed that MIN 3SAT DELETION is not approximable up to a factor of $n^{1-\varepsilon}$, for every $\varepsilon > 0$, if *P*≠ *NP*, where n is the number of variables. However, MIN 2SAT DELETION is approximable up to a factor of $O(\log n \log\log n)$ [KLE 97] and it does not have an approximation scheme.

MIN kCONJ DELETION, $k \geqslant 2$, is approximable up to a factor of $2(1 - \frac{1}{2^k})$ [BER 96], and *MAX SNP*-hard [KOH 94] and therefore it does not have an approximation scheme. MIN 2CONJ DELETION is approximable up to a factor of 1.103 and it is not approximable up to a factor of $\frac{7}{6} - \varepsilon$, for every $\varepsilon > 0$, if *P*≠ *NP* and MIN 3CONJ DELETION is approximable up to a factor of 1.213 and it is not approximable up to a factor of $\frac{15}{14} - \varepsilon$, for every $\varepsilon > 0$, if *P*≠ *NP* [AVI 02]. MIN CONJ DELETION is as hard to approximate as MIN VERTEX COVER [CRE 96], that is it is approximable up to a factor of 2 and it does not have an approximation scheme.

The MIN E2-LIN2 DELETION problem has been shown to be *MAX SNP*-hard in [GAR 93] and therefore does not allow a polynomial time approximation scheme. On the other hand, it is approximable up to a factor of $O(\log n)$ [GAR 93]. The MIN Ek-LIN2 DELETION problems are extremely hard to approximate for every $k \geq 3$. In fact, they are not approximable in polynomial time up to a factor of $n^{\Omega(1)/\log\log n}$, unless *P* = *NP* [DIN 98]. A first polynomial algorithm with a sublinear approximation factor, $O(n/\log n)$, has been established for the general problem MIN Ek-LIN2 DELETION [BER 02].

1.5. Particular instances of constraint satisfaction problems

Certain optimization problems become easier to approximate when we restrict ourselves to particular instances. In this part, we will study various types of particular instances of optimization problems: planar, dense instances, and with a bounded number of occurrences of each variable.

1.5.1. *Planar instances*

We generally talk about planar instances of a problem when the problem is defined on a graph. In the case of satisfiability problems, a natural manner of associating a graph with such a problem exists.

DEFINITION 1.4.– *Given an instance I of a Boolean constraint satisfaction problem, m constraints $C_1, \ldots, C_m$ defined over n Boolean variables $x_1, \ldots, x_n$, the associated graph $G_I = (V, E)$ is a bipartite graph defined in this way:*

– $V = \{x_1, \ldots, x_n\} \cup \{C_1, \ldots, C_m\}$, where x_i is the vertex associated with the variable x_i, and C_j is the vertex associated with the constraint C_j.

– $E = \{(x_i, C_j) : x_i$ appears in $C_j\}$.

DEFINITION 1.5.– *An instance of a satisfiability problem is planar if the associated graph is planar.* PLANAR A *is the* A *problem reduced to planar instances, where* A *is a decision or optimization problem.*

The complexity of planar instances has been studied for a long time. For example, Lichtenstein showed in [LIC 82] that PLANAR 3SAT remains *NP*-hard, and Dyer and Frieze [DYE 86] showed that PLANAR 1IN3SAT remains *NP*-hard. More generally, we can show [CRE 01] that for each $\mathcal{F}$-set of constraints, if SAT($\mathcal{F}$) is *NP*-complete, then PLANAR SAT($\mathcal{F} \cup \{F, T\}$) is also *NP*-complete. Furthermore, if the set $\mathcal{F}$ is not complement-closed then PLANAR SAT($\mathcal{F}$) is *NP*-complete when SAT($\mathcal{F}$) is *NP*-complete. An example of a SAT($\mathcal{F}$) problem where $\mathcal{F}$ is complement-closed is NAE3SAT. Kratochvil and Tuza [KRA 00] have shown that PLANAR NAE3SAT is polynomial while NAE3SAT is *NP*-hard.

As for approximation complexity, Hunt *et al.* [HUN 94] gave a polynomial time approximation scheme for the PLANAR MAX kSAT($\mathcal{F}$) problem for every set $\mathcal{F}$, which means, for example, that PLANAR MAX 3SAT has an approximation scheme. Khanna and Motwani [KHA 96] have generalized the previous result by showing that PLANAR MAX SAT and more generally PLANAR MAX SAT($\mathcal{F}$) have an approximation scheme.

Before explaining the idea of this last scheme, let us define the idea of a t-exterior planar graph.

DEFINITION 1.6.– *A 1-exterior planar graph is a planar graph that allows a representation in the plane where all the vertices appear on the exterior face. A t-exterior planar graph is a planar graph that has a representation in the plane such that by removing the vertices on the exterior face we obtain a $(t-1)$-exterior planar graph.*

THEOREM 1.10.– *[KHA 96]* PLANAR MAX SAT($\mathcal{F}$) *has an approximation scheme.*

Proof. Let I be an instance of PLANAR MAX SAT($\mathcal{F}$) with n variables and m constraints, and let $G_I = (V, E)$ be the graph associated with I. Since $|V| = n + m$, the graph G_I is t-exterior planar graph where $t \leqslant n + m$. Let $L_1, \ldots, L_t$ be the sets of vertices such that L_t corresponds with the exterior face and each L_i is the exterior face obtained by removing the vertices of the sets $L_t, \ldots, L_{i+1}$.

Let us consider an optimal assignment for I and let n_i be the number of satisfied constraints that correspond with the vertices that belong to L_i. We partition the faces $L_1, \ldots, L_t$ into $p+1$ groups $S_0, \ldots, S_p$ (where p will be determined according to the maximal error ε with which we wish to find a solution), where each group S_r is the union of the faces L_i where i is equal to $3r$, $3r+1$ or $3r+2$ modulo q and $q = 3(p+1)$. By using the pigeonhole principle, we can deduce that there exists a group S_j such that $\sum_{L_i \in S_j} n_i \leqslant \frac{opt(I)}{p+1}$. This group will be determined by trying all the possibilities and choosing the best solution from them. When we choose S_j, we remove the vertices of the faces with an index equal to $3j + 1$ modulo q, which separates the graph in this way into a family of disjoint $(q-1)$-exterior planar graphs, $G_1, G_2, \ldots, G_\ell$, such that the total sum of the corresponding n_i is at least $(1 - \frac{1}{p+1})opt(I)$. A k-exterior planar graph has a treewidth of at most $3k - 1$ ([BOD 98]). By using dynamic programming, we can establish a polynomial algorithm that provides an optimal solution for graphs with a bounded treewidth, in particular for the graphs $G_1, G_2, \ldots, G_\ell$. Since the sum of the values of the optimal solutions obtained for each G_t will be at least equal to the total sum of the corresponding n_i, when we choose $p = \lceil \frac{1}{\varepsilon} - 1 \rceil$, we obtain an approximation of a factor of $(1 - \varepsilon)$. ∎

1.5.2. *Dense instances*

Two types of dense instances exist that are studied in the literature: everywhere dense and average dense instances.

DEFINITION 1.7.– *An instance of a* MAX kSAT($\mathcal{F}$) *or* MIN kSAT DELETION($\mathcal{F}$) *problem over n variables is* everywhere α-dense *if, for each variable, the total number of occurrences of the variable and of its negation is at least αn^{k-1}, and it is* average α-dense *if the number of constraints is of at least αn^k.*

DEFINITION 1.8.– *A set of instances is* everywhere dense *if there exists a constant $\alpha > 0$ such that each instance is everywhere α-dense, and a set of instances is* average dense *if there exists a constant $\alpha > 0$ such that each instance is average α-dense.*

Therefore, a set of everywhere dense instances is average dense but the opposite is not true.

Arora, Karger and Karpinski [ARO 95] started the systematic study of the approximation complexity of dense instances of optimization problems. They have shown that average dense (and everywhere dense) instances of MAX kSAT, MAX CUT, MAX DICUT, DENSE kSUBGRAPH and more generally of every MAX kSAT($\mathcal{F}$) problem have a polynomial time approximation scheme. Arora, Karger and Karpinski observed that the optima of average dense instances of MAX kSAT($\mathcal{F}$) problems are "large" ($\Omega(n^k)$, where n is the number of variables) and that, in this case, an additive approximation means a relative approximation. The basic idea is to represent the problems as mathematical programs in integer numbers of a certain type [ARO 95], then to apply general approximation results for these programs to obtain an additive approximation.

Dense instances of minimization problems have also been studied. In [ARO 95], Arora, Karger and Karpinski established polynomial time approximation schemes for everywhere dense instances of the following minimization problems: MIN BISECTION and MIN kCUT. For these latter problems, they used supplementary ideas in relation to maximization problems because the values of the optimal solutions of dense instances of minimization problems can be close to zero and in this case an additive approximation does not necessarily provide a relative approximation.

Bazgan and Fernandez de la Vega [BAZ 99] initiated the systematic study of dense instances of the minimization versions of satisfiability problems with the MIN E2-LIN2 DELETION problem. More exactly, they showed [BAZ 99] that the everywhere dense instances of MIN E2-LIN2 DELETION have a polynomial time approximation scheme. In [BAZ 02, BAZ 03] Bazgan, Fernandez de la Vega and Karpinski have generalized the result obtained for MIN E2-LIN2 DELETION to the two problems, MIN kCONJ DELETION, $k \geqslant 2$, and MIN Ek-LIN2 DELETION, $k \geqslant 3$, that belong to MIN kSAT DELETION($\mathcal{F}$).

The polynomial time approximation scheme for the everywhere dense instances of these MIN kSAT DELETION($\mathcal{F}$) problems is made up of two algorithms (as in [ARO 95] for MIN BISECTION). The first guarantees a good solution when the optimum of the problem is $\Omega(n^k)$; the second guarantees a good solution when the optimum of the problem is $O(n^k)$. When the optimum is large, the idea consists of expressing the problem as an integer program of a certain type, then using the method from [ARO 95] that provides a solution with an additive error in the order of $O(n^k)$. When the optimum is small, the idea of the algorithm is to make an exhaustive sampling in a graph or hypergraph and to take as the solution the best one obtained by "completing" each possibility of placing the variables. The algorithm obtained is a random algorithm that can be derandomized as in [ARO 95].

Certain optimization problems do not allow a polynomial time approximation scheme on everywhere dense instances. An example of such a problem is MIN 2SAT DELETION. In fact, we can render the instances of MIN 2SAT DELETION everywhere dense, without changing the value of the optimum, by adding disjoint copies of the

original variables, then by adding all the conjunctions that have exactly one original variable and one copy. Since MIN 2SAT DELETION does not have a polynomial time approximation scheme, the everywhere dense instances of MIN 2SAT DELETION do not have a polynomial time approximation scheme.

Let us emphasize that the average dense instances of MIN kCONJ DELETION and MIN Ek-LIN2 DELETION, $k \geqslant 2$, are as hard to approximate as the general instances of these problems [BAZ 03]. The idea is to construct a reduction of the general case to the particular case by doubling the number of variables and by considering all the clauses or equations over exactly k variables.

In conclusion, the MAX kSAT($\mathcal{F}$) problems have an approximation scheme for the everywhere dense instances as well as for the average dense instances; however, most of the MIN kSAT DELETION($\mathcal{F}$) problems that have an approximation scheme for the everywhere dense instances remain as hard to approximate for the average dense instances as for the general instances.

1.5.3. *Instances with a bounded number of occurrences*

Certain decision problems remain *NP*-complete even in the case where each variable only appears a bounded number of times.

Let us express by EtOCC-EkSAT the variant of EkSAT where each clause contains exactly k literals and each variable appears exactly t times, positively or negatively.

THEOREM 1.11.– 3SAT *remains NP-complete even if each variable appears at most three times, at least once positively and at least once negatively.*

Proof. The idea is to reduce 3SAT to this particular case by replacing a variable x that appears $k \geqslant 3$ times with k copies $x_1, \ldots, x_k$, and to make sure that these k copies take the same truth value by adding the clauses $(\bar{x}_1 \vee x_2) \wedge (\bar{x}_2 \vee x_3) \wedge \ldots \wedge (\bar{x}_k \vee x_1)$. ∎

In the previous theorem, it is important that each variable appears at most (and not exactly) three times, and each clause has at most (and not exactly) three literals because, if not, the problem becomes polynomial.

THEOREM 1.12.– EkOCC-EkSAT, $k \geqslant 2$, *is polynomial [[PAP 94], Problem 9.5.4 (b)].*

Proof. Let I be an instance of E3OCC-E3SAT with n variables $x_1, \ldots, x_n$ and n clauses $C_1, \ldots, C_n$. We construct a bipartite graph $G = (V_1, V_2, E)$, where $V_1 = \{x_1, \ldots, x_n\}$, $V_2 = \{C_1, \ldots, C_n\}$, and we create an edge between x_i and C_j if and only if the clause C_j contains x_i or $\bar{x}_i$. The k-regular bipartite graph constructed

in this way contains a perfect coupling $M = \{(x_{i_1}, C_{j_1}), \ldots, (x_{i_n}, C_{j_n})\}$ (from the König–Hall theorem [HAL 34]) that can be found by using, for example, the Ford–Fulkerson algorithm. The following assignment obtained from M satisfies I: consider $x_{i_\ell} = 1$ if C_{j_ℓ} contains x_{i_ℓ} and $x_{i_\ell} = 0$ if C_{j_ℓ} contains $\bar{x}_{i_\ell}$, for $\ell = 1, \ldots, n$. ∎

Tovey [TOV 84] showed that E4OCC-E3SAT is *NP*-hard and MAX E4OCC-E3SAT is APX-hard. Berman, Karpinski and Scott [BER 03c] showed that these results remain true even for the variants of these problems where each variable appears exactly twice positively and twice negatively. Dubois [DUB 90] showed the *NP*-hardness of E6OCC-E4SAT and E11OCC-E5SAT. In [KRA 93], Kratochvil, Savicky and Tuza defined the function $f(k)$ as being the largest t such that every instance of EtOCC-EkSAT is always satisfiable and they showed that if $t > f(k)$ then EtOCC-EkSAT is *NP*-hard. Furthermore, $f(k+1) \leqslant 2f(k) + 1$ and $f(k) \geqslant \lfloor \frac{2^k}{ek} \rfloor$. Berman, Karpinski and Scott [BER 03b] showed that if $t > f(k)$ then MAX EtOCC-EkSAT is *APX*-hard, and they also improved certain bounds for the function f. More exactly, $f(5) < 9$ and $f(6) \geqslant 7$. In [KAR 01, BER 03a, BER 03b, BER 03c] we can find certain lower and upper approximation bounds for various problems, for example for MAX 3OCC-E2SAT, MAX 3OCC-E2-LIN2 and MIN 3OCC-E3-LIN2 DELETION.

1.6. Satisfiability problems under global constraints

Global constraints naturally appear in certain optimization problems. For example, MIN BISECTION is the MIN CUT problem under the constraint that the two parts separated by the cut must be of equal size. It is known that MIN CUT is polynomial while MIN BISECTION is *NP*-hard. Several optimization problems, for example MAX BISECTION and MIN BISECTION, can be formulated as Boolean constraint satisfaction problems where a feasible solution is a solution with as many variables set to 0 as variables set to 1. For example, for an instance of MIN BISECTION represented by a graph G with n vertices and m edges, we consider n Boolean variables $x_1, \ldots, x_n$ and m constraints by associating with each edge (i, j) of G the constraint $x_i \oplus x_j = 0$. In this way, MIN BISECTION is the problem that consists of finding a solution with as many variables set to 0 as to 1 such that a minimum number from the m previous constraints are non-satisfied.

DEFINITION 1.9.– *An assignment is said to be balanced if the number of variables set to 1 is the same as the number of variables set to 0.* BALANCED A *is the variant of the* A *problem where the feasible solutions are balanced assignments and* A *is a decision or optimization problem.*

The exact and approximation complexities of the balanced versions of decision or optimization satisfiablility problems have been studied by Bazgan and Karpinski in [BAZ 05]. In general, if a problem is hard, its balanced version remains hard. On the other hand, several trivial problems become hard.

More exactly, if SAT($\mathcal{F}$) is *NP*-complete then BALANCED SAT($\mathcal{F}$) is also *NP*-complete [BAZ 05]. It is easy to see that MONOTONE-EkSAT is trivial because the assignment of 1 for each variable (if the expression is composed only of positive literals), or the assignment of 0 for each variable (if the expression is composed only of negative literals), is a satisfying assignment. On the other hand, BALANCED MONOTONE-EkSAT is *NP*-complete, for every $k \geqslant 2$ [BAZ 05]. As specified in theorem 1.3, Ek-LIN2, for every $k \geqslant 2$, is polynomial. In the balanced case, the situation is different because for $k = 2$ the problem remains polynomial; however, for $k \geqslant 3$ the problem becomes *NP*-complete even for a set of 0-homogenous or 1-homogenous linear equations [BAZ 05].

The balanced versions of maximization problems have also been studied. As in the case of decision problems, we can show that MAX SAT($\mathcal{F}$) is E-reducible to BALANCED MAX SAT($\mathcal{F}$) [BAZ 05]. It follows that the balanced version is at least as hard to approximate as the general version. Furthermore, BALANCED MAX MONOTONE-EkSAT is *APX*-hard, for every $k \geqslant 2$ [BAZ 05]. BALANCED MAX SAT is approximable up to a factor of $(1 - \frac{1}{e})$ [SVI 01], and BALANCED MAX 2SAT is approximable up to a factor of 0.66 [BLA 02] and randomly approximable up to a factor of $\frac{3}{4}$ [HOF 03]. BALANCED MAX MONOTONE-EkCONJ, $k \geqslant 2$, does not have an approximation scheme if *NP* $\not\subseteq \cap_{\delta>0}$ *BTIME*(2^{n^δ}) [BAZ 05]. BALANCED MAX E2-LIN2 in the 1-homogenous case, which corresponds to MAX BISECTION, is *APX*-hard [PAP 88, HÅS 97], and BALANCED MAX E2-LIN2 in the 0-homogenous case, which corresponds to BALANCED MAX UNCUT, does not have an approximation scheme if *NP* $\not\subseteq \cap_{\delta>0}$ *BTIME*(2^{n^δ}) [BAZ 05]. Furthermore, BALANCED MAX Ek-LIN2 is *APX*-hard for every $k \geqslant 3$ even in the 0-homogenous or 1-homogenous case [BAZ 05]. Also, using the PCP (*probabilistically checkable proof*) technique, Holmerin [HOL 02] showed that the BALANCED MAX E4-LIN2 problem in the 0-homogenous case is not approximable up to a factor of 0.912. Holmerin and Khot [HOL 03] showed that the BALANCED MAX E3-LIN2 problem in the 0-homogenous case is not approximable up to a factor of $(\frac{3}{4} - \varepsilon)$, for every $\varepsilon > 0$. Recently, Holmerin and Khot [HOL 04] showed that the BALANCED MAX E3-LIN2 problem in the 0-homogenous case is not approximable up to a factor of $(\frac{1}{2} - \varepsilon)$, for every $\varepsilon > 0$, if *NP* $\not\subseteq \cap_{\delta>0}$ *DTIME*(2^{n^δ}), obtaining in this way the best non-approximability result for this problem because it is easily approximable up to a factor of $\frac{1}{2}$.

For minimization problems, we can establish the same result as for maximization problems: BALANCED MIN SAT DELETION($\mathcal{F}$) is at least as hard to approximate as MIN SAT DELETION($\mathcal{F}$), for every set $\mathcal{F}$. BALANCED MIN MONOTONE-EkSAT DELETION, $k \geqslant 2$, does not have an approximation scheme if *P*≠*NP*, and BALANCED MIN MONOTONE-EkCONJ DELETION, $k \geqslant 2$, does not have an approximation scheme if *NP* $\not\subseteq \cap_{\delta>0}$ *BTIME*(2^{n^δ}) [BAZ 05]. Holmerin and Khot [HOL 03] established a lower bound for a generalization of MIN BISECTION. More exactly, they showed that BALANCED MIN E3-LIN2 DELETION, even in the 0-homogenous

and 1-homogenous cases, is not c-approximable, for every constant $c > 1$, if $P \neq NP$. BALANCED MIN E2-LIN2 DELETION in the 1-homogenous case corresponds to BALANCED MIN UNCUT, which has been shown to be *APX*-hard [GAR 93]. BALANCED MIN E2-LIN2 DELETION in the 0-homogenous case is MIN BISECTION. The approximation complexity of MIN BISECTION has not been established. The best algorithm approximates the problem up to a factor of $O(\log n^2)$ [FEI 00]. Recently, Khot [KHO 04] established that if $NP \not\subseteq \cap_{\delta>0}$ $BTIME(2^{n^\delta})$ then MIN BISECTION does not have a polynomial time approximation scheme. BALANCED MIN Ek-LIN2 DELETION, for $k \geqslant 4$, in the 0-homogenous and 1-homogenous cases does not have an approximation scheme if $P \neq NP$ [BAZ 05].

1.7. Conclusion

Satisfiability problems remain central problems to the theory of complexity. This chapter shows the considerable theoretical progress made in the study of satisfiability problems in recent decades. We now have an almost complete characterization of the exact and approximation complexity of these problems as well as of various particular instances.

1.8. Bibliography

[ARO 95] ARORA S., KARGER D., KARPINSKI M., "Polynomial time approximation schemes for dense instances of NP-hard problems", *Proceedings of 27th Annual ACM Symposium on the Theory of Computing*, p. 284–293, 1995, also published in *Journal of Computer and System Sciences*, 58, 193–210, 1999.

[ASA 00] ASANO T., WILLIAMSON D. P., "Improved approximation algorithms for MAX SAT", *Proceedings of the 11th ACM-SIAM Symposium on Discrete Algorithms*, p. 96–105, 2000.

[AVI 02] AVIDOR A., ZWICK U., "Approximating MIN 2-SAT and MIN 3-SAT", *Proceeding of 13th Annual International Symposium on Algorithms and Computation, 465–475*, vol. LNCS 2518, Springer-Verlag, 2002, also published in *Theory of Computing Systems*, 38(3), 329–345, 2005.

[BAZ 99] BAZGAN C., FERNANDEZ DE LA VEGA W., "A polynomial time approximation scheme for dense MIN 2SAT", CIOBANU G., PAUN G., Eds., *Proceedings of the 12th International Symposium on the Fundamentals of Computation Theory*, LNCS 1684, Springer-Verlag, p. 91–99, 1999.

[BAZ 02] BAZGAN C., FERNANDEZ DE LA VEGA W., KARPINSKI M., "Approximability of Dense Instances of Nearest Codeword Problem", PENTTONEN M., MEINECHE SCHMIDT E., Eds., *Proceedings of the 8th Scandinavian Workshop on Algorithm Theory*, LNCS 2368, Springer-Verlag, p. 298–307, 2002.

[BAZ 03] BAZGAN C., FERNANDEZ DE LA VEGA W., KARPINSKI M., "Polynomial time approximation schemes for dense instances of the minimum constraint satisfaction", *Random Structures and Algorithms*, vol. 23(1), p. 73–91, 2003.

[BAZ 05] BAZGAN C., KARPINSKI M., "On the Complexity of Global Constraint Satisfaction", *Proceeding of 16th Annual International Symposium on Algorithms and Computation*, vol. LNCS 3827, Springer-Verlag, p. 624–633, 2005.

[BER 96] BERTSIMAS D., TEO C.-P., VOHRA R., "On dependent randomized rounding algorithms", *Proceeding of the 5th International Integer Programming and Combinatorial Optimization Conference*, vol. LNCS 1084, Springer-Verlag, p. 330–344, 1996.

[BER 02] BERMAN P., KARPINSKI M., "Approximation Hardness of Bounded Degree MIN-CSP and MIN-BISECTION", *Proceeding of the 29th International Colloquium on Automata, Languages and Programming*, vol. LNCS 2380, Springer-Verlag, p. 623–632, 2002.

[BER 03a] BERMAN P., KARPINSKI M., Improved Approximation Lower Bounds on Small Occurrence Optimization, ECCC Technical Report, TR03-008, 2003.

[BER 03b] BERMAN P., KARPINSKI M., SCOTT A., Approximation Hardness and Satisfiability of Bounded Occurrence Instances of SAT, ECCC Technical Report, TR03-022, 2003.

[BER 03c] BERMAN P., KARPINSKI M., SCOTT A., Approximation Hardness of Short Symmetric Instances of MAX-3SAT, ECCC Technical Report, TR03-049, 2003.

[BLA 02] BLASER M., MANTHEY B., "Improved Approximation Algorithms for Max 2Sat with Cardinality Constraint", *Proceedings of the 13th Annual International Symposium on Algorithms and Computation*, LNCS 2518, Springer-Verlag, p. 187–198, 2002.

[BOD 98] BODLAENDER H. L., "A partial k-arboretum of graphs with bounded treewidth", *Theoretical Computer Science*, vol. 209, p. 1–45, 1998.

[COO 71] COOK S., "The complexity of theorem-proving procedures", *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, p. 151–158, 1971.

[CRA 94] CRAWFORD J., BAKER A., "Experimental results on the application of satisfiability algorithms to scheduling problems", *Proceedings of the 12th National Conference on IA*, p. 1092–1097, 1994.

[CRE 95a] CREIGNOU N., "A dichotomy theorem for maximum generalized satisfiability problems", *Journal of Computer and System Sciences*, vol. 51(3), p. 511–522, 1995.

[CRE 95b] CRESCENZI P., KANN V., A compendium of NP optimization problems, http://www.nada.kth.se/∼viggo/problemlist/compendium.html, 1995.

[CRE 96] CRESCENZI P., SILVESTRI R., TREVISAN L., "To weight or not to weight: where is the question?", *Proceedings of the 4th Israeli Symposium on Theory of Computing and Systems*, p. 68–7, 1996.

[CRE 01] CREIGNOU N., KHANNA K., SUDAN M., *Complexity Classifications of Boolean Constraint Satisfaction Problems*, SIAM Monographs on Discrete Mathematics and Applications, 2001.

[DIN 98] DINUR I., KINDLER G., SAFRA S., "Approximating-CVP to within almost-polynomial factors is NP-hard", *Proceeding of the 39th IEEE Annual Symposium on Foundations of Computer Science*, p. 99–109, 1998.

[DOW 84] DOWLING W. F., GALLIER J. H., "Linear time algorithms for testing the satisfiability of propositional Horn formulae", *Journal of Logic Programming*, vol. 3, p. 267–284, 1984.

[DUB 90] DUBOIS O., "On the r, s-SAT satisfiability problem and a conjecture of Tovey", *Discrete Applied Mathematics*, vol. 26, p. 51–60, 1990.

[DYE 86] DYER M., FRIEZE A., "Planar 3DM is NP-complete", *Journal of Algorithms*, vol. 7, p. 174–184, 1986.

[ERD 73] ERDŐS P., SELFRIDGE J., "On a combinatorial game", *Journal of Combinatorial Theory A*, vol. 14, p. 298–301, 1973.

[FEI 95] FEIGE U., GOEMANS M. X., "Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT", *Proceedings of the 3rd Israel Symposium on Theory of Computing and Systems*, p. 182–189, 1995.

[FEI 00] FEIGE U., KRAUTHGAMER R., "A polylogarithmic approximation of the Minimum Bisection", *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, p. 105–115, 2000.

[GAR 74] GAREY M. R., JOHNSON D. S., STOCKMEYER L., "Some simplified NP-complete problems", *Proceedings of the Conference record of 6th Annual ACM Symposium on Theory of Computing*, p. 47–63, 1974, also published in *Theoretical Computer Science*, 1, 237–267, 1976.

[GAR 79] GAREY M. R., JOHNSON D. S., *Computers and Intractability/A Guide to the Theory of NP-Completeness*, W.H. Freeman & Company, San Francisco, 1979.

[GAR 93] GARG N., VAZIRANI V., YANNAKAKIS M., "Approximate Max-flow Min-(multi)cut theorems and their applications", *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, p. 698–707, 1993, also published in *SIAM Journal on Computing*, 25, 235–251, 1996.

[GEN 99] GENT I., WALSH T., The search for satisfaction, Report , Internal Report, Dept. of Computer Science, University of Strathclyde, 1999.

[GOE 94] GOEMANS M., WILLIAMSON D., "New 3/4-approximation algorithms for Max Sat", *SIAM Journal on Discrete Mathematics*, vol. 7, p. 656–666, 1994.

[GOE 95] GOEMANS M., WILLIAMSON D., "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming", *Journal of the ACM*, vol. 42, p. 1115–1145, 1995.

[GU 97] GU J., PURDOM P., FRANCO J., WAH B., "Algorithms for the satisfiability problem: a survey", *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society 35, p. 19–151, 1997.

[HAL 34] HALL P., "On representations of subsets", *Journal London Math. Soc.*, vol. 10, p. 26, 1934.

[HÅS 97] HÅSTAD J., "Some optimal inapproximability results", *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, p. 1–10, 1997.

[HOF 03] HOFMEISTER T., "An approximation algorithm for MAX-2-SAT with cardinality constraint", *Proceedings of the 11th Annual European Symposium on Algorithms*, LNCS 2832, Springer-Verlag, p. 301–312, 2003.

[HOL 02] HOLMERIN J., PCP with Global Constraints – Balanced Homogeneous Linear Equations, manuscript, 2002.

[HOL 03] HOLMERIN J., KHOT S., "A strong inapproximability result for a generalization of Minimum Bisection", *Proceedings of the 18th IEEE Conference on Computational Complexity*, p. 371–378, 2003.

[HOL 04] HOLMERIN J., KHOT S., "A new PCP outer verifier with applications to homogeneous linear equations and Max-Bisection", *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, p. 11–17, 2004.

[HOO 98] HOOS H., Stochastic Local Search – Methods, Models, Applications, http://www.cs.ubc.ca/spider/hoos/publ-ai.html, PhD thesis, TU Darmstadt, 1998.

[HUN 94] HUNT III H., MARATHE M., RADHAKRISHNAN V., RAVI S., ROSENKRANTZ D., STEARNS R., "Approximation schemes using L-reductions", *Proceedings of the 14th Conference on Foundations of Software Technology and Theoretical Computer Science*, LNCS 880, Springer-Verlag, p. 342–353, 1994.

[JIA 95] JIANG Y., KAUTZ H., SELMAN B., "Solving problems with hard and soft constraints using a stochastic algorithm for Max-Sat", *First International Joint Workshop on Artificial Intelligence and Operation Research*, p. 1–15, 1995.

[JOH 74] JOHNSON D. S., "Approximation algorithms for combinatorial problems", *Journal of Computer and System Sciences*, vol. 9, p. 256–278, 1974.

[JON 77] JONES N. D., LAASER W. T., "Complete problems for deterministic polynomial time", *Theoretical Computer Sciences*, vol. 3, p. 107–117, 1977.

[KAR 01] KARPINSKI M., "Approximating bounded degree instances of NP-hard problems", *Proceedings of the 13th Symposium on Fundamentals of Computation Theory*, LNCS 2138, Springer-Verlag, p. 24–34, 2001.

[KAU 92] KAUTZ H., SELMAN B., "Planning as Satisfiability", *Proceedings of the 10th ECAI*, p. 359–363, 1992.

[KHA 96] KHANNA S., MOTWANI R., "Towards a syntactic characterization of PTAS", *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, p. 329–337, 1996.

[KHA 97a] KHANNA S., SUDAN M., TREVISAN L., "Constraint satisfaction: the approximability of minimization problems", *Proceedings of the 12th Annual IEEE Conference on Computational Complexity*, p. 282–296, 1997.

[KHA 97b] KHANNA S., SUDAN M., WILLIAMSON D., "A complete classification of the approximability of maximization problems derived from Boolean constraint satisfaction", *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, p. 11–20, 1997.

[KHO 04] KHOT S., "Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique", *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, p. 136–145, 2004.

[KLA 96] KLAUCK H., "On the hardness of global and local approximation", *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory*, LNCS 1097, Springer-Verlag, p. 88–99, 1996.

[KLE 97] KLEIN P., PLOTKIN S., RAO S., TARDOS E., "Approximation algorithms for Steiner and directed multicuts", *Journal of Algorithms*, vol. 22, num. 2, p. 241-269, 1997.

[KOH 94] KOHLI R., KRISHNAMURTI R., MIRCHANDANI P., "The minimum satisfiability problem", *SIAM Journal on Discrete Mathematics*, vol. 7, p. 275–283, 1994.

[KRA 93] KRATOCHVIL J., SAVICKY P., TUZA Z., "One more occurrence of variable makes satisfiability jump from trivial to NP-complete", *SIAM Journal on Computing*, vol. 22(1), p. 203–210, 1993.

[KRA 00] KRATOCHVIL J., TUZA Z., "On the complexity of bicoloring clique hypergraphs of graphs (extended abstract)", *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, p. 40–41, 2000.

[LEV 73] LEVIN L. A., "Universal sorting problems", *Problems of Information Transmission*, vol. 9, p. 265–266, 1973.

[LIC 82] LICHTENSTEIN D., "Planar formulae and their uses", *SIAM Journal on Computing*, vol. 11(2), p. 329-343, 1982.

[MIN 88] MINOUX M., "LTUR: a simplified linear-time unit resolution algorithm for Horn formulae and computer implementation", *Information Processing Letters*, vol. 29, p. 1–12, 1988.

[PAP 88] PAPADIMITRIOU C., YANNAKAKIS M., "Optimization, approximation, and complexity classes", *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, p. 229-234, 1988, also published in *Journal of Computer and System Sciences*, 43, 425–440, 1991.

[PAP 94] PAPADIMITRIOU C., *Computational Complexity*, Addison-Wesley Publishing Company, Reading, 1994.

[SCH 78] SCHAEFER T., "The complexity of satisfiability problems", *In Conference Record of the 10th Annual ACM Symposium on Theory and Computing*, p. 216–226, 1978.

[SVI 01] SVIRIDENKO M. I., "Best possible approximation algorithm for MAX SAT with cardinality constraint", *Algorithmica*, vol. 30(3), p. 398–405, 2001.

[TAR 72] TARJAN R., "Depth first search and linear graph algorithms", *SIAM Journal on Comput.*, vol. 1(2), p. 146–160, 1972.

[TOV 84] TOVEY C., "A simplified satisfiability problem", *Discrete Applied Mathematics*, vol. 8, p. 85–89, 1984.

[TRE 96] TREVISAN L., "Parallel approximation algorithms by positive linear programming", *Proceedings of the 4th European Symposium on Algorithms*, LNCS 1136, Springer-Verlag, p. 62-75, 1996, also published in *Algorithmica*, 21, 72–88, 1998.