Mechanism Design for Boolean Constraint Satisfaction Problems

Ákos Lukovics

School of Computer and Communication Sciences
Computer Science
Master Project

June 2014

Supervisor
Dr. Moran Feldman
EPFL / THL2

Responsible
Prof. Ola Svensson
EPFL / THL2

Contents

Abstract

In this thesis, we explore mechanism design without money for boolean constraint satisfaction problems. We present three main results. First, we show that for every boolean constraint satisfaction problem there is an optimal deterministic strategy proof mechanism. Second, we prove that there does not exist a deterministic group-strategy proof mechanism with a non-zero constant approximation ratio for Max-SAT, Max-Cut and Max-DiCut. Third, we describe a randomized group-strategy proof mechanism providing an approximation ratio of $1 - \varepsilon$ (for every constant $\varepsilon > 0$) for boolean constraint satisfaction problems without tautologies and contradictions. We also present some additional results not listed here, and describe and analyze mechanisms for Max-SAT, Max-Cut and Max-DiCut.

Chapter 1

Introduction

In multi-agent[1] settings of optimization problems, multiple self-interested players try to reach a common decision. This decision making process can be carried out by functions aggregating the preferences of the players. Algorithmic mechanism design is an emerging field designing such functions, which are called mechanisms in this context. The goal of a mechanism is to maximize an objective function. These objective functions usually depend on private information known only to the players themselves. However, as the players are not forced to reveal their true preferences, the mechanism has to incentivize the players in some way to do so. Most previous works achieved this through monetary payments. Unfortunately, payments are not possible in all settings, for example, when matching organ donors with patients, payments are considered unethical. Without payments achieving truthfulness is more difficult. Procaccia and Tennenholtz suggested in their seminal paper [1] to use approximations to achieve truthfulness. They show, in the context of facility location problems, that it is possible to overcome some impossibility results by designing mechanisms that are only approximately optimal. Prior to their paper, most works in mechanism design aimed for exact results. Since the publication of their paper, approximation mechanisms without money have been designed for a multitude of other problems.[2] However, most problems considered were computationally tractable, and approximation was only used to overcome impossibilities regarding truthfulness. We consider both polynomial and exponential time mechanisms in this document. In this thesis, we explore mechanism design without money for boolean constraint satisfaction problems, which

[1] Agents are referred to as players in the remainder of this thesis.
[2] See [2] for a survey.

are computationally intractable.[3] We consider general boolean constraint satisfaction problems and a few special classes, in particular Max-SAT, Max-DiCut, and Max-Cut.

Section 1.1 provides an introduction to boolean constraint satisfaction problems, Section 1.2 introduces the reader to mechanism design, and Section 1.3 summarizes our main results.

1.1 Boolean constraint satisfaction problems

A (weighted) constraint satisfaction problem (CSP) is defined by a quadruplet $\langle X, C, D, W\rangle$ with $X = \{x_1, \ldots, x_n\}$ being a set of variables, $C = \{c_1, \ldots, c_m\}$ being a set of clauses (or contraints), $D = \{d_1, \ldots, d_n\}$ being a set of domains for the variables (that is, each variable x_i takes its value from the non-empty set d_i), and $W = \{w_1, \ldots, w_m\}$ (with every w_i being a positive rational number) being a set of weights assigned to the clauses. The unweighted version is a special case of CSPs where every clause has weight 1. An assignment $A = (v_1, \ldots, v_n)$ with $v_i \in d_i$ $(1 \leq i \leq n)$ assigns a value to each variable $x \in X$ from its respective domain. A clause is a function $c : d_1 \times \cdots \times d_n \to \{0, 1\}$, in other words, a mapping from an assignment to a logical value[4]. A clause c is satisfied by an assignment a if the clause evaluates to 1 given the assignment a, i.e., $c(a) = 1$. The objective of a constraint satisfaction problem is to find an assignment maximizing the number of satisfied clauses.

$w_a(C)$ and $w(a, C)$ are used as a shorthand for the weight of an assignment, i.e.,

$$w_a(C) = w(a, C) = \sum_{\substack{i \in \{1,\ldots,n\} \\ c_i(a) = true}} w_i.$$

$W(c_i)$ refers to the weight of clause c_i, i.e., $W(c_i) = w_i$. In this thesis, we constrain ourselves to boolean constraint satisfaction problems (bCSP). These are CSP instances with boolean variables, that is, the domain of every variable $x \in X$ is $\{0, 1\}$.

[3] However, this is not the first work to explore approximate mechanism design without money for computationally intractable problems. For example, Dughmi and Ghosh considered variants of the generalized assignment problem [3].

[4] 0/1 and false/true is used interchangeably in this text.

The next subsections describe a few interesting classes of bCSPs for which we present results in later chapters. The difference between general bCSP and the classes defined in the next subsections is that each class is characterized by a limited set of allowed clauses (e.g., in Max-SAT only disjunctive clauses are allowed), whereas in the general case every boolean function of n variables is allowed.

All of the problems we consider in this thesis (except for Max-1-SAT) are NP-hard. The decision version of boolean constraint satisfiability was shown to be NP-complete by Cook [4].

1.1.1 Max-SAT

A literal is an expression consisting of a single boolean variable and a sign. A positive (resp. negative) literal is satisfied if and only if its variable is set to true (resp. false). Clauses in Max-Sat are defined to be disjunctions of literals. Such clauses are satisfied if and only if at least one of its literals is satisfied. A CNF ϕ is a logical formula that is a conjunction of clauses $c_1, \ldots, c_k$, i.e., $\phi = \bigwedge c_i$. Given a CNF formula ϕ, the objective of maximum-satisfiability (or Max-SAT) is to find an assignment a maximizing the number of clauses satisfied in ϕ. Max-k-SAT is a subclass of Max-SAT where every clause contains at most k literals. Horn-SAT is another subclass of Max-SAT where the clauses contain at most one positive literal[5].

The decision version of Max-2-SAT is known to be NP-complete [5], hence variants of Max-SAT with clauses of size 2 or larger allowed are NP-hard. Max-1-SAT can be solved in polynomial time using a simple greedy algorithm.

Max-SAT can be approximated to a constant factor in polynomial time. A simple randomized algorithm setting each variable uniformly to 0 or 1 achieves an approximation ratio of $1/2$, Yannakakis improved this to $3/4$ using network flow techniques [6]. The best known algorithm achieves an approximation ratio of 0.7968 [7]. On the other hand, it is impossible to approximate Max-SAT in polynomial time to a factor of $7/8 + \varepsilon$ (for any constant $\varepsilon > 0$) unless P=NP [8].

[5] Such clauses are also called Horn-clauses.

1.1.2 Max-Cut

Given an undirected (weighted) graph $G = (V, E)$ and edge weights $w_e \in \mathbb{R}^+$ (for every $e \in E$), a cut C of graph G is defined as a bipartition (V_0, V_1) of the graphs vertices. An edge $e = (u, v)$ is said to be cut (or cross the cut) if its endpoints are in different partitions of the cut. The weight of a cut C for a graph G is defined as the sum of the weights of the graph's edges crossing the cut.

Max-Cut is a combinatorial optimization problem in which given an undirected (weighted) graph $G = (V, E)$ and edge weights, the objective is to find a cut C of maximum weight.

Max-Cut can also be formulated as a bCSP by replacing each vertex v_i with a variable x_i, and every edge (v_i, v_j) with a constraint $x_i \oplus x_j$[6] with a matching weight. An assignment of false to variable x_i corresponds to vertex v_i being placed in partition V_0, and an assignment of true to variable x_i corresponds to vertex v_i being placed in partition V_1. Clearly, a constraint c corresponding to an edge e is satisfied if and only if edge e is cut.

The decision version of Max-Cut was shown to be NP-complete by Karp [9], thus the optimization version is NP-hard. The optimization version can be approximated to a factor of $1/2$ using a simple randomized algorithm. An SDP-based algorithm proposed by Goemans and Williamson improves the approximation ratio to $\alpha_{GW} \approx .878$ [10]. Assuming the unique games conjecture[11], this is the best possible approximation ratio achievable by a polynomial time algorithm [12].

1.1.3 Max-DiCut

Given a directed (weighted) graph $G = (V, E)$ and edge weights $w_e \in \mathbb{R}^+$ (for every $e \in E$), a directed cut C of graph G is defined as a bipartition (V_0, V_1) of the graphs vertices. A directed edge $e = (u, v)$ is cut by C (or crosses cut C) if vertex u is in partition V_0, and vertex v is in partition V_1. The weight of a directed cut C is equal to the sum of the weights of the cut edges. Max-DiCut can be seen as a generalization of Max-Cut for directed graphs.

[6] $\oplus$ denotes the exclusive-or (xor) operator.

Analogously to Max-Cut, Max-DiCut can also be formulated as a bCSP problem by replacing each vertex v_i with a variable x_i, and every edge (v_i, v_j) with a constraint $\neg x_i \wedge x_j$ with a matching weight. An assignment of false to variable x_i corresponds to vertex v_i being placed in partition V_0, and an assignment of true to variable x_i corresponds to vertex v_i being placed in partition V_1. It can be verified that an edge e is cut if and only if the matching constraint c is satisfied.

Max-DiCut is known to be NP-hard, as Max-Cut can be reduced to Max-DiCut by replacing every undirected edge (u, v) with two directed edges (u, v) and (v, u). Max-DiCut can be approximated to a factor of $1/4$ using a simple randomized algorithm. Using linear programming the approximation ratio can be improved to $1/2$ [13]. The best known algorithm achieves an approximation ratio of 0.874 using SDP [14]. A class of algorithms that choose a cut based only on the degrees of the vertices, proposed by Feige and Jozeph, called oblivious algorithms achieve an approximation ratio of .483 [15]. In the same paper, it was shown that there is no oblivious algorithm with an approximation ratio of 0.4899.

1.2 Mechanism design

Mechanism design is a field interested in designing functions, or so called mechanisms, that facilitate collective decision making for a set of players. In our setting, m players $p_1, \ldots, p_m$ try to decide on an assignment a to n boolean variables $x_1, \ldots, x_n$. Every player p_i has a weight w_i, and a private clause c_i^{priv} known only to him. The weights of the players and the set of variables are assumed to be common knowledge. Given an assignment a, we define the utility function of player p_i as $u_i = c_i^{priv}(a)$. Alternatively, given a probability distribution P over assignments and a random assignment a from P, the utility of p_i is equal to the probability of the assignment a satisfying c_i^{priv}: $u_i = \mathbb{E}[c_i^{priv}(a)]$. The goal of every player is to maximize its utility function. A mechanism $\mathcal{M}$ is a function that aggregates the preferences of the players and chooses an assignment. Every player is required to report a clause c_i^{public} to $\mathcal{M}$. We refer to c_i^{public} as the public or reported clause of p_i. The reported clause c_i^{public} does not necessarily match the private clause $c_i^{private}$. That is, players might misreport their preference. If a player p_i changes his report from c_i to c_i', we say p_i deviated from c_i to c_i'. Formally, we define a deterministic mechanism $\mathcal{M}$ as a mapping from a set of m reported clauses to an assignment to n variables, i.e.,

$$\mathcal{M} : \{\{0,1\}^n \to \{0,1\}\}^m \to \{0,1\}^n.$$

A randomized mechanism maps the reported clauses to a probability distribution of assignments. We sometimes use the notation $a = \mathcal{M}(C)$ to denote the assignment a chosen by mechanism $\mathcal{M}$ for reported clauses C. We are interested in designing mechanisms choosing assignments that (approximately) maximize the social welfare (SW). The SW of an assignment a is defined as the sum of the players' utilities times their weight: $SW(a) = \sum_i w_i \cdot c_i^{priv}(a)$. As calculating SW requires knowledge about the players' private clauses, the mechanism has to incentivize players to report truthfully (do not misreport their true preference)[7]. A mechanism is strategy proof (SP) if none of the players can gain by misreporting. That is, for every player p_i and every possible reported clause c_i, the utility of the player is maximal when reporting truthfully, i.e.,

$$\forall p_i : \forall c_i : u_i(c_i^{priv}, c_{-i}) \geq u_i(c_i, c_{-i}),$$

where c_{-i} denotes the reports of the other players. A mechanism is group-strategy proof (GSP) if no coalition of players can gain by misreporting. That is, there is no coalition of players P_{coal} for which after a joint deviation from their private clauses c_{coal}^{priv} to some c_{coal} none of the coalitions' players lose, and at least one of the players strictly gains, i.e.,

$$\begin{aligned} \forall P_{coal} : \forall c_{coal} : (\exists p_i \in P_{coal} : u_i(c_{coal}, c_{-coal}) > u_i(c_{coal}^{priv}, c_{-coal})) \\ \rightarrow (\exists p_i \in P_{coal} : u_i(c_{coal}, c_{-coal}) < u_i(c_{coal}^{priv}, c_{-coal})), \end{aligned}$$

where c_{-coal} denotes the reports of the players outside the coalition. Clearly, a GSP mechanism is always SP[8], but an SP mechanism is not necessarily GSP. Unfortunately, for some problems it is not possible to design an optimal SP (or GSP) mechanism, so one has to resort to approximations. We say a mechanism provides an α approximation, if for every possible set of private clauses, the social welfare of the chosen assignment a is worse by at most a factor of α than the social welfare of the optimal assignment a_{opt}, i.e., $\frac{SW(a)}{SW(a_{opt})} \geq \alpha$.[9] The same clause is allowed to be reported by multiple players even in the unweighted case. We show in chapter 2 that strategy

[7] In most previous works on mechanism design, truthfulness is incentivized using payments. That is, the mechanism not only chooses an assignment, but also a value v_i that every player p_i has to "pay". Thus, the utility function depends not only on the chosen assignment but also on v_i. However, in some scenarios payments are impossible to enforce (e.g., in the case of mechanisms involving the web) or are unethical (for example, when matching kidney patients to organ donors). In such scenarios there are no payments, or equivalently they are always set to zero. This is the case we consider in this thesis.

[8] This follows from the fact that coalitions consisting of a single player are allowed.

[9] In this case the social welfare is calculated with respect to the reported clauses instead of the private clauses.

proofness and group-strategy proofness for the unweighted case generalize to the weighted case (assuming rational weights). In some problems the players' private and public clauses are restricted to a set of allowed clauses (e.g., in case of Max-SAT only disjunctive clauses are allowed).

Approximate mechanism design without money was introduced by Procaccia and Tennenholtz in their paper [1] in which they describe mechanisms without money for facility location problems. Several other classes of problems have been considered since then, Boicheva surveys several of them in her thesis [2]. A multi-agent version of Max-SAT was considered by O'Connell and Stearns [16]. In contrast to this thesis, they restrict themselves to polynomial time computation, and use different notions of truthfulness. They describe a deterministic $1/2$-approximation algorithm similar to one of our mechanisms (see Mechanism 3.33).

1.3 Our results

The main results of this thesis are showing that for every class of bCSP without tautologies there exists a $(1 - \varepsilon)$ (for every constant $\varepsilon > 0$), and proving that there does not exist a deterministic GSP mechanism for Max-SAT, Max-Cut, and Max-DiCut with a non-zero constant approximation ratio.

In Chapter 2, we prove that there exists an optimal deterministic SP mechanism for every class of bCSP problems (see Theorem 2.6), and describe a randomized GSP mechanism that provides a $(1-\varepsilon)$-approximation (for every constant $\varepsilon > 0$) for classes of bCSP problems without tautologies and contradictions (see Corollary 2.9). Furthermore, we prove that approximation guarantees for the unweighted case generalize to the weighted case for every bCSP class (see Section 2.5). Finally, we show that for classes of bCSP with tautologies and/or contradictions mechanisms choosing assignments from distributions depending only on the number of variables are almost optimal (see Section 2.4). In later chapters we complement this result by showing that the almost best randomized mechanism for Max-SAT, Max-Cut, and Max-DiCut with tautologies and/or contradictions is the one that chooses the value of every variable with uniform probability.

In chapter 3 and 4, we show that there is no deterministic GSP mechanism for Max-SAT, Max-Cut, and Max-DiCut with a constant non-zero approximation ratio.

We also describe and analyze several mechanisms for these bCSP classes. Table 1.1 provides an overview of the approximation ratios of the best polynomial time mechanisms we describe. The $1/2$ and $1/4$ randomized approximations for Max-SAT, Max-Cut, and Max-DiCut are achieved by a simple randomized algorithm setting each variable to one of the values $\{true, false\}$ with uniform probability.

Class	Deterministic		Randomized	
	SP	GSP	SP	GSP
Max-SAT	1/2 (3.34)	non-const (3.32)	1/2	1/2
Max-1-SAT	1 (2.6)	1/2 (3.34)	1 (2.6)	1-ε (2.9)
Max-Cut	1/2 (4.48)	–	1/2	1/2
Max-DiCut	1/4 (4.48)	non-const (4.43)	0.483 (4.44)	1/4

Table 1.1: Approximation ratios of the best polynomial time mechanisms. The numbers in parantheses refer to the theorems proving the corresponding results.

Thesis organization

In Chapter 2, we describe results that apply for a large class of boolean constraint satisfaction problems. In Chapter 3, we show results specific for Max-SAT. Finally, in Chapter 4, we show results specific for Max-Cut and Max-DiCut.

Chapter 2

General bCSP results

In this chapter, we present results that hold for general bCSPs. First, we describe a class of deterministic mechanisms, which we call *enumerative mechanisms*, and show that all of them are strategy proof. Second, we prove by construction that for every bCSP problem there is an optimal deterministic strategy proof mechanism. Third, we describe a construction that given an enumerative mechanism (with some restrictions) providing an α-approximation yields a randomized group-strategy proof mechanism providing an $\alpha(1-\varepsilon)$ approximation. Finally, we show that for bCSP problems that allow tautological (or contradictory) clauses, the best approximation ratio is achieved by a mechanism that picks an assignment from a distribution independent of the reported clauses.

2.1 Enumerative mechanisms

We start by defining the class of deterministic mechanisms, which we call *enumerative mechanisms*. Then we show that every enumerative mechanism is strategy proof.

Mechanism 2.1. *Given a bCSP instance* $\mathcal{I} = \langle X, C, D_b, W\rangle$[1]*,* $n = |X| \geq 1$*, and a function* $A(n) \subseteq \{0,1\}^n$*. For every* n*, fix a total order over* $A(n)$*. Choose assignment* a *from set* $A(n)$ *maximizing the total weight of satisfied clauses. In case there are multiple assignments with maximal size, break ties according to the ordering defined on* $A(n)$*.*

In other words, an enumerative mechanism has a pre-defined set of possible assignments for instances with a given number of variables. From this set the assignment with largest weight is chosen (using a consistent tie-breaking

[1] D_b assigns the domain $\{0,1\}$ to every variable. That is, the variables are boolean.

rule).

We will refer to an enumerative mechanism by $\mathcal{M}^{enum}_{A(n)}$.

Definition 2.2. *Given two assignments $a_1, a_2 \in A(n)$ with weights w_1, w_2, we say that $(w_1, a_1) \succ (w_2, a_2)$ if and only if one of the following conditions is met:*

1. *$w_1 > w_2$,*
2. *$w_1 = w_2$ and $a_1 > a_2$ according to the ordering over $A(n)$.*

Basically, $(w_1, a_1) \succ (w_2, a_2)$ means the first pair is lexicographically larger than the second.

Observation 2.3. *Given an enumerative mechanism $\mathcal{M}^{enum}_{A(n)}$ and an instance $\mathcal{I}$ with n variables. Define the set*

$$A_{\mathcal{I}}(n) = \left\{ \Big(w_a(C), a\Big) \;\middle|\; a \in A(n) \right\},$$

and order its tuples lexicographically. Let $(w^{max}_{\mathcal{I}}, a^{max}_{\mathcal{I}})$ be the lexicographically largest pair in $A_{\mathcal{I}}(n)$. Observe that $\mathcal{M}^{enum}_{A(n)}$ chooses $a^{max}_{\mathcal{I}}$.

Theorem 2.4. *Every enumerative mechanism is strategy proof.*

Proof. Consider player p_i with private clause c^{priv}_i. In case p_i reports truthfully, let the chosen assignment be $a^{max}_{\mathcal{I}}$.

If $c^{priv}_i(a^{max}_{\mathcal{I}}) = 1$, the player has no incentive to deviate, as his utility is already maximal.

In the remaining part of this proof we assume $c^{priv}_i(a^{max}_{\mathcal{I}}) = 0$. We only need to show that there is no deviation after which the player's utility increases to 1.

Assume p_i deviates to some clause c'_i. Let us define the problem instance after the deviation as $\mathcal{I}' = \langle X, C - c^{priv}_i + c'_i, W\rangle$. $A_{\mathcal{I}'}$ can be defined in terms of $A_{\mathcal{I}}$ (see Observation 2.3 for the definition) as

$$A_{\mathcal{I}'} = \left\{ f(w, a) \;\middle|\; (w, a) \in A_{\mathcal{I}} \right\},$$

where

$$f(w,a) = \begin{cases} (w,a), & \text{for } c_i^{priv}(a) = c_i'(a) \quad (2.1) \\ (w + W(c_i'), a) & \text{for } c_i^{priv}(a) = 0, c_i'(a) = 1 \quad (2.2) \\ (w - W(c_i^{priv}), a) & \text{for } c_i^{priv}(a) = 1, c_i'(a) = 0 \ . \quad (2.3) \end{cases}$$

From the definition of $f(w,a)$ we see that after any deviation

- the weight $w_{\mathcal{I}}^{max}$ of the assignment $a_{\mathcal{I}}^{max}$ does not decrease, as the assignment did not satisfy c_i^{priv} before the deviation. Thus, for $f(w_{\mathcal{I}}^{max}, a_{\mathcal{I}}^{max})$ either 2.1 or 2.2 applies.
- the weight w_a of any assignment a for which $c_i^{priv}(a) = 1$ holds does not increase, as for $f(w_a, a)$ either 2.1 or 2.3 applies.

By the definition of enumerative mechanisms and by the assumption that $c_i^{priv}(a_{\mathcal{I}}^{max}) = 0$, we know that before the deviation the following was true:

$$\forall (w_a, a) \in A_{\mathcal{I}}, c_i^{priv}(a) = 1 : (w_{\mathcal{I}}^{max}, a_{\mathcal{I}}^{max}) > (w_a, a) \quad (2.4)$$

As the weight of $a_{\mathcal{I}}^{max}$ does not decrease, the weight of any a for which $c_i^{priv}(a) = 1$ holds does not increase, and the ordering on $A(n)$ does not change, 2.4 must still hold after the deviation. In other words, the mechanism prefers $a_{\mathcal{I}}^{max}$ to any of the assignments satisfying c_i^{priv} both before and after the deviation as well.[2] Thus, the assignment chosen for $\mathcal{I}'$ does not satisfy c_i^{priv}, and the utility of p_i remains 0 after any deviation. As the player does not gain by any deviation, the mechanism is indeed strategy proof. □

2.2 Optimal mechanisms

In this section, we show that an optimal deterministic algorithm using a lexicographic tie-breaking rule yields an optimal deterministic strategy proof mechanism. This implies that for every bCSP problem there is an optimal deterministic strategy proof mechanism.

Mechanism 2.5. *Given an instance $\mathcal{I}$ and an optimal algorithm $\mathcal{A}$. Solve $\mathcal{I}$ optimally using $\mathcal{A}$ breaking ties lexicographically.*

Theorem 2.6. *Mechanism* 2.5 *is an optimal strategy proof mechanism.*

[2]The mechanism does not necessarily choose $a_{\mathcal{I}}^{max}$ after the deviation. The only important observation is that it does not choose an assignment satisfying c_i^{priv} after the deviation.

Proof. As Mechanism 2.5 picks an optimal assignment for every instance $\mathcal{I}$, the mechanism is indeed optimal.

Set $A(n) = \{0,1\}^n$. Order the elements of $A(n)$ lexicographically[3]. For strategy proofness, it is enough to note that the enumerative mechanism $\mathcal{M}_{A(n)}^{enum}$ is equivalent to Mechanism 2.5. □

Note 2.7. *Given an optimal algorithm $\mathcal{A}$ for a family of bCSP problems in n variables, the lexicographically largest optimum can be found using $\mathcal{O}(n)$ calls to $\mathcal{A}$.*

2.3 $\alpha(1-\varepsilon)$ randomized GSP mechanisms

In this section, we describe a group-strategy proof randomized mechanism that provides an $\alpha(1-\varepsilon)$ approximation for a class of bCSP given an enumerative mechanism that provides an α-approximation for the same class. The mechanism requires that the bCSP class does not allow tautological or contradictory clauses.[4]

The main theorem and a corollary of it are stated below. We prove the theorem in a later part of this section.

Theorem 2.8. *Mechanism 2.11 is a randomized $\alpha(1-\varepsilon)$-optimal group-strategy proof mechanism.*

Corollary 2.9. *Every class of bCSP without tautologies and contradictions has a randomized GSP mechanism that provides an approximation ratio of $(1-\varepsilon)$.*

Throughout this section $\mathcal{M}_{A(n)}^{enum}$ refers to an enumerative mechanism providing an approximation ratio of α. We denote the assignment chosen by $\mathcal{M}_{A(n)}^{enum}$ by $\widetilde{a}$, and its weight by $\widetilde{w}$.

For a totally ordered set of possible assignments $A(n)$ and assignment $a \in A(n)$, the function $Ord(a, A(n))$ assigns a unique non-negative integer to every assignment a. It assigns zero to the assignment with lowest order, one to the assignment with second lowest order, and so on.

[3]This defines a total ordering on $A(n)$.

[4]A tautological clause is a clause that is satisfied by every assignment, e.g., $x \vee \neg x$. A contradictory clause is one that is not satisfied by any assignment, e.g., $x \wedge \neg x$.

Define the function

$$f_{A(n)}(w,a) = 1 - \frac{(w-1)\cdot|A(n)| + Ord(a, A(n))}{n\cdot|A(n)|} \quad (2.5)$$

and fix a constant $1 > \varepsilon > 0$.

Observation 2.10. *Notice that, by the definition of function f, for two pairs $(w_1, a_1) \neq (w_2, a_2)$*

$$f_{A(n)}(w_1, a_1) > f_{A(n)}(w_2, a_2) \quad \textit{if and only if} \quad (w_1, a_1) \prec (w_2, a_2).$$

The main intuition behind Mechanism 2.11 is to assign a tuple consisting of the weight of an assignment a and a to every instance.
In Mechanism 2.11 and its proof we assume that the instances are unweighted (i.e., the weight of all clauses is 1). The mechanism can be adapted for the weighted case, but we omit details.

Mechanism 2.11 $\alpha(1-\varepsilon)$-optimal randomized mechanism $\mathcal{M}_{rand}$

Require: an instance $\mathcal{I}$ with n variables,
an α approximate enumerative mechanism $\mathcal{M}^{enum}_{A(n)}$,
a constant $0 < \varepsilon < 1$

1: $\widetilde{a} \leftarrow$ the assignment chosen by $\mathcal{M}^{enum}_{A(n)}$ for instance $\mathcal{I}$
2: $\widetilde{w} \leftarrow$ the weight of $\widetilde{a}$
3: $a_{rand} \leftarrow$ a random assignment chosen uniformly from $\{0,1\}^n$
4: $p \leftarrow \varepsilon f_{A(n)}(\widetilde{w}, \widetilde{a})$
5: **return** $\widetilde{a}$ with probability $1-p$ and a_{rand} with probability p

Consider Mechanism 2.11. From this point on till the end of this section we refer to players with private clauses satisfied by $\widetilde{a}$ as winners, and to players with private clauses not satisfied by $\widetilde{a}$ as losers. The expected utility of player p_i for a randomly chosen assignment[5] is denoted by u_i^{random}. The utility of p_i for $\widetilde{a}$ is denoted by u_i^{chosen}. If p_i is a winner $u_i^{chosen} = 1$, if p_i is a loser $u_i^{chosen} = 0$. Observe that $0 < u_i^{random} < 1$, as tautologies and contradictions are not allowed. Let $0 < p < 1$ be the probability of choosing a random

[5]This assignment is chosen uniformly from the set of all assignments $\{0,1\}^n$, and not from $A(n)$.

assignment, and $1-p$ be the probability of choosing $\widetilde{a}$. The expected utility of a winner is

$$u_i = pu_i^{random} + (1-p)u_i^{chosen} = u_i^{random} + (1-p)(1-u_i^{random}). \qquad (2.6)$$

The expected utility of a loser is

$$u_i = pu_i^{random} + (1-p)u_i^{chosen} = pu_i^{random}. \qquad (2.7)$$

Observe, that the utility of a winner is always greater than the utility of a loser, regardless of p.

We will refer to the assignment chosen by the enumerative mechanism after the deviation by $\widetilde{a}'$ and to its weight by $\widetilde{w}'$.

Lemma 2.12. *For a winner p_i and any gainful deviation, the following conditions must hold:*

1. *p_i remains a winner after the deviation: $c_i^{priv}(\widetilde{a}') = 1$,*
2. *$(\widetilde{w}', \widetilde{a}') \succ (\widetilde{w}, \widetilde{a})$.*

Proof. If the first condition is not satisfied, then p_i becomes a loser after the deviation. From equations 2.6 and 2.7 we know that the utility of a loser is always less than the utility of a winner, thus, if the first condition were not satisfied, the utility of p_i would decrease. Hence, the first condition must be satisfied by any gainful deviation.

Suppose the second condition is not satisfied, and $(\widetilde{w}', \widetilde{a}') \preceq (\widetilde{w}, \widetilde{a})$ holds. If $(\widetilde{w}', \widetilde{a}') = (\widetilde{w}, \widetilde{a})$, the probability of choosing a random assignment does not change by the deviation, and the utility of player p_i stays unchanged. If $(\widetilde{w}', \widetilde{a}') \prec (\widetilde{w}, \widetilde{a})$, by Observation 2.10 the value of f increases. By the definition of Mechanism 2.11 (see line 4 of the definition), the probability p of choosing a random assignment increases with increasing f. From equation 2.6, we know that the utility of a winner decreases with increasing p. Hence, no deviation that does not satisfy condition 2 is gainful for a winner. □

Lemma 2.13. *For a loser p_i and any gainful deviation, one of the following two conditions must hold:*

1. *p_i becomes a winner after the deviation: $c_i^{priv}(\widetilde{a}') = 1$,*
2. *(a) p_i remains a loser after the deviation: $c_i^{priv}(\widetilde{a}') = 0$,*

(b) $(\widetilde{w}', \widetilde{a}') \prec (\widetilde{w}, \widetilde{a})$.

Proof. Assume there is a gainful deviation not satisfying any of the two conditions, that is, after the deviation the following two hold:

(I) p_i remains a loser after the deviation: $c_i^{priv}(\widetilde{a}') = 0$, and

(II) $(\widetilde{w}', \widetilde{a}') \succ (\widetilde{w}, \widetilde{a})$.

Condition (II) implies that the probability p of choosing a random assignment after the deviation decreases (see Observation 2.10, and line 4 of the definition of Mechanism 2.11). By equation 2.7, the utility of a loser decreases with decreasing p. Hence, there is no gainful assignment that does not satisfy any of the conditions listed in the lemma. □

Proof of Theorem 2.8. Recall that $\widetilde{a}$ is chosen with probability $1-p$ with $p = \varepsilon f_{A(n)}(w_{opt}, a_{opt})$. As $f_{A(n)}(\widetilde{w}, \widetilde{a}) < 1$ for every possible pair $(\widetilde{w}, \widetilde{a})$, $\widetilde{a}$ is chosen with probability not less than $1-\varepsilon$. Since $\widetilde{a}$ is an α-approximation, the mechanism provides indeed an approximation ratio of $\alpha(1-\varepsilon)$.

For group-strategy proofness we need to show that for every coalition any deviation from truthful reporting that strictly increases at least one of the coalition's players' expected utility, also strictly decreases at least one of its players' utility. We show this by considering all possible types of coalitions in terms of their composition by winners and losers.

We denote the set of players in the coalition by P_{coal}, the reported clause of player p_i after the deviation by c_i', and the set of reported clauses before (resp. after) the deviation by C (resp C').

Observation 2.14. *Notice, that before the deviation*

$$w(\tilde{a}, C) \geq w(\tilde{a}', C), \tag{2.8}$$

and after the deviation

$$w(\tilde{a}, C') \leq w(\tilde{a}', C'). \tag{2.9}$$

Both inequalities follow from the definition of enumerative mechanisms.

The weight of an assignment a for a coalition P_{coal} before the deviation can be calculated by

$$w(a, C) = \sum_{\substack{p_i \in P_{coal}, \\ c_i^{priv}(a)=1}} 1 + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(a)=1}} 1, \tag{2.10}$$

and after the deviation by

$$w(a, C') = \sum_{\substack{p_i \in P_{coal}, \\ c'_i(a)=1}} 1 + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(a)=1}} 1. \tag{2.11}$$

Notice that the only difference between the equations is in the calculation of the weight of the coalition's players.

Observation 2.15. *No gainful deviation leaves the utility of any player unchanged, and every gainful deviation for a coalition P_{coal} must be strictly gainful for every player of coalition P_{coal}.*

Proof. After a gainful deviation either the chosen assignment, or its weight (or both) has to change. Otherwise, the utilities of the players remain unchanged, and the deviation is not gainful. Thus, after the deviation the probability p of choosing a random assignment changes (see Observation 2.10). □

There are three types of coalitions we need to consider. Coalitions that contain both losers and winners, coalitions that contain only winners, and coalitions that contain only losers.

Coalition consisting of both winners and losers. By Observation 2.15, the deviation must be strictly gainful for both losers and winners, hence, the conditions from Lemma 2.12 and 2.13 must be satisfied at the same time. Thus, for any gainful deviation the following must hold:

1. every player of the coalition becomes a winner after the deviation, i.e., $\forall p_i \in P_{coal} : c_i^{priv}(\widetilde{a}') = 1$,
2. the probability of choosing the assignment chosen by the enumerative mechanism increases after the deviation, i.e., $(w(\widetilde{a}', C'), \widetilde{a}') \succ (w(\widetilde{a}, C), \widetilde{a})$.

The first condition is implied by the fact that a change to the probability p of choosing a random assignment can not be gainful for winners and losers at the same time, hence, each player must become a winner. The second condition follows from Lemma 2.13 and condition 2.

As all of the coalition's players are satisfied by the newly chosen assignment $\widetilde{a}'$, its weight before the deviation must be equal to

$$w(\widetilde{a}', C) = |P_{coal}| + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(\widetilde{a}')=1}} 1.$$

The coalition's players' reported clauses after the deviation are not necessarily satisfied by $\widetilde{a}'$, hence, $w(\widetilde{a}', C) \geq w(\widetilde{a}', C')$. From Observation 2.14, we know that $w(\tilde{a}, C) \geq w(\tilde{a}', C)$, and $w(\widetilde{a}, C') \leq w(\widetilde{a}', C')$. To satisfy condition 2, it must be that $w(\widetilde{a}, C) = w(\widetilde{a}', C) = w(\widetilde{a}', C')$. As $w(\widetilde{a}, C) = w(\widetilde{a}', C')$ and $\widetilde{a}$ was chosen before the deviation, it must be that $(w(\widetilde{a}', C'), \widetilde{a}') \prec (w(\widetilde{a}, C), \widetilde{a})$, contradicting condition 2. Thus, there are no gainful deviations for coalitions of this type.

Coalition consisting of losers only. In this case for any gainful deviation one of the conditions from Lemma 2.13 must hold:

1. every player of the coalition becomes a winner after the deviation, i.e., $\forall p_i \in P_{coal} : c_i^{priv}(\widetilde{a}') = 1$,
2. (a) at least one of the players remains a loser after the deviation, i.e., $\exists p_i \in P_{coal} : c_i^{priv}(\widetilde{a}') = 0$
   (b) the probability of choosing the assignment chosen by the enumerative mechanism decreases after the deviation, i.e., $(w(\widetilde{a}', C'), \widetilde{a}') \prec (w(\widetilde{a}, C), \widetilde{a})$,

Observe that the equation

$$w(\widetilde{a}, C) = \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(\widetilde{a})=1}} 1 \tag{2.12}$$

holds, as none of the coalition's players are satisfied by the assignment chosen before the deviation. Hence, $w(\widetilde{a}, C') \geq w(\widetilde{a}, C)$.

We start by showing that there is no deviation for which the first condition holds. Suppose there is a deviation that satisfies condition 1. As all of the coalition's players become winners after the deviation, it must be that

$$w(\widetilde{a}', C) = |P_{coal}| + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(\widetilde{a}')=1}} 1.$$

Observe that $w(\widetilde{a}', C) \geq w(\widetilde{a}', C')$. From Observation 2.14, it follows that $w(\tilde{a}, C) \geq w(\tilde{a}', C)$. Hence, $w(\widetilde{a}, C') \geq w(\widetilde{a}, C) \geq w(\widetilde{a}', C) \geq w(\widetilde{a}', C')$. This implies $(w(\widetilde{a}', C'), \widetilde{a}') \prec (w(\widetilde{a}, C'), \widetilde{a})$. Thus, $\widetilde{a}$ is chosen after the deviation as well. As $\widetilde{a}$ does not satisfy any of the coalitions' players, this deviation does not satisfy condition 1, contradicting our initial assumption.

Now, assume there is a deviation for which condition 2 holds. From equation 2.12 and Observation 2.14, we know that $w(\widetilde{a}', C) \leq w(\widetilde{a}, C) \leq w(\widetilde{a}, C')$. As $\widetilde{a}'$ is chosen after the deviation, it must be that $w(\widetilde{a}', C') > w(\widetilde{a}, C')$. However, this implies $(w(\widetilde{a}', C'), \widetilde{a}') \succ (w(\widetilde{a}, C), \widetilde{a})$, contradicting condition 2(b). Thus, there is no gainful deviation satisfying condition 2.

As there is no gainful deviation satisfying either of the two conditions, coalitions of this type do not gain by deviating.

Coalition consisting of winners only. In this case for any gainful deviation both conditions from Lemma 2.12 must hold:

1. every player of the coalition remains a winner after the deviation, i.e., $\forall p_i \in P_{coal} : c_i^{priv}(\widetilde{a}') = 1$,
2. the probability of choosing the assignment chosen by the enumerative mechanism increases after the deviation, i.e., $(w(\widetilde{a}', C'), \widetilde{a}') \succ (w(\widetilde{a}, C), \widetilde{a})$.

As all of the coalition's players are winners both before and after the deviation, the following two equations must hold:

$$w(\widetilde{a}, C) = |P_{coal}| + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(\widetilde{a})=1}} 1,$$
$$w(\widetilde{a}', C) = |P_{coal}| + \sum_{\substack{p_i \notin P_{coal}, \\ c_i^{priv}(\widetilde{a}')=1}} 1.$$

Notice that $w(\widetilde{a}, C) \geq w(\widetilde{a}, C')$ and $w(\widetilde{a}', C) \geq w(\widetilde{a}', C')$. This observation together with observation 2.14 implies that $w(\widetilde{a}, C) \geq w(\widetilde{a}', C) \geq w(\widetilde{a}', C')$. Condition 2 requires that $w(\widetilde{a}', C') \geq w(\widetilde{a}, C)$, hence, $w(\widetilde{a}', C') = w(\widetilde{a}, C)$. As $\widetilde{a}'$ has lower order than $\widetilde{a}$, it must be that $(w(\widetilde{a}', C'), \widetilde{a}') \prec (w(\widetilde{a}, C), \widetilde{a})$, contradicting condition 2. Therefore, there is no gainful deviation for this type of coalition.

As there are no gainful deviations for any of the three possible types of coalitions, the mechanism is indeed group-strategy proof. □

Note 2.16. *The time complexity of Mechanism 2.11 depends mainly on the enumerative mechanism used by it. It runs in polynomial time when the enumerative mechanism used by it runs in polynomial time.*

2.4 Randomized GSP hardness for bCSP with tautologies

In this section, we show that for classes of bCSP where tautologies (or contradictions) are allowed, the best possible approximation ratio is achieved by a mechanism that returns the same distribution of assignments independently of the players' reports. This result holds only for classes where every player is allowed to report any clause regardless of his private clause (e.g., this result does not hold for classes with publicly known clause sizes[6]).

Theorem 2.17. *Consider a class of bCSP problems with n_{var} variables and a set of allowed clauses C, with at least one clause $c \in C$ being a tautology or a contradiction. Let C' be the set of all allowed non-tautologic and non-contradictory clauses. Over all possible probability distributions P, let p_{best} be an upper bound on the minimum probability that P satisfies a clause $c \in C'$:*

$$\forall P : p_{best} \geq \min_{c \in C'} \mathop{\mathbb{P}}_{a \sim P}[c(a) = 1]. \tag{2.13}$$

There is no randomized group-strategy proof mechanism for this class of problems that provides an approximation ratio better than $p_{best} + \varepsilon$, for any constant $\varepsilon > 0$.

Proof. Let $n = |C'|$. Fix $c_{taut} \in C$ to be a tautological (or contradictory) clause, and choose an integral m such that $m \geq \frac{n}{\varepsilon}$. Consider $n+1$ instances $\mathcal{I}_i$ (for $0 \leq i \leq n$), each with $n + m$ players. In instance $\mathcal{I}_i$ ($i > 0$) every clause $c \in C'$ is reported exactly once, except for clause $c_i \in C'$ that is reported $m+1$ times. In instance $\mathcal{I}_0$, every clause $c \in C'$ is reported exactly once, clause c_{taut} is reported m times.

Let $\mathcal{M}$ be a randomized group-strategy proof mechanism. Suppose, that for instance $\mathcal{I}_0$ mechanism $\mathcal{M}$ chooses a probability distribution P. Observe, that for every instance $\mathcal{I}_i$ ($i > 0$) $\mathcal{M}$ must choose a distribution P_i such that for every clause $c \in C'$ the probability that it is satisfied by an assignment chosen from P_i is less than equal to an assignment chosen from P, i.e.,

$$\forall c \in C' : \mathop{\mathbb{P}}_{a \sim P}[c(a) = 1] \geq \mathop{\mathbb{P}}_{a \sim P_i}[c(a) = 1]. \tag{2.14}$$

Assume condition 2.14 does not hold. Hence, $\exists c \in C' : \mathbb{P}_{a \sim P}[c(a) = 1] < \mathbb{P}_{a \sim P_i}[c(a) = 1]$ must be true. Assume c' is a clause that satisfies this. A

[6] In this class players are only allowed to report clauses with the same size as their private clause.

coalition of the player reporting c' and the players reporting c_{taut} in instance $\mathcal{I}_0$ could deviate such that the new reports would match those of instance $\mathcal{I}_i$. The player reporting c' would gain, and the players reporting c_{taut} would not lose as their utility is always 1 (or 0). Hence, condition 2.14 must hold, otherwise $\mathcal{M}$ would not be GSP.

From the definition of p_{best} (see 2.13) it follows that there must be a clause $c_i \in C'$ such that:

$$\underset{a \sim P}{\mathbb{P}}[c_i(a) = 1] \leq p_{best}.$$

Also, from 2.14 it follows that the same holds for the distribution P_i chosen for instance $\mathcal{I}_i$:

$$\underset{a \sim P_i}{\mathbb{P}}[c_i(a) = 1] \leq p_{best}.$$

The optimal distribution for instance $\mathcal{I}_i$ satisfies at least m clauses (a distribution that satisfies clause c_i with probability 1 has this property, for example), the distribution P_i satisfies at most $n + p_{best}m$ clauses (c_i is satisfied with probability at most p_{best}, everything else with probability at most 1). Thus, the approximation ratio provided by $\mathcal{M}$ for $\mathcal{I}_i$ is no better than:

$$\frac{n + p_{best}m}{m} \leq \frac{n + p_{best}\frac{n}{\varepsilon}}{\frac{n}{\varepsilon}} = p_{best} + \varepsilon. \quad \square$$

The implication of the above theorem is that for classes that allow for tautologies or contradictions, the best randomized group-strategy proof mechanism is only marginally better than a trivial mechanism that chooses the same distribution for every possible instance (the chosen distribution might depend on the number of variables). For Max-Sat, Max-Cut, and Max-DiCut the (asymptotically) best trivial algorithm is the one that sets every variable to true with probability one half. It achieves an approximation ratio of $1/2$ for Max-Sat and Max-Cut, and an approximation ratio of $1/4$ for Max-DiCut. These claims are proven in the next chapters, in theorems 3.30 and 4.41.

2.5 Reduction between unweighted and weighted bCSPs

In this section, we show that the same hardness results for (group-)strategy proofness apply for both the weighted and unweighted cases of bCSPs (assuming rational weights) by reducing the weighted version to an unweighted version. The opposite direction follows directly from the fact that an unweighted bCSP is a special case of the corresponding weighted bCSP with every weight set to one.

The reduction from the unweighted to weighted version introduces a dependency in the time complexity on the weights of the players.

Recall, that the weights are assumed to be public knowledge. In the unweighted case, the weight of every players is assumed to be one. Keep in mind that the same clause can be reported by multiple players, so the unweighted multi-agent version is not equivalent to the unweighted version of the corresponding combinatorial version. The reduction we show here requires the weights to be rational.

Mechanism 2.18 α-approximation mechanism for weighted bCSP

Require: instance $\mathcal{I} = \langle X, C, W, D_b \rangle$,
α approximation mechanism $\mathcal{M}_{unw}$ for unweighted bCSPs

1: Let $\mathcal{I}_{scaled}$ be a scaled version of $\mathcal{I}$ where every weight is scaled to an integer by the same factor f.
2: Let $\mathcal{I}_{unw}$ be an instance where every reported clause c of $\mathcal{I}_{scaled}$ with weight w is replaced by w unweighted clauses.

3: **return** assignment a chosen by $\mathcal{M}_{unw}$ for instance $\mathcal{I}_{unw}$

Observation 2.19. *Every α-approximation mechanism for the unweighted instance provides an α-approximation for the weighted instance.*

Proof. Consider Mechanism 2.18. Observe that every assignment a satisfies the same fraction of the total weight in all three instances $\mathcal{I}, \mathcal{I}_{scaled}, \mathcal{I}_{unw}$. □

Observation 2.20. *A deviation of a player with weight w from clause c to c' in instance $\mathcal{I}$ is equivalent to a deviation of $w \cdot f$ players from clause c to c' in $\mathcal{I}_{unw}$.*

Theorem 2.21. *Mechanism 2.18 is a GSP mechanism for weighted bCSP if $\mathcal{M}_{unw}$ is a GSP mechanism for the unweighted version of the same class.*

Proof. By Observation 2.20, for every deviation on $\mathcal{I}$, there is an equivalent deviation on $\mathcal{I}_{unw}$. Hence, if there is a coalition of players for instance $\mathcal{I}$ that can gain by deviating, there must also be such a coalition for instance $\mathcal{I}_{unw}$. As $\mathcal{M}_{unw}$ is GSP, there is no such coalition for $\mathcal{I}_{unw}$. □

Theorem 2.22. *Mechanism 2.18 is a SP mechanism for weighted bCSP if $\mathcal{M}_{unw}$ is a SP mechanism for the unweighted version of the same class.*

Proof. By Observation 2.20, if Mechanism 2.18 is SP for weighted bCSP, it must be that for every instance $\mathcal{I}$ and every coalition P_{coal} of size w where every player reports clause c, there is no clause c', such that a deviation where every player of P_{coal} deviates from c to c' is gainful.

Assume for the sake of contradiction that there is an instance $\mathcal{I}$ where a coalition P_{coal} of size w can gain by deviating from a clause to c to a clause c'. Consider $w+1$ instances: $\mathcal{I}_i$ $(0 \leq i \leq w)$. In instance $\mathcal{I}_i$, $w-i$ players of P_{coal} report clause c, and i players of P_{coal} report c'. The reports of all other players are equivalent to their reports in $\mathcal{I}$. Notice that if in instance $\mathcal{I}_i$ one of the players reporting c deviates to c', the reported clauses after the deviation match those of instance $\mathcal{I}_{i+1}$. Similarly, if one of the players reporting c' deviates to c, the reported clauses after the deviation match those of instance $\mathcal{I}_{i-1}$. By assumption, in instance $\mathcal{I}_0$ mechanism $\mathcal{M}_{unw}$ chooses an assignment that does not satisfy c, and in instance $\mathcal{I}_w$ it chooses an assignment that satisfies c. Therefore, there must exist an i, such that the assignment chosen for $\mathcal{I}_i$ satisfies c and the one chosen for $\mathcal{I}_{i+1}$ does not satisfy c (or vice versa). However, this means that mechanism $\mathcal{M}_{unw}$ is not SP, as there is a player in either $\mathcal{I}_i$ or $\mathcal{I}_{i+1}$ that can gain by deviating. As this contradicts are initial assumption, there can not exist such an i, and neither (or both) of the assignments for $\mathcal{I}_0$ and $\mathcal{I}_w$ must satisfy clause c. Hence, Mechanism 2.18 must be strategy proof. □

The previous theorems together with Observation 2.19 imply that if there is SP (resp. GSP) α-approximation mechanism for the unweighted version of some class of GSP, there must also be a SP (resp. GSP) mechanism for the weighted version of the same class with matching approximation ratio.

Chapter 3

Max-SAT

In this chapter, we show hardness results for deterministic group-strategy proof mechanisms for Max-SAT and Max-1-SAT, and present deterministic mechanisms with matching approximation ratios.

3.1 Hardness results

In this section, we first show that there is no deterministic group-strategy proof mechanism for Max-Horn-2-SAT[1] that provides a non-zero constant approximation ratio. Second, we show that there is no deterministic group-strategy proof mechanism for Max-1-SAT and Max-Horn-2-SAT with publicly known clause sizes[2] that provides an approximation ratio larger than $1/2$. Third, we prove that a trivial randomized algorithm setting each variable to true with probability $1/2$ provides the best possible approximation ratio for Max-SAT with tautologies.

Theorem 3.23. *There is no deterministic group-strategy proof mechanism for Max-Horn-2-SAT with an approximation ratio of ε (for any constant $\varepsilon > 0$).*

Proof. Let n be an integer larger than $4/\varepsilon$. All the instances we consider in this proof have $n+4$ players, to which we refer by p_i $(1 \le i \le n+4)$, and two variables x, y. There are 4 possible assignments for these instances:

$$a_1 = \{0,0\},\ \ a_2 = \{0,1\},\ \ a_3 = \{1,0\},\ \ a_4 = \{1,1\}.$$

[1]Max-Horn-2-SAT is a subclass of Max-SAT where only Horn-clauses, clauses with at most one positive literal, are allowed, and every clause contains at most 2 literals.

[2]By publicly known clause sizes, we mean that the players are only allowed to deviate to clauses that have the same number of literals as their private clauses.

We fix the reported and private clauses of the last four players $p_{n+1\leq i\leq n+4}$ in all instances to the following:

$$c_{n+1} = x, \;\; c_{n+2} = \neg x, \;\; c_{n+3} = y, \;\; c_{n+4} = \neg y.$$

Consider the following 7 clauses:

$$c^1 = x \vee \neg y, \;\; c^2 = \neg x \vee y \;\; c^3 = \neg x \vee \neg y,$$

$$c^4 = x, \quad c^5 = \neg x, \quad c^6 = y, \quad c^7 = \neg y.$$

Consider 7 instances $\mathcal{I}_j$ $(1 \leq j \leq 7)$, in instance $\mathcal{I}_j$ all of the first n players $p_{1\leq i\leq n}$ report clause c^j. Suppose that $\mathcal{M}$ is a deterministic GSP mechanism that provides an approximation ratio of ε.

First, we prove two auxiliary lemmas.

Lemma 3.24. *For every instance $\mathcal{I}_j$ $(1 \leq j \leq 7)$, mechanism $\mathcal{M}$ chooses an assignment that agrees*[3] *with clause c^j on at least one literal.*

Proof. Any assignment a for instance $\mathcal{I}_j$ that does not satisfy clause c^j satisfies at most 4 clauses. As c^j is reported by more than 4 players in instance $\mathcal{I}_j$, the optimum must satisfy that clause, and therefore satisfy at least n clauses. Hence, if $\mathcal{M}$ chooses a, the provided approximation ratio is less than or equal to $\frac{4}{n}$ and strictly less than $\frac{4}{4/\varepsilon} = \varepsilon$. Therefore, $\mathcal{M}$ has to choose an assignment that agrees with clause c^j on at least one literal for instance $\mathcal{I}_j$ (for every $1 \leq j \leq 7$). □

Lemma 3.25. *For instances $\mathcal{I}_j$ $(1 \leq j \leq 3)$, the assignment chosen by $\mathcal{M}$ must agree with c^j on both literals.*

Proof. Assume the players report truthfully, and their reported clauses match the clauses of instance $\mathcal{I}_1$. From Lemma 3.24, we know that the chosen assignment must agree with c^1 on at least one literal, so mechanism $\mathcal{M}$ can not choose assignment a_2. We claim that $\mathcal{M}$ can not choose assignments a_1 and a_4 either. Suppose $\mathcal{M}$ chooses a_1. Consider a coalition P_{coal} consisting of players $p_{1\leq i\leq n}$ and p_{n+1}. Notice that p_{n+1} has utility 0 before the deviation. Let all of P_{coal}'s players deviate to clause c^4. After the deviation the new set of reported clauses matches $\mathcal{I}_4$. By Lemma 3.24, for instance $\mathcal{I}_4$ mechanism $\mathcal{M}$ chooses either a_3 or a_4. Thus, after the deviation player p_{n+1} becomes satisfied, and gains by the deviation. Observe, that all of the other players of the coalition remain satisfied after the deviation.

[3] An assignment agrees with a given literal, if that literal is satisfied by the assignment.

Hence, if $\mathcal{M}$ chooses a_1 for instance $\mathcal{I}_1$, coalition P_{coal} can gain by deviating. Therefore, $\mathcal{M}$ can not choose a_1 for instance $\mathcal{I}_1$. The same reasoning can be used to show that $\mathcal{M}$ can not choose a_4 for instance $\mathcal{I}_1$ either.

Thus, for instance $\mathcal{I}_1$ mechanism $\mathcal{M}$ has to choose a_3, that is an assignment that agrees on both literals with c^1.

We do not prove the claim for instances $\mathcal{I}_2$ and $\mathcal{I}_3$, as the proofs for those instances follow the same structure. □

Now suppose the players report truthfully, and the set of reported clauses matches instance $\mathcal{I}_3$. Consider a coalition P_{coal} consisting of players $p_{1\leq i\leq n}$ and p_{n+1}. From Lemma 3.25, we know that for instance $\mathcal{I}_3$ mechanism $\mathcal{M}$ chooses assignment a_1. Assume players $p_{1\leq i\leq n}$ deviate to clause c^1, such that the new set of reported clauses matches $\mathcal{I}_1$. By Lemma 3.25, for instance $\mathcal{I}_1$ mechanism $\mathcal{M}$ chooses a_3. However, a_3 satisfies the previously unsatisfied player p_{n+1}, and all of the other players of coalition P_{coal}. Thus, P_{coal} gains by this deviation. Therefore, $\mathcal{M}$ is not group-strategy proof, contradicting our initial assumption. Hence, there is no deterministic group-strategy proof mechanism providing approximation ratio of ε (for any constant $\varepsilon > 0$)[4]. □

Theorem 3.26. *There is no deterministic group-strategy proof mechanism providing an approximation ratio of* $1/2 + \varepsilon$ *(for any constant* $\varepsilon > 0$*) for Max-1-SAT.*

Proof. Consider an instance $\mathcal{I}$ with $k > \lceil \frac{1}{2\varepsilon} + 1 \rceil$ variables $X = \{x_1, \ldots, x_k\}$ and $n = 2km$ ($m \in \mathbb{Z}^+$) players. There are $2k$ possible singletons. Assume for every singleton clause c there are exactly m players having c as their private clause. Consider a deterministic GSP mechanism $\mathcal{M}$. The assignment a chosen by $\mathcal{M}$ for $\mathcal{I}$ satisfies $km = n/2$ players[5]. Let clause c_{unsat} be a clause not satisfied by a. Suppose all players of the coalition consisting of the players not satisfied by a deviate to c_{unsat}. The new optimum has size $2(k-1)m$, however $\mathcal{M}$ has to still choose assignment a not satisfying c_{unsat}. Otherwise, $\mathcal{M}$ would not be GSP. As a satisfies km clauses, $\mathcal{M}$ provides an approximation ratio of $\frac{km}{2(k-1)m} = \frac{k}{2(k-1)}$. By the definition of k, the approximation ratio provided by $\mathcal{M}$ is strictly less than $1/2 + \varepsilon$. Thus, there is no deterministic GSP mechanism for Max-1-SAT providing an approximation of $1/2 + \varepsilon$ (for any constant $\varepsilon > 0$). □

[4] This also implies that there is no deterministic GSP mechanism with an approximation ratio of ε (for any constant $\varepsilon > 0$) for general Max-SAT.

[5] Every assignment satisfies exactly km players in this instance.

Corollary 3.27. *There is no deterministic group-strategy proof mechanism providing an approximation ratio of* $1/2 + \varepsilon$ *(for any constant* $\varepsilon > 0$*) for Max-Horn-SAT with publicly known clause sizes.*

The corollary follows from the fact that Max-1-SAT is a subclass of Max-Horn-SAT with publicly known clause sizes.

Lemma 3.28. *For every distribution over assignments to* n *variables* $x_1, \ldots, x_n$*, there is a clause that is satisfied with probability less than or equal to* $1/2$*.*

Proof. Let p be the probability that an assignment setting x_1 to true is chosen. Hence, a positive singleton with x_1 is satisfied with probability p, and a negative singleton with x_1 is satisfied with probability $1 - p$. As $min(p, 1 - p) \leq 1/2$, no distribution over assignments satisfies every clause with probability strictly larger than $1/2$. □

Mechanism 3.29. *Set every variable to one of the values* $\{true, false\}$ *with uniform probability.*

Theorem 3.30. *Mechanism 3.29 is a GSP mechanism that provides an approximation ratio of* $1/2$ *for general Max-SAT that allows tautologies. There is no GSP mechanism that provides an approximation ratio of* $1/2 + \varepsilon$ *(for any constant* $\varepsilon > 0$*) for this class.*

Proof. In Lemma 3.28, we proved that there is no distribution that satisfies every clause with probability strictly larger than $1/2$. Hence, by Theorem 2.17, there is no mechanism that provides an approximation ratio of $1/2+\varepsilon$ (for any constant $\varepsilon > 0$). A disjunctive clause is satisfied if at least one of its literals is satisfied. As Mechanism 3.29 satisfies every literal with probability $1/2$, it achieves an approximation ratio of $1/2$ for general Max-SAT. The mechanism is GSP, as the distribution from which it chooses an assignment does not depend on the reports of the players. □

This means that this simple mechanism achieves the best possible approximation ratio for Max-SAT with tautologies.

3.2 Deterministic SP and GSP mechanisms

In this section, we describe a deterministic GSP mechanism for Max-SAT that provides an approximation ratio of $\frac{1}{n}$, and a deterministic mechanism that provides an approximation ratio of $1/2$ that is GSP for Max-1-SAT and Max-Horn-2-SAT with publicly known clause sizes, and SP for general

Max-SAT.

In the remainder of this section, we denote the assignment that sets all variables to true by a_{true}, and the assignment that sets all variables to false by a_{false}.

Mechanism 3.31 Deterministic GSP mechanism for Max-SAT

Require: instance $\mathcal{I} = \langle X, C \rangle, |X| = n, |C| = m$

```
if w_{a_true}(C) = m then
    return a_true
else
    return a_false
end if
```

Theorem 3.32. *Mechanism 3.31 is a group-strategy proof mechanism for Max-SAT providing an approximation ratio of $\frac{1}{n}$. This bound is tight.*

Proof. Observe that the two possible assignments are each others complements, therefore, every clause is satisfied by at least one of them. As a_{false} is chosen only in case there is at least one clause not satisfied by a_{true}, the chosen assignment satisfies at least one clause. Therefore, the mechanism must provide an approximation of $\frac{1}{n}$.

Consider an instance with n players and two variables x, y. The first player reports $c_1 = \neg x$, all other players report $c_i = y$. Mechanism 3.32 sets both x and y to false, satisfying one clause. The optimal assignment sets x to false and y to true, satisfying n players. Thus, the mechanism provides an approximation ratio of at most $\frac{1}{n}$. Hence, the bound on the approximation ratio is tight.

Recall that the mechanism chooses a_{false} if and only if there is at least one player not satisfied by a_{true}. Hence, if a_{true} is chosen, it is guaranteed that every player is satisfied. Thus, the only case where a deviation can be gainful is when a_{false} is chosen in case of truthful reporting and there is at least one player not satisfied by it. The only clauses not satisfied by a_{false} are clauses consisting of only positive literals. The mechanism chooses a_{true} only in case there are no players with clauses consisting of only negative literals. Thus, the only deviations after which the mechanism changes to a_{true} are the ones where every player p_i with private clause c_i consisting of only negative literals deviates to a clause c_i' containing at least one positive literal. However, as the private clauses of these players are not satisfied by a_{true}, such a deviation would decrease their utilities. Hence, there is no (strictly) gainful deviation for any coalition, and the mechanism is group-strategy proof. □

Mechanism 3.33 Deterministic mechanism for Max-SAT

Require: instance $\mathcal{I} = \langle X, C\rangle, |C| = m$

if $w_{a_{false}}(C) \geq {}^{m}/_{2}$ **then**
  return a_{false}
else
  return a_{true}
end if

Theorem 3.34. *Mechanism 3.33 is a deterministic GSP mechanism for Max-1-SAT and Max-Horn-SAT with publicly known clause sizes, and a SP mechanism for general Max-SAT. The mechanism provides an approximation ratio of* $1/2$.

Proof. To see that the mechanism provides an approximation ratio of $1/2$, notice that the mechanism always chooses an assignment that satisfies at least half of all clauses. It is guaranteed that one of the assignments (or both) satisfies at least half of all clauses, as they are each other's complements.

To prove group-strategy proofness for Max-Horn-SAT with publicly known clause sizes (and implicitly for Max-1-SAT), notice that any Horn-clause with size of at least 2 is satisfied by a_{false}. The only clauses that are not satisfied by a_{false} are positive singletons. Hence, a_{true} is chosen only in case more than half of all clauses are positive singletons. If the mechanism chooses a_{false} in case of truthful reporting, any deviation after which the mechanism chooses a_{true} must have players with negative singleton private clauses deviate to positive singletons. However, as their private clauses are not satisfied by a_{false}, they would lose by such a deviation. Analogously, any deviation after which the mechanism changes from a_{true} to a_{false} has players with positive singleton private clauses deviate to negative singletons (recall, that every clause with a size of at least two satisfies a_{false}, hence we do not need to consider those). However, such a deviation would make them lose as well. Thus, this mechanism is indeed GSP for Max-Horn-SAT with publicly known clause sizes (and implicitly for Max-1-SAT).

Finally, to see that it is a strategy-proof mechanism for general Max-SAT, notice that the only deviations after which the mechanism chooses a different assignment are those after which the number of clauses satisfied by a_{false} drops below (or exceeds) half of the total number of clauses. However, a player satisfied (resp. not satisfied) by a_{false} can not increase (resp. decrease) the number of clauses satisfied by a_{false} by deviating. Thus, the mechanism is indeed SP for general Max-SAT. □

Alternativly, an enumerative mechanism with two complementary clauses and any tie-breaking rule also yields a polynomial deterministic SP mechanism for general Max-SAT providing an approximation ratio of $1/2$. Such a mechanism is equivalent to the mechanism described by O'Connell and Stearns [16].

3.3 Randomized mechanism

In this section, we present a randomized GSP mechanism for Max-SAT with publicly known clause sizes.

Mechanism 3.35. *Let a_{rand} be an assignment chosen with uniform probability from the set of all possible assignments, and a_{sing} be an assignment chosen by Mechanism 2.11 only based on the singleton clauses. Choose a_{rand} with probability $4/5$, and a_{sing} with probability $1/5$.*

Theorem 3.36. *Mechanism 3.35 is a polynomial time GSP mechanism for Max-SAT with publicly known clauses sizes providing an approximation ratio of $\frac{3-\varepsilon}{5}$ (for any constant $\varepsilon > 0$).*

Proof. Let s be the total number of singletons satisfiable by a single assignments, and let d be the total number of non-singletons. By the definition of Mechanism 2.11, assignment a_{sing} satisfies in expectation at least $(1-\varepsilon)\cdot s$ clauses. The randomly chosen assignment a_{rand} satisfies in expectation at least $\frac{s}{2}+\frac{3d}{4}$ clauses. Hence, the expected number of clauses satisfied by Mechanism 3.35 is at least

$$\frac{1-\varepsilon}{5}\cdot s+\frac{4}{5}\cdot\left(\frac{s}{2}+\frac{3d}{4}\right)=\frac{3-\varepsilon}{5}\cdot s+\frac{3}{5}\cdot d\geq\frac{3-\varepsilon}{5}\cdot(s+d).$$

As the optimal assignment satisfies at most $(s+d)$ clauses, Mechanism 3.35 provides indeed an approximation ratio of $\frac{3-\varepsilon}{5}$ (for any constant $\varepsilon > 0$).

Notice that both a_{sing} and a_{rand} are computable in polynomial time, hence, Mechanism 3.35 runs in polynomial time.

Observe, that only a_{sing} depends on the reports of the players. As it depends only on the singleton clauses, and Mechanism 2.11 is GSP, Mechanism 3.35 must be GSP as well. □

By choosing a_{sing} optimally instead of using the GSP $(1-\varepsilon)$-approximation mechanism, we get a polynomial time SP (instead of GSP) mechanism providing a clean approximation ratio of $\frac{3}{5}$.

Chapter 4

Max-Cut and Max-DiCut

In this chapter, we show hardness results and describe a few mechanisms for Max-Cut and Max-DiCut. First, we show that there is no deterministic group-strategy proof mechanism that provides a non-zero constant approximatio ratio for either problem. Second, we present a deterministic GSP mechanism that provides an approximation ratio of $1/n$ for Max-DiCut. Third, we prove that every oblivious algorithm[1] is a polynomial-time randomized SP mechanism for Max-DiCut. Finally, we construct a deterministic SP mechanism for Max-Cut and Max-DiCut by derandomizing a simple randomized algorithm that chooses the partition for every vertex with uniform probability using a pairwise independent distribution.

In contrast to the previous chapters, we use the graph theoretical notation in this chapter. That is, players report edges, instead of clauses, the variables are replaced by vertices, the mechanism chooses a cut instead of an assignment, and the players are satisfied if and only if their private edge is cut by the chosen cut.

4.1 Hardness results

In this section, we show that there is no deterministic GSP mechanism for Max-Cut and Max-DiCut that provides a non-zero approximatio ratio.

Theorem 4.37. *There is no deterministic GSP mechanism for Max-DiCut that provides an approximation ratio of ε (for any constant $\varepsilon > 0$).*

[1] In an oblivious algorithm the partition for every vertex v depends only on the degrees of that vertex. The partition for every vertex is chosen independently of the other vertices.

Proof. Let n be an even integer larger than 4 satisfying the following conditions:

$$\frac{4}{n^2 - n - 2} > \varepsilon,$$
$$\frac{n^2 - n - 2}{2} \geq \frac{n^2}{4}.$$

Let $G = (V, E_0)$ be a complete directed graph with n vertices and $m = n^2 - n$ edges. All of the instances considered in this proof have m players. Consider an instance $\mathcal{I}_0$ where each edge of G is reported by exactly one player.

Observation 4.38. *For every pair of non-trivial directed cuts*[2] C_a, C_b *of* G *there exists a pair of edges* e_a, e_b *such that* e_a *is cut only by* C_a *but not by* C_b*, and* e_b *is cut only by* C_b *but not by* C_a*.*

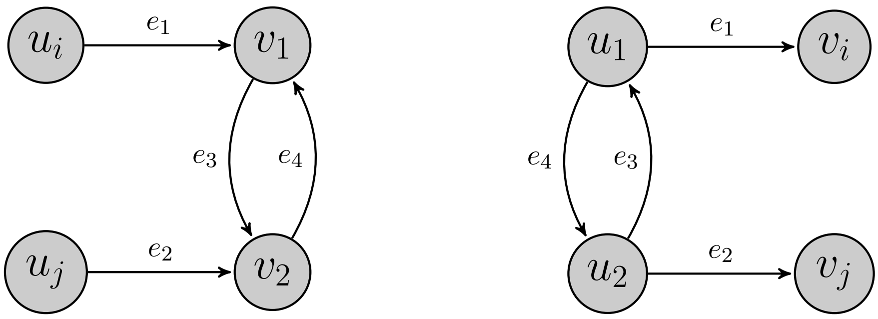
Consider a deterministic GSP mechanism $\mathcal{M}$ with an approximation ratio of ε and let $C_0 = (V_0, V_1)$ be the cut that $\mathcal{M}$ chooses for $\mathcal{I}_0$.

Claim 4.39. *For every instance* $\mathcal{I}$ *with* m *players where every edge* $e \in E_0$ *that is cut by* C_0 *is reported by at least one player, mechanism* $\mathcal{M}$ *chooses cut* C_0*.*

Proof. Suppose $\mathcal{M}$ chooses a cut C different from C_0 for such an instance $\mathcal{I}$. From Observation 4.38 we know that there is at least one edge $e \in E_0$ cut by C and not cut by C_0. Let P_{coal} be a coalition consisting of all players of $\mathcal{I}_0$ whose reported edges are not cut by C_0. Observe that there is a deviation for coalition P_{coal} from truthful reporting, such that before the deviation the reported clauses of all players (including those outside the coalition) match $\mathcal{I}_0$ and after the deviation the reported clauses (again including the players outside the coalition) match $\mathcal{I}$. As there exists at least one player $p \in P_{coal}$ whose private edge is cut by C but not by C_0, and none of P_{coal}'s players' private edges are cut by C_0, the coalition gains by such a deviation. As $\mathcal{M}$ is GSP, it must choose C_0 for every instance $\mathcal{I}$ where every edge cut by C_0 is reported by at least one player. □

If $|V_1| > 1$, let v_1, v_2 be two vertices from the set V_1, and let u_i, u_j be two vertices from the set V_0 (u_i and u_j may refer to the same vertex). And denote four directed edges $e_1 = (u_i, v_1)$, $e_2 = (u_j, v_2)$, $e_3 = (v_1, v_2)$, and $e_4 = (v_2, v_1)$. For the case where $|V_1| = 1$ (and $|V_0| > 1$), we use a slightly different definition. Let u_1, u_2 be two vertices from the set V_0, and let v_i, v_j be two vertices from the set V_1 (v_i and v_j may refer to the same vertex). Denote four directed edges $e_1 = (u_1, v_i)$, $e_2 = (u_2, v_j)$, $e_3 = (u_2, u_1)$, and $e_4 = (u_1, u_2)$.

[2] A non-trivial cut is a cut $C = (V_0, V_1)$, where both partitions V_0, V_1 are non-empty.



(a) Graph for the case $|V_1| > 1$ (b) Graph for the case $|V_0| > 1$

Figure 4.1: Graphs for instance $\mathcal{I}_1$

The edges are depicted on Figure 4.1. The edges are defined such that it does not matter for the remainder of the proof whether $|V_1| > 1$ or $|V_1| = 1$.

Let $\mathcal{I}_1$ be an instance where one player reports each one of the edges e_1 and e_2, and exactly $\frac{n^2-n-2}{2}$ players report e_3 and e_4. For the remainder of this proof, assume that $\mathcal{I}_1$ represents the players' private clauses.

Let C_1 be the cut chosen by $\mathcal{M}$ for instance $\mathcal{I}_1$. As the optimum cuts e_3 or e_4, the size of the optimum is at least $\frac{n^2-n-2}{2}$. Any cut that does not cut e_3 and e_4, cuts at most 2 edges. Therefore, the approximation ratio provided by such a cut is $\frac{2}{\frac{n^2-n-2}{2}} < \varepsilon$. Hence, C_1 chosen by $\mathcal{M}$ for $\mathcal{I}_1$ must cut either e_3 or e_4.

Notice that e_3 and e_4 can not be cut by the same cut, thus, there are at least $\frac{n^2-n-2}{2}$ players whose edges are left uncut. Furthermore, any cut that cuts e_4 does not cut e_2 and e_3, and any cut that cuts e_3 does not cut e_1 and e_4. W.l.o.g., assume C_1 cuts e_3. Let P_{coal} be a coalition consisting of all of the players reporting e_4 and e_1. Notice that none of P_{coal}'s players have their edges cut by c_1. Suppose the players of P_{coal} deviate in a way that after the deviation every edge $e \in E_0$ cut by C_0 is reported by at least one player. There is at least one such deviation as the number of players in the coalition is larger than the number of edges cut by C_0[3]. By Claim 4.39, for an instance with such reports $\mathcal{M}$ chooses cut C_0. As the private clause of the player reporting e_1 before the deviation become cut after the deviation, the coalition gains by deviating. Hence, for $\mathcal{M}$ to be GSP, it must choose C_0 for

[3]The number of edges cut by C_0 is at most $n^2/4$, the size of P_{coal} is at least $\frac{n^2-n-2}{2}$. n satisfies $\frac{n^2-n-2}{2} \geq \frac{n^2}{4}$ by definition.

instance $\mathcal{I}_1$. However, C_0 cuts 2 edges in case of $\mathcal{I}_1$, whereas the optimal cut cuts at least $\frac{n^2-n-2}{2}$ edges. Thus, the approximation ratio provided by the mechanism is strictly less than ε, contradicting our initial assumption. □

Theorem 4.40. *There is no deterministic GSP mechanism for Max-Cut that provides an approximation ratio of ε (for any constant $\varepsilon > 0$).*

Proof. Let n be an integer larger than $5/\varepsilon$. Consider 6 instances $\mathcal{I}_i$ $(1 \le i \le 6)$ with 4 vertices and $5+n$ players each. Notice that for graphs with 4 vertices, 6 different edges are possible (self-loops excluded). We refer to these edges by e_i $(1 \le i \le 6)$. In instance $\mathcal{I}_i$ edge e_i is reported by n players, the other 5 edges are reported by only one player each. Let $\mathcal{M}$ be a deterministic group-strategy proof mechanism providing an approximation ratio of ε. Denote the cut chosen by $\mathcal{M}$ for instance $\mathcal{I}_i$ by C_i.

Observe, that for every instance $\mathcal{I}_i$, the chosen cut C_i has to cut edge e_i. Every cut that does not cut e_i cuts at most 5 reported edges. The optimum cuts e_i, and cuts, therefore, at least n reported edges. As $5/n < \varepsilon$, the approximation ratio provided by a mechanism choosing such a cut is less than ε. Hence, C_i must cut e_i.

Next, we show that $\delta(C_i) \cap \delta(C_j) \neq \varnothing$ implies $C_i = C_j$.[4] Assume this is not the case, and there exist a pair of cuts C_i, C_j with $\delta(C_i) \neq \delta(C_j)$, and an edge $e^* \in \delta(C_i) \cap \delta(C_j)$. The following three cases are possible (the case where $e^* = e_j$ is not considered explicitly as it is equivalent to one of the other cases after swapping the definitions of e_i and e_j):

1. $e^* = e_i$, and there is an edge e such that $e \in \delta(C_j)$ and $e \notin \delta(C_i)$. Notice, that cut C_j cuts both e_i and e, but C_i cuts only e_i. Consider a coalition P_{coal} consisting of all players reporting edges e or e_i. Suppose in case of truthful reporting the reported edges match instance $\mathcal{I}_i$. After a deviation by $n-1$ players from reporting e_i to reporting e_j, the new reports match instance $\mathcal{I}_j$. As cut C_j satisfies every player of P_{coal}, the coalition gains by such a deviation. As $\mathcal{M}$ is GSP, such an e^* does not exist .

2. $e^* = e_i$, and there is no edge e that satisfies $e \in \delta(C_j)$ and $e \notin \delta(C_i)$. As $\delta(C_j) \neq \delta(C_i)$, there must exist an edge e satisfying $e \in \delta(C_i)$ and $e \notin \delta(C_j)$. Therefore, it must be that $\delta(C_j) \subset \delta(C_i)$. Consider a coalition P_{coal} consisting of all players reporting edges e or e_j. Suppose in case of truthful reporting the reported edges match instance $\mathcal{I}_j$. After

[4] $\delta(C)$ is the set of edges cut by C.

a deviation by $n-1$ players from reporting e_j to reporting e_i, the new reports match instance $\mathcal{I}_i$. As $e \in \delta(C_i)$, $e \notin \delta(C_j)$ and $\delta(C_j) \subset \delta(C_i)$, the coalition gains by such a deviation. Hence, there does not exist such an e^*.

3. $e^* \neq e_i$ and $e^* \neq e_j$. In this case, there must exist a cut C_k with $e^* = c_k$. As $\delta(C_i) \neq \delta(C_j)$, $\delta(C_k)$ must be different from $\delta(C_i)$ or $\delta(C_j)$ (or both). By relabeling i, j, k we get to the same scenario as in case 1 or case 2. Therefore, there is no such e^*.

As there is no edge e^* for which $e^* \in \delta(C_i) \cap \delta(C_j)$ (when $\delta(C_i) \neq \delta(C_j)$), it follows that $\delta(C_i) \cap \delta(C_j) \neq \varnothing$ implies $\delta(C_i) = \delta(C_j)$. Thus, any pair of cuts chosen by the mechanism must either be completely disjoint or equal, while still satisfying $e_i \in C_i$.

The condition that $e_i \in C_i$ implies that every edge must be cut by at least one of the cuts. For a complete graph with 4 vertices, every cut cuts either 3 or 4 edges. As the cuts must be completely disjoint (or equal) and every edge must be cut by at least one cut, every C_i must be equal to some cut of size 3 (notice that setting any C_i to a cut of size 4 immediately violates the disjointness property). However, it is not possible to assign cuts of size 3 to every C_i in a way that satisfies both disjointness and $e_i \in C_i$. □

Theorem 4.41. *There is no distribution over cuts (resp. directed cuts) for graphs with n vertices where the minimum probability of an edge being cut is larger than $\frac{n}{2(n-1)} = 1/2 + o(1)$ (resp. $\frac{n}{4(n-1)} = 1/4 + o(1)$).*

Proof. Observe that any cut for a graph with n vertices cuts at most $n^2/4$ edges. A complete graph with n vertices has $\frac{n^2-n}{2}$ edges, and a complete directed graph with n vertices has $n^2 - n$ edges. Therefore, the minimum number of cuts (resp. directed cuts) required to cut every edge is at least $\frac{\frac{n^2-n}{2}}{n^2/4} = \frac{2(n-1)}{n}$ (resp. $\frac{n^2-n}{n^2/4} = \frac{4(n-1)}{n}$). □

4.2 Deterministic GSP mechanisms

In this section we describe a deterministic GSP mechanism for Max-DiCut.

Mechanism 4.42 Deterministic GSP mechanism for Max-DiCut

Define C_i (for $1 \leq i \leq n$) as the cut $(\{v_i\}, V - \{v_i\})$.
return the C_i with the minimal i cutting at least one edge.

Theorem 4.43. *Mechanism 4.42 is a polynomial time deterministic group-strategy proof mechanism providing an approximation ratio of $\frac{1}{n}$ for Max-DiCut. This bound is tight.*

Proof. The mechanism is, clearly, deterministic. It is polynomial, as it considers only a polynomial number of cuts.

Notice, that for every pair of cuts C_i, C_j ($1 \leq i, j, \leq n$ and $i \neq j$) the sets of cut edges $\delta(C_i), \delta(C_j)$ are pairwise disjoint. Assume that for an instance $\mathcal{I}$ Mechanism 4.42 chooses cut C_i. By the definition of Mechanism 4.42, there is no cut C_j with $j < i$ that satisfies at least one player. Therefore, we only need to consider deviations from $\mathcal{I}$ after which the mechanism chooses a cut C_j with $j > i$. Such a deviation requires all players with private edges cut by C_k for every $k < j$ deviate to an edge cut by C_j. However, players with edges cut by C_i lose by such a deviation, as their private edges are not cut by C_j. Thus, such a deviation is not gainful, and the mechanism is indeed GSP.

Observe that for every possible instance the mechanism chooses a cut that cuts at least one edge, hence, for instances with n players, it must have an approximation ratio of $\frac{1}{n}$. Consider an instance where $n-1$ players report edge (v_1, v_2), and 1 player reports edge (v_0, v_2). Mechanism 4.42 chooses a cut that cuts only (v_0, v_2) (the optimal cut cuts edge (v_1, v_2) as well), therefore the bound on the approximation ratio is tight. □

4.3 Oblivious algorithms for Max-DiCut

The bias of a vertex v is defined as the fraction of its in-degree compared to its total degree:

$$bias(v) = \frac{d_{in}(v)}{d_{in}(v) + d_{out}(v)}.$$

An oblivious algorithms [15] for Max-DiCut is a randomized polynomial time algorithm in which the probabilities of putting a vertex into V_0 or V_1 depend only on the bias of the vertex and are independent of the other vertices. We say an oblivious algorithm is monotone if the probability of a vertex being put into partition V_0 is non-decreasing with increasing bias. Every algorithm described by Feige and Jozeph [15] is monotone.

Theorem 4.44. *A monotone oblivious algorithm is a strategy-proof mechanism for Max-DiCut.*

Proof. Consider a player p_i with a private edge (u,v). After any deviation by p_i from truthful reporting the following conditions hold:

$$d'_{out}(u) \leq d_{out}(u),$$
$$d'_{out}(v) \geq d_{out}(v),$$
$$d'_{in}(v) \leq d_{in}(v),$$
$$d'_{in}(u) \geq d_{in}(u).$$

Thus, $bias(u)' \geq bias(u)$, $bias(v)' \leq bias(v)$, and $p'[u \in V_0, v \in V_1] \leq p[u \in V_0, v \in V_1]$. Hence, no deviation is gainful for p_i and the mechanism is indeed strategy proof. □

The best known oblivious algorithm provides an approximation ratio of 0.483, and the best possible approximation ratio attainable by an oblivious algorithm is strictly less than 0.4899 [15].

4.4 Simple polynomial time randomized GSP and deterministic SP mechanisms

In this section, we present a simple randomized GSP mechanism for Max-Cut and Max-DiCut, and show that this mechanism can be derandomized to provide a deterministic SP mechanism with a matching approximation ratio.

Mechanism 4.45 Simple randomized mechanism

Pick n random bits $b_1, \ldots, b_n$.
return cut $C = (V_0, V_1)$ with vertex v_i being in set V_{b_i}.

Theorem 4.46. *Mechanism 4.45 is group-strategy proof and provides a* $1/2$ *approximation for Max-Cut and a* $1/4$ *approximation for Max-DiCut.*

Proof. As the cut is chosen independently of the reported edges, no deviations are gainful, and the mechanism is GSP.

The expected number of edges cut in the undirected case is:

$$E[|C|] = \sum_{(v_i,v_j)\in E} Pr[v_i \neq v_j] = \sum_{(v_i,v_j)\in E} \frac{1}{2} = \frac{|E|}{2}.$$

Analogously, in the directed case it is:

$$E[|C|] = \sum_{(v_i,v_j)\in E} Pr[v_i = 0, v_j = 1] = \sum_{(v_i,v_j)\in E} \frac{1}{4} = \frac{|E|}{4}.$$

As the number of edges cut by the optimum is always less than or equal to $|E|$, the mechanism indeed provides a $1/2$-approximation for Max-Cut and $1/4$-approximation for Max-DiCut. □

The approximation ratio of Mechanism 4.45 relies only on the fact that for every edge $e = (v_i, v_j)$ the partitions for the endpoints are chosen uniformly and independently of each other. Thus, the only requirement on the probability distribution is that the random bits $b_1, \ldots, b_n$ are uniformly distributed and pairwise independent.

It is possible to construct 2^{l-1} pairwise independent and uniformly distributed random variables (bits) from l independent random variables (for more details see Section 2 in [17]). Thus, $\lceil \log_2 n + 1 \rceil$ uniform random bits are needed to construct n pairwise independent uniform random bits. As there is only a linear number of possible assignments to these logarithmic number of bits, such a distribution can generate only a linear number of different cuts. It is possible to generate and enumerate all of these in polynomial time. As the expected number of edges cut by a cut taken randomly from these is $1/2$ for Max-Cut and $1/4$ for Max-DiCut, there must exist at least one cut in the support of this distribution with a matching size.

Mechanism 4.47 Enumerative mechanism based on pairwise-independence

1: Let X_n be a pairwise independent distribution over n bits with polynomial sized support R_n
2: Order R_n lexicographically
3: **return** lexicographically largest cut with maximal size from R_n

Theorem 4.48. *Mechanism 4.47 is a polynomial time deterministic strategy proof mechanism providing an approximation ratio of $1/2$ for Max-Cut and $1/4$ for Max-DiCut.*

Proof. The mechanism is an enumerative mechanism with $A(n) = R_n$ and lexicographic ordering. Thus, by Theorem 2.4, it is strategy proof and deterministic.

As the expected number of edges cut by a random cut (resp. directed cut) generated by X_n is $|E|/2$ (resp. $|E|/4$), its support must contain at least one cut of size $|E|/2$ and at least one directed cut of size $|E|/4$. Therefore, the mechanism provides a $1/2$-approximation for Max-Cut, and $1/4$-approximation for Max-DiCut. □

Chapter 5

Conclusions

In this thesis, we presented several results for multi-agent models of boolean constraint satisfaction problems. However, many interesting questions about these models are still open. The simplest open problem in our opinion is whether a deterministic group-strategy proof mechanism for Max-Cut with a non-zero non-constant approximation ratio exists. We strongly believe there does not exist such a mechanism. This may sound counter-intuititve, as such a mechanism exists for Max-DiCut, which can be viewed as a generalization of Max-Cut. A second interesting problem is finding a polynomial time randomized mechanism with an approximation ratio better than that of the simple uniform randomized mechanism setting each variable to one of the values $\{true, false\}$ with uniform probability for Max-SAT and Max-Cut. The fact that such mechanisms exist for Max-DiCut provides a reason to believe in the existence of such mechanisms. Another open problem worth looking at is proving upper bounds for polynomial time randomized (both SP and GSP) mechanisms and polynomial time deterministic SP mechanisms. The known hardness results for approximation algorithms for these problems extend immediately to our model, but it might be possible to get stronger bounds using both truthfulness and complexity arguments. None of the upper bounds proven in this thesis take computation time into account. Also, it is unclear if there are polynomial time deterministic mechanisms providing a better approximation ratio than the derandomization of the simple uniform randomized mechanism. All of the polynomial time deterministic SP mechanisms described in this thesis are enumerative mechanisms. We do not know if there exist any SP non-enumerative mechanisms providing better approximation ratios.

Bibliography

[1] Ariel D. Procaccia and Moshe Tennenholtz. Approximate mechanism design without money. In *Proceedings of the 10th ACM Conference on Electronic Commerce*, EC '09, pages 177–186, New York, NY, USA, 2009. ACM.

[2] Sylvia Boicheva. Mechanism design without money. Master's thesis, 2012.

[3] Shaddin Dughmi and Arpita Ghosh. Truthful assignment without money. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 325–334. ACM, 2010.

[4] Stephen A Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158. ACM, 1971.

[5] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete problems. In *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, STOC '74, pages 47–63, New York, NY, USA, 1974. ACM.

[6] Mihalis Yannakakis. On the approximation of maximum satisfiability. In *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '92, pages 1–9, Philadelphia, PA, USA, 1992. Society for Industrial and Applied Mathematics.

[7] Adi Avidor, Ido Berkovitch, and Uri Zwick. Improved approximation algorithms for max nae-sat and max sat. In *Approximation and Online Algorithms*, pages 27–40. Springer, 2006.

[8] Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.

[9] Richard M Karp. *Reducibility among combinatorial problems.* Springer, 1972.

[10] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[11] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775. ACM, 2002.

[12] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM Journal on Computing*, 37(1):319–357, 2007.

[13] Luca Trevisan. Parallel approximation algorithms by positive linear programming. *Algorithmica*, 21(1):72–88, 1998.

[14] Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the max 2-sat and max di-cut problems. In *Integer Programming and Combinatorial Optimization*, pages 67–82. Springer, 2002.

[15] Uriel Feige and Shlomo Jozeph. Oblivious algorithms for the maximum directed cut problem. *Algorithmica*, pages 1–20, 2010.

[16] Thomas C. O'Connell and Richard E. Stearns. Polynomial time mechanisms for collective decision making. In Simon Parsons, Piotr Gmytrasiewicz, and Michael Wooldridge, editors, *Game Theory and Decision Theory in Agent-Based Systems*, volume 5 of *Multiagent Systems, Artificial Societies, and Simulated Organizations*, pages 197–216. Springer US, 2002.

[17] Michael Luby and Avi Wigderson. *Pairwise independence and derandomization*. 1995.