On the Optimality of Some Semidefinite Programming-Based Approximation Algorithms under the Unique Games Conjecture

A Thesis presented

by

Seth Robert Flaxman

to

Computer Science

in partial fulfillment of the honors requirements

for the degree of

Bachelor of Arts

Harvard College

Cambridge, Massachusetts

April 1, 2008

Contents

Chapter 1

Introduction

Computational complexity theory is the branch of computer science concerned with investigating the efficiency of algorithms for solving computational problems. The fundamental question in computational complexity theory is: how hard is it for a computer to solve instances of a given problem? Much of the work in the 1960's and 1970's involved problems with solutions of the form "yes" or "no," so called *decision problems.* Among these problems are a huge number of important ones including the boolean satisfiability problem, 3-SAT, CLIQUE, and VERTEX COVER. Decision problems often have closely related versions called *optimization problems*, which ask for a numeric solution rather than "yes" or "no." For example, the decision problem CLIQUE asks if a certain graph G has a clique of size at least k (for a certain value of k). The optimization version, MAXIMUM CLIQUE, asks instead for the *size* of the largest clique of G.

The concept of NP-hardness[1] makes precise the question of how hard it is for a computer to solve instances of a given problem. Roughly speaking, NP-hard problems are those for which no efficient algorithm could possibly exist unless $P = NP$. Many important optimization and decision problems are NP-hard. Unlike decision problems, optimization problems are natural candidates for polynomial time *approximation algorithms.* These efficient algorithms get provably "close" to the optimal solution. In the search for approximation algorithms, a parallel route to devising algorithmic results is the proof of *inapproximability*

[1] We do not define NP-hardness formally. See any introductory computer science text for a definition.

results. An inapproximability result states that it is NP-hard to find an efficient algorithm that achieves an approximate solution better than some bound.

In this work, we focus on the connections between approximation algorithms and inapproximability results. For a particular optimization problem, there is the direct connection: approximation algorithms give lower bounds on how well we can do while inapproximability results give upper bounds. We look for a deeper connection in the case of two techniques: semidefinite programming, which has proved very successful in devising approximation algorithms, and the Unique Games Conjecture, which has led to many inapproximability results. Although on the surface these techniques seem to be unrelated, a series of recent papers suggests otherwise. Is it possible that the Unique Games Conjecture exactly captures the power of semidefinite programming? We state a conjecture formalizing this connection and investigate this conjecture for a small set of problems.

While many of the techniques for proving that a decision problem is NP-hard are often elementary and have been known for decades, proofs of most inapproximability results require more sophisticated techniques that have only been devised relatively recently. During the 1990's, the body of work relating to probabilistically checkable proofs (PCP), often called the PCP theory or PCP theorems, came as a set of breakthrough results that gained wide use in proving the inapproximability of many optimization versions of NP-hard problems. A rigorous treatment of the PCP theorems is beyond the scope of this work. During the same decade, work on approximation algorithms took a leap forward with Goemans and Williamson's introduction of the technique of randomized rounding of semidefinite programs (SDPs) [GW95]. In the next chapter we will demonstrate and prove results about approximation algorithms based on rounding SDPs.

Both directions of research on approximation—approximation algorithms and inapproximability results—have had important implications for constraint satisfaction problems (CSPs). An instance of a CSP decision problem is a set of variables subject to a set of constraints. The objective of CSP decision problem is to find an assignment to the variables that satisfies all the constraints. The natural optimization version, MAX CSP, is: given a set of constraints, find an assignment that maximizes the number of satisfied constraints.

A weighted version of this problem assigns weights to the constraints and the objective is to find an assignment that maximizes the total weight of the satisfied constraints.

CSPs generalize many problems, including the boolean satisfiability problem already mentioned. Subproblems of the boolean satisfiability problem include 3-Sat, MaxCut[2], and 2-Sat. While the decision problem 3-Sat was on Karp's original list of NP-complete problems in 1972, 2-Sat is in P, but Max 2-Sat, the optimization version, is NP-hard to approximate to within any constant factor. Håstad's proof of this fact relies on the PCP theory [Has97].

Goemans and Williamson considered some of these CSPs, including MaxCut and Max 2-Sat [GW95]. Before their work, various algorithms for MaxCut had been proposed, yet none of them achieved approximation ratios with a constant term better than $\frac{1}{2}$, the approximation ratio of the simple algorithm which randomly assigns vertices to the two different sets. Goemans and Williamson improved this approximation ratio to .879. For Max 2-Sat they improved the best known approximation ratio of $\frac{3}{4}$ (which can be found by picking a random assignment) to .879. As we will see in Section 2.2, Goemans and Williamson's technique is to create a semidefinite program which captures a relaxed version of the instance they wish to solve. After solving this SDP optimally, they apply a simple rounding technique.

Although the successes of the PCP theory have been widespread, in the special case of 2-CSPs, in which constraints are limited to acting on two variables, tight inapproximability results have been harder to come by. The best known inapproximability result for Max 2-Sat says that it is 21/22-hard to approximate [Has97], leaving a gap between the lower bound of .879 and upper bound of .955 ($\approx 21/22$).

In 2002, Khot introduced the Unique Games Conjecture (UGC) as a way of making progress on the approximability of 2-CSPs [Kho02]. The UGC is a stronger conjecture than $P \neq NP$, because $P \neq NP$ follows immediately from an unconditional proof of the UGC but

[2]Given a graph $G(V, E)$, partition V into sets S and $V - S$ such that the number of edges crossing the cut, $C(S, V - S)$, is maximized. This is a boolean satisfiability problem because the constraints are boolean constraints of the form, $(x_i \wedge \neg x_j) \vee (\neg x_i \wedge x_j)$ where x_i and x_j are vertices connected by an edge. It is easy to check that this constraint is satisfied if and only if x_i and x_j get different assignments, meaning they are placed on different sides of the cut.

the UGC does not follow immediately from $P \neq NP$. Though Khot's conjecture remains open, and little progress has been made in resolving it, it implies a number of attractive results: hardness results for 2-Linear-Equations and Not-all-equal 3-Sat [Kho02], an optimal hardness result for VertexCover [KR03], and a tight .879-hardness result for MaxCut [KKMO04], matching the approximation ratio of the Goemans-Williamson algorithm. These surprising results are not evidence for or against the Unique Games Conjecture, but they do make determining the status of the Unique Games Conjecture an interesting open problem. We also note that work on the Unique Games Conjecture has led to results which do not require the UGC, including disproving a conjecture about the embeddability of a certain metric [KV05] and an approximation algorithm for constraint satisfaction problems [Rag08].

The surprising appearance of Goemans-Williamson's constant (which we will derive from a geometric argument in Section 2.2) in the UGC-based MaxCut work suggests a connection between the Unique Games Conjecture and semidefinite programming. This is borne out by a series of other papers giving semidefinite programming-based approximation algorithms (or in many cases, semidefinite programming-based integrality gaps) that match inapproximability results, assuming the Unique Games Conjecture. As Austrin writes in the introduction to [Aus07a], there "appears to be a very strong connection between the power of the semidefinite programming paradigm for designing approximation algorithms, and the power of UGC-based hardness of approximation results."

We state a set of conjectures formalizing a connection between semidefinite programming and Unique Games. Under the Unique Games Conjecture our conjectures imply that for some class of problems, semidefinite programming gives optimal approximability results. While the status of the Unique Games Conjecture remains unresolved, proving this conjecture would explain the broad picture of previous work on Unique Games. It would also provide intuition about the implications of the Unique Games Conjecture.

After stating our conjectures, we investigate them for a small class of problems. In the specific case that we investigate, previous work by Austrin [Aus07a] has established tight UGC-based inapproximability results for Max 2-Sat and two closely related problems which we will formally define later: Balanced Max 2-Sat and Δ-Imbalanced Max 2-Sat. Austrin's work contains a surprising result under the UGC, which we wish to investigate further. We provide the first numerical calculations of the inapproximability of Δ-Imbalanced Max 2-Sat using a formula of Austrin's. We continue Austrin's line of work by proving a tight result for Δ-Imbalanced Max 2-Sat. We do so under a plausible assumption which we do not prove analytically, similar to the one in Austrin's work [Aus07a].

Our work is organized as follows: in Chapter 2 we state the background on approximation algorithms and PCP-based inapproximability results. In Chapter 3 we develop conjectures relating the power of semidefinite programming and the Unique Games Conjecture. In Chapter 4 we restate Austrin's results for Max 2-Sat and develop our own for Δ-Imbalanced Max 2-Sat. In Chapter 5 we conclude.

Chapter 2

Background

In this chapter, we formally define approximation algorithms and semidefinite programming. We describe inapproximability results and the successes of probabilistically checkable proofs in obtaining these results. We also describe the successes of the use of semidefinite programming in obtaining approximation algorithms. We state the Unique Games Conjecture (UGC) and summarize some UGC-based inapproximability results. We conclude by summarizing recent work relating the UGC to semidefinite programming.

2.1 Approximation Algorithms

A polynomial time α-approximation algorithm is a polynomial time algorithm that solves instances of a combinatorial optimization problem to within a worst-case factor α of the optimal solution for every instance. As an example, consider an instance of a profit maximization problem that has an optimal solution yielding \$20 of profit. A $\frac{1}{2}$-approximation algorithm gives a solution with a guarantee that it will yield at least \$10 of profit.

We define an approximation algorithm with approximation ratio $\alpha \leq 1$ in the case of combinatorial maximization problems (a similar definition for $\alpha \geq 1$ can be made for combinatorial minimization problems):

Definition 2.1.1. *An approximation algorithm A for a combinatorial maximization problem achieves an approximation ratio $\alpha \leq 1$ (and is called an α-approximation) if given an* *instance I with optimum value OPT, A outputs a solution with value S where* $S \geq \alpha \cdot OPT$*.* α *is called an approximation ratio because it bounds the ratio* $\frac{S}{OPT}$*.*

We now describe and analyze a prototypical approximation algorithm based on randomized rounding[1] and linear programming. Our polynomial time algorithm is as follows: first, express the problem to be solved as an integer linear program. Second, "relax" this program so that the variables are no longer restricted to integers. Third, the relaxed program is a linear program, so solve it optimally in polynomial time. Fourth, use a randomized polynomial time rounding procedure to make the solution integral, thus arriving at a feasible solution (a solution satisfying the constraints of the original integer linear program). Output this solution.

Now we want to analyze this algorithm by proving a guarantee on how far the solution is from optimal. We call the optimum value of the original problem OPT and the optimum value of the relaxed program OPT'. We are interested in the value S of the (rounded) integral solution because this is the output of our approximation algorithm. We want to find the approximation ratio α by proving a lower bound on $\frac{S}{OPT}$ but we do not usually know anything about OPT. Instead we observe $OPT' \geq OPT$ (since the relaxation of a maximization problem has value greater than or equal to the original version), so $\frac{S}{OPT'} \leq \frac{S}{OPT}$. Thus if we can prove a lower bound on $\frac{S}{OPT'}$ this will serve as our approximation ratio α. This is represented in Figure 2.1.

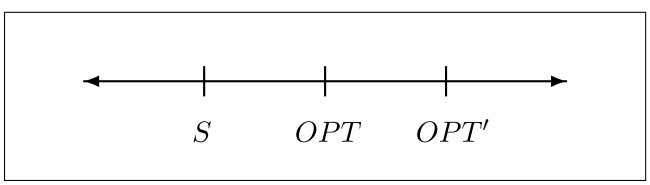

Figure 2.1: In analyzing an approximation algorithm for a maximization problem based on relaxation and rounding, OPT is the value we are trying to approximate, OPT' is the value of the relaxed (and probably infeasible) solution, and S is the value of the rounded solution. We are interested in the approximation ratio α so we find it as a lower bound $\alpha \leq \frac{S}{OPT'} \leq \frac{S}{OPT}$.

[1]Most approximation algorithms use randomness and for the purposes of our work, we will assume that randomness is always allowed.

2.2 Semidefinite Programming (SDP)

In the previous section we outlined an approximation algorithm based on relaxing an integer linear program into a linear program, solving optimally, and then rounding. Using the same idea, we create a more sophisticated approximation algorithm as follows: represent the problem as an integer quadratic program, relax it to a semidefinite program, solve it optimally, and then round.

We define a semidefinite program as follows:

$$\begin{aligned} \max\ & C \cdot X \\ \text{subject to: } & A_i \cdot X = b_i, \quad i = 1, \ldots, m, \\ & X \in S_+^n \quad \textit{i.e., } X \text{ is positive semidefinite} \\ & C, A_1, \ldots, A_m \ \text{ symmetric matrices} \end{aligned}$$

Goemans and Williamson devised the SDP-based approximation algorithm and applied it to MaxCut, Max 2-Sat, and Max DiCut [GW95]. Their techniques proved very successful and will be central to our work. We outline the proof of Goemans and Williamson [GW95] for the case of MaxCut.

Theorem 2.2.1. MaxCut *can be approximated to within a constant* $\alpha_{GW} = .879$.

Proof. Given a graph $G = (V, E)$ with $|V| = n$ and edge weights $w_{ij} = w_{ji} \geq 0$, we would like to find a set $S \subset V$ that maximizes the weight of the edges crossing from S to $V - S$, which we denote by $C(S, V - S)$. We create variables $x_1, \ldots, x_n$ for the vertices, with $x_j = 1$ if vertex j is in S and $x_j = -1$ otherwise. Then we can express this problem as an integer quadratic program:

$$\begin{aligned} \max\ & \frac{1}{2} \sum_{i<j} w_{ij}(1 - x_i x_j) \\ \text{subject to: } & x_j \in \{-1, 1\},\ j = 1, \ldots, n \end{aligned}$$

Now we relax this program to a semidefinite program, relaxing some of the constraints and allowing the objective function to take values in a larger space. This will guarantee that the relaxed program has an optimum value at least as large as that of the integer program.

Following Goemans and Williamson, we replace scalar variables x_j with vectors $\mathbf{v_j} \in \mathbb{S}^n$.

$$\max \ \frac{1}{2}\sum_{i<j} w_{ij}(1 - \mathbf{v_i} \cdot \mathbf{v_j})$$

$$\text{subject to: } \mathbf{v_j} \in \mathbb{S}^n, \ j = 1, \ldots, n$$

Now, we solve this relaxation with semidefinite programming to obtain a (nearly) optimal set of vectors $\mathbf{v_1}, \ldots, \mathbf{v_n}$. Next, we pick a vector $\mathbf{r}$ uniformly at random in $\mathbb{S}^n$. Finally, we round each vector $\mathbf{v_j}$ to a scalar x_j with value $sign(\mathbf{v_j} \cdot \mathbf{r}) \geq 0$. Equivalently, for each vector $\mathbf{v_j}$ we place vertex j in S if $\mathbf{v_j} \cdot \mathbf{r} \geq 0$.

We analyze this algorithm as follows:

Lemma 2.2.2. *For two vectors* $\mathbf{v_i}$ *and* $\mathbf{v_j}$*, the probability that the rounding places them on different sides of the cut is given by:* $\Pr[x_i \neq x_j] = \frac{\arccos(\mathbf{v_i} \cdot \mathbf{v_j})}{\pi}$.

Proof.

$$\Pr[x_i \neq x_j] = \Pr[sign(\mathbf{v_i} \cdot \mathbf{r}) \neq sign(\mathbf{v_j} \cdot \mathbf{r})] \tag{2.1}$$

$$= 2\Pr[sign(\mathbf{v_i} \cdot \mathbf{r}) \geq 0 \text{ and } sign(\mathbf{v_j} \cdot \mathbf{r}) < 0] \tag{2.2}$$

We argue geometrically, picturing $\mathbf{v_i}$, $\mathbf{v_j}$, and $\mathbf{r}$ as vectors on an n-dimensional sphere. We want to calculate $\Pr[sign(\mathbf{v_i} \cdot \mathbf{r}) \geq 0 \text{ and } sign(\mathbf{v_j} \cdot \mathbf{r}) < 0]$, the probability that $\mathbf{v_i}$ is above the random hyperplane normal to $\mathbf{r}$ and $\mathbf{v_j}$ is below this hyperplane.

The sets $A = \{\mathbf{r} : \mathbf{v_i} \cdot \mathbf{r} \geq 0\}$ and $B = \{\mathbf{r} : \mathbf{v_j} \cdot \mathbf{r} < 0\}$ are half-spheres bounded by planes $\{\mathbf{r} : \mathbf{v_i} \cdot \mathbf{r} = 0\}$ and $\{\mathbf{r} : \mathbf{v_j} \cdot \mathbf{r} = 0\}$ respectively. Note that B contains vectors pointing in the direction $-\mathbf{v_j}$ since these are the ones for which $\mathbf{v_j} \cdot \mathbf{r} < 0$.

We want to calculate the intersection of A and B. The intersection is directly proportional to the angle between $\mathbf{v_i}$ and $\mathbf{v_j}$, $\arccos(\mathbf{v_i} \cdot \mathbf{v_j})$. To find the constant of proportionality we notice that if $\arccos(\mathbf{v_i} \cdot \mathbf{v_j}) = \pi$ then $\mathbf{v_i}$ and $\mathbf{v_j}$ point in opposite directions, so the probability of $\mathbf{v_i}$ lying above the hyperplane and $\mathbf{v_j}$ below must be $\frac{1}{2}$. Equivalently, we argue that in this case A and B are the same half-sphere, so their intersection must be half the volume of the sphere.

Thus the constant of proportionality must be $\frac{1}{2\pi}$ and in general the formula is:

$$\frac{1}{2\pi} \arccos(\mathbf{v_i} \cdot \mathbf{v_j})$$

We calculate twice this probability to account for the other, equivalent case with $\mathbf{v_i}$ below the hyperplane and $\mathbf{v_j}$ above as in Equation 2.2 and arrive at what we wanted:

$$\frac{\arccos(\mathbf{v_i} \cdot \mathbf{v_j})}{\pi}$$

□

Now we want to prove a guarantee on the ratio between the expected performance of this rounding algorithm, $E[C(S, V-S)]$ and the value of the relaxed program, OPT'. As in the algorithm sketched in Section 2.1 for linear programming, this ratio will give us a bound on the approximation ratio, the ratio between the performance of this rounding algorithm and the optimum value OPT of the original instance of MaxCut.

Lemma 2.2.3. $E[C(S, V - S)] \geq .879\ OPT'$

Proof.

$$E[C(S,V-S)] = E[\frac{1}{2}\sum_{i<j} w_{ij}(1-x_ix_j)] \tag{2.3}$$

$$= \sum_{i<j} w_{ij} \Pr[x_i \neq x_j] \tag{2.4}$$

$$= \sum_{i<j} w_{ij} \frac{\arccos(\mathbf{v_i} \cdot \mathbf{v_j})}{\pi} \tag{2.5}$$

$$= \frac{1}{2}\sum_{i<j} w_{ij}(1-\mathbf{v_i}\cdot\mathbf{v_j})\frac{2}{\pi}\frac{\arccos(\mathbf{v_i}\cdot\mathbf{v_j})}{(1-\mathbf{v_i}\cdot\mathbf{v_j})} \tag{2.6}$$

$$\geq \frac{1}{2}\sum_{i<j} w_{ij}(1-\mathbf{v_i}\cdot\mathbf{v_j})\frac{2}{\pi}\min_{-1\leq t<1}\frac{\arccos(t)}{(1-t)} \tag{2.7}$$

$$= \left(\frac{1}{2}\sum_{i<j} w_{ij}(1-\mathbf{v_i}\cdot\mathbf{v_j})\right)\frac{2}{\pi}\min_{0<\theta\leq\pi}\frac{\theta}{1-\cos(\theta)} \tag{2.8}$$

$$= OPT' \cdot \frac{2}{\pi}\min_{0<\theta\leq\pi}\frac{\theta}{1-\cos(\theta)} \tag{2.9}$$

$$\geq OPT' \cdot .879 \tag{2.10}$$

□

The minimization at the end follows from a straightforward application of calculus. Thus we have demonstrated Goemans and Williamson's .879-approximation for MaxCut.

2.3 Inapproximability Results

A natural question to ask in the area of approximation algorithms is: what are lower and upper bounds on the approximation ratios we can achieve? For example, it is easy to show the following:

- MaxCut is NP-complete, so efficiently finding the optimal solution is intractable unless P = NP
- There is a simple approximation algorithm with approximation ratio 2 for MaxCut

This is by no means the end of the story. With a 2-approximation for MaxCut but no exact polynomial time algorithm, the natural question to ask is: can we do better? We would like an algorithmic result, that is, an approximation algorithm with a ratio smaller than 2; we might even hope for a polynomial time approximation scheme which would give a $(1 + \epsilon)$-approximation algorithm for any $\epsilon > 0$.

Inapproximability results tell us how much better we could possibly hope to do. They are hardness results because they take the following form: it is NP-hard to approximate a problem to within a factor α. Though a small number of inapproximability results can be proved with elementary techniques, it was not until the advent of the PCP theory in the 1990's that many more inapproximability results could be proved.

2.4 Probabilistically Checkable Proofs (PCP)

Probabilistically checkable proofs are at the heart of the PCP theory, which implies a number of important inapproximability results. Ryan O'Donnell gives a nice tour of the history of the PCP theory in [O'D05] and Sanjeev Arora gives a survey in [Aro02]. Though we will not delve very deeply into the PCP theory in this work, we make the following definition and state the central theorem of the PCP theory so as to motivate our description of the Unique Games Conjecture in the next section [KR03]:

Definition 2.4.1. *An instance $\Phi = (X, Y, R, \Psi, W)$ of* Label Cover *is a set of left vertices X and a set of right vertices Y, together with a set of possible labels R. For each $x \in X$, $y \in Y$, Ψ contains a constraint $\psi_{xy} \subseteq R \times R$, with weight $w_{xy} \in W \geq 0$. A labeling is a function L that maps $X \cup Y$ to R. If $(L(x), L(y)) \in \psi_{xy}$ we say that L satisfies the constraint ψ_{xy}. For short-hand, we will refer to the sum of the weights of the constraints satisfied by a certain labeling as the weight of a labeling. The sum of the weights of all of the constraints is always* 1*.*

We show an example of an instance of Label Cover in Figure 2.2.

A central theorem for hardness results which follows from the PCP theorems [AS98, ALM$^+$98], and the parallel repetition theorem of [Raz98] shows that Label Cover is hard

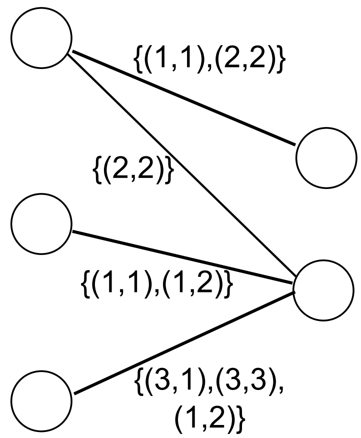

Figure 2.2: This figure shows an instance of Label Cover. The objective is to assign a label to each vertex so that all of the constraints are satisfied. Each edge is shown with a constraint ψ, *i.e.,* the vertices in the edge with constraint $\{(1,1),(2,2)\}$ must be labeled 1 and 1 or 2 and 2 in order to satisfy this constraint. It is easy to check that there is a labeling which satisfies all of the constraints, so the weight of this labeling is 1.

(see also [KR03]):

Theorem 2.4.2. *For any $\gamma > 0$ there exists a suitably large set R such that it is NP-hard to distinguish between an instance of* Label Cover *for which there exists a labeling with weight* 1 *and an instance of* Label Cover *for which every labeling has weight $\leq \gamma$.*

2.5 The Unique Games Conjecture (UGC)

In place of the Label Cover problem described in the previous section, Khot conjectured in [Kho02] that Unique Label Cover is NP-hard. (Even in Khot's original paper, he found it more convenient to talk about Unique Label Cover instead of Unique Games, to which it is equivalent and the source of the name of his conjecture.)

Definition 2.5.1. *An instance $\Phi = (X, Y, R, \Psi, W)$ of* Unique Label Cover *is an instance of* Label Cover *in which for each constraint $\psi_{xy} \in \Psi$ and each label $r_1 \in R$ there exists exactly one label $r_2 \in R$ such that $(r_1, r_2) \in \psi_{xy}$ and for each label $r_2 \in R$ there exists exactly one label $r_1 \in R$ such that $(r_1, r_2) \in \psi_{xy}$.*

We show an example of an instance of Unique Label Cover in Figure 2.3.

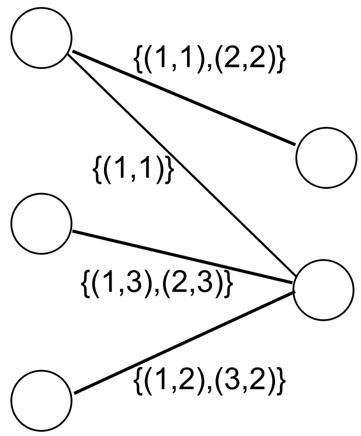

Figure 2.3: This figure shows an instance of UNIQUE LABEL COVER. The objective is to assign a label to each vertex, so as to maximize the weight of the satisfied constraints. Each edge is shown with a constraint ψ. Unlike in the instance of LABEL COVER in Figure 2.2, each ψ is a bijection. It is easy to check that no labeling can satisfy more than half of the constraints.

Given an instance of UNIQUE LABEL COVER a trivial algorithm can determine if all of constraints are satisfiable (unlike in the case of LABEL COVER), so Khot makes the following conjecture (*cf.,* Theorem 2.4.2):

Conjecture 2.5.2 (first conjectured in [Kho02])**.** *It is NP-hard to distinguish between an instance of* UNIQUE LABEL COVER *in which there exists a labeling* L *with weight at least* $1-\zeta$ *and an instance of* UNIQUE LABEL COVER *in which every labeling has weight at most* γ.

2.6 SDP-based Integrality Gaps

Despite the great successes of the PCP theory, there are still a fair number of problems for which no inapproximability results are known. Short of an unconditional inapproximability result, one way to make modest progress is to rule out the possibility of achieving better approximations using a certain approximation technique. Since semidefinite programming is one of the most successful approximation techniques, proving that a better approximation algorithm must use techniques other than semidefinite programming is a useful result, even if it does not imply that progress is impossible (assuming $P \neq NP$). We formalize this idea through the concept of integrality gaps.

Definition 2.6.1. *Given a combinatorial maximization problem, with an integer program with optimum value OPT and a relaxed SDP with optimum value OPT', the integrality gap α is the worst-case ratio $\frac{OPT}{OPT'}$, i.e.:*

$$\alpha = \min_{\textit{instances of the problem}} \frac{OPT}{OPT'}$$

Intuitively, the integrality gap is the gap between the integral solution to the original problem and the solution to the SDP. An integrality gap α implies that if we were to create an approximation algorithm based on this SDP, it would not be able to achieve an approximation ratio better than α. We can demonstrate a bound on the integrality gap for a certain problem by carefully constructing an instance and showing the value of its optimal and relaxed solutions.

2.7 UGC-based Inapproximability Results

Khot's Unique Games Conjecture is useful because we can use UNIQUE LABEL COVER in the place of LABEL COVER in the construction of PCP-based inapproximability proofs. This is especially useful in cases of satisfiability on two variables. As mentioned in the introduction, Khot proved UGC-hardness results for 2-LINEAR-EQUATIONS and NOT-ALL-EQUAL 3-SAT in [Kho02]. Khot and Regev proved results for VERTEXCOVER in [KR03]. A series of papers [KO06, KKMO04, OW07] all contain results for MAXCUT based on the UGC. Chawla *et al.* proved results for Multicut and Sparsest-Cut in [CKK+05]. And finally, Austrin proved results we will consider closely for MAX 2-SAT in [Aus07a] and any 2-CSP in [Aus07b].

2.8 Relating UGC-based Inapproximability and Semidefinite Programming

As we described in the previous section, the Unique Games Conjecture has been used in a number of interesting papers. In many of these, the authors use SDP-based approximation algorithms and integrality gaps in various ways. In a few of these papers, the authors use inapproximability results to produce SDP-based integrality gaps. In [KV05], Khot and Vishnoi construct a semidefinite program for UNIQUE GAMES and show that it has an integrality gap such that for parameters N and η, the SDP solution has weight at least $1-\eta$ while the actual weight of the best labeling is $\leq \frac{1}{N^\eta}$. Then, by applying known PCP-based reductions for various cut problems (including SPARSEST CUT, MINIMUM UNCUT, and MAXCUT) to this program, they create semidefinite programs for these problems. They can use this reduction to prove integrality gaps for these semidefinite programs.

Conceptually, for these cut problems the UGC implies that an SDP-based approximation algorithm cannot achieve an approximation ratio better than the one given by the UG-hardness result, even if the SDP is strengthened in various ways, including adding triangle inequalities or using the Lovász-Schrijver method [LS91]. Furthermore, because the SDPs that Khot and Vishnoi use include the triangle inequalities, these inequalities do not add any power to the SDP for MAXCUT.

Working in the other direction, Khot and O'Donnell give an SDP-based integrality gap for MAXCUT and translate it directly into a Long Code test[2] [KO06]. This Long Code test is then used in a UNIQUE LABEL COVER reduction following [KKMO04], thus proving a UG-hardness result for MAXCUT. This shows that an integrality gap is sometimes as good as a hardness result, at least assuming the UGC.

Some of these papers directly connect semidefinite programming and the UGC. Austrin shows that Lewin, Livnat and Zwick's SDP-based approximation algorithm for MAX 2-SAT [LLZ02] is optimal under the Unique Games Conjecture [Aus07a]. We will build on this work, which is discussed at length in Chapter 4. In [Aus07b] Austrin proves sharp

[2]A discussion of Long Codes, a key component of the PCP theory, are beyond the scope of this work. See Arora's survey [Aro02].

approximability results under the Unique Games Conjecture for any boolean 2-CSP. In [Rag08], Raghavendra makes precise a version of the conjecture we state in the next chapter (independently of our work), showing that under the UGC, for every constraint satisfaction problem an SDP-based approximation algorithm is optimal. See Section 3.2 for a full discussion. In the next chapter, we place these results in the framework of a conjecture we state relating semidefinite programming to the Unique Games Conjecture.

Chapter 3

Our Conjectures

In this chapter, we state formal conjectures connecting semidefinite programming and the Unique Games Conjecture, show that one of our conjectures is trivially true, and categorize the research described in the previous chapter in the framework of our conjectures.

As the results at the end of the previous chapter suggest, there is already a connection between semidefinite programming and the Unique Games Conjecture: for the problems we cited, the Unique Games Conjecture implies that semidefinite programming-based approximation algorithms are optimal. The project we propose in this thesis (but do not come close to solving) is to discover a class of problems, $\mathcal{P}$, for which this relationship between SDP-based approximation algorithms and the UGC holds.

To be precise, we state the following conjecture:

Conjecture 3.0.1. *For a problem* Π *in* $\mathcal{P}$*, there exists an SDP for* Π *and a rounding algorithm for this SDP which gives an optimal approximation ratio, assuming the Unique Games Conjecture.*

A natural candidate for $\mathcal{P}$, inspired by previous work, is constraint satisfaction problems with constraints limited to acting on 2 variables (2-CSPs). As we will see, this is not ambitious enough.

Based again on previous work, we make two related conjectures which specify a possible connection between semidefinite programming and the Unique Games Conjecture. Where our first conjecture only posits a tight algorithmic result based on rounding a semidefinite program, much of the previous work in this area used or created integrality gaps for semidefinite programs. Thus we conjecture that the existence of SDP-based integrality gaps for these problems is equivalent to the existence of inapproximability results under the Unique Games Conjecture. There is a direct connection between SDP-based integrality gaps and approximation algorithms: integrality gaps serve as upper bounds for approximation ratios, and the existence of an approximation ratio α implies the existence of an integrality gap which is at least as bad as α.

Before we make these conjectures, we note that they will only make sense for a class $\mathcal{P}$ of optimization problems for which there is a "natural" semidefinite programming relaxation. As an example, we propose a natural semidefinite program for MAX CSP based on Karloff and Zwick's program in [KZ97]:

Definition 3.0.2. *A standard semidefinite relaxation of an instance I of* MAX CSP *is a semidefinite program with the following properties:*

1. *For each variable x_i of I there is a unit vector $\mathbf{v_i}$. There is also a unit vector $\mathbf{v_0}$ for* FALSE.
2. *For each constraint of I there is a scalar z_a which represents whether or not it is satisfied.*
3. *The objective is $\max_{z_1,\ldots z_m} \sum_{a=1}^{m} w_a z_a$ where the w_a are the weights of the m constraints.*
4. *Every constraint of I corresponds to a set of linear (in)equalities with inner products of the form $\mathbf{v_i} \cdot \mathbf{v_j}$, designed so as to capture the constraint. For example, a constraint in* 2-SAT *is a clause $(x_i \vee x_j)$ which corresponds to $z_j = 1 - \mathbf{v_i} \cdot \mathbf{v_j}$.*

We now formally state these connections:

Conjecture 3.0.3. *For any optimization problem $\Pi \in \mathcal{P}$ which is UG-hard to approximate to within some factor α, there is a natural SDP relaxation for Π which achieves an integrality gap of α.*

Conjecture 3.0.4. *For every problem* $\Pi \in \mathcal{P}$ *with a natural SDP relaxation achieving an integrality gap of* α*,* Π *is UG-hard to approximate to within a factor of* α*.*

These two conjectures imply a deep connection between the UGC and SDP for the problems in $\mathcal{P}$. Conjecture 3.0.3 implies that a problem in $\mathcal{P}$ that is UG-hard to approximate within α cannot have an SDP-based rounding algorithm which does better than α because there is an SDP-based integrality gap of α for the problem. (This statement is trivially true, as we will show.) Conversely, Conjecture 3.0.4 implies that under the UGC, if we demonstrate an SDP-based integrality gap of α for a problem, it is equivalent to proving that the problem is NP-hard to approximate within α.

3.1 Preliminary Results

We can prove a non-constructive version of Conjecture 3.0.3 by contradiction under the Unique Games Conjecture. Given a problem which is UG-hard to approximate to within a factor α, we assume for the sake of contradiction that an SDP-based integrality gap of α does not exist. This means that there is no SDP for which the gap is ever worse than α (*i.e.,* $\geq \alpha$). This is equivalent to saying that in *every* SDP we always achieve a gap better than α. Thus we can choose to round an SDP of our choice in an arbitrary way, because we are assured that the integrality gap is always $\leq \alpha$. This means we have an SDP-based rounding algorithm for this problem which does better than α, contradicting the existence of the UGC-based α-inapproximability result. This proves the theorem.

Despite having this proof, Conjecture 3.0.3 is still interesting if we ask for a constructive proof, in the style of [KV05] which uses the Long Code test for the UG-hardness proof in the construction of an SDP for Π with an integrality gap equal to α.

3.2 Previous Work in the Framework of Our Conjectures

Table 3.1 summarizes the results of the papers described in Section 2.8 in the framework of our conjectures by listing the conjecture which the paper supports. To be precise, if we restrict $\mathcal{P}$ to the problem or problems considered in the paper, than the conjecture is true.

Paper	Problems considered	Conjecture
[Aus07a]	Max 2-Sat and imbalanced Max 2-Sat	3.0.1
[Aus07b]	Max 2-CSP	3.0.1
[KV05]	Sparsest Cut, MinUncut, MaxCut	3.0.3
[KO06]	MaxCut, computing $\|\| \cdot \|\|_{\infty \to 1}$ norm of a matrix	3.0.4
[OW07]	MaxCut	3.0.4

Table 3.1: Previous work in the framework of our conjectures

The most conclusive work on this conjecture is in very recent independent work by Raghavendra [Rag08]. Raghavendra proves Conjecture 3.0.1 for all constraint satisfaction problems. In proving this theorem, he proves Conjecture 3.0.4 for CSPs, demonstrating a conversion from SDP-based integrality gaps to UG-hardness results. Interestingly, he relies on this connection to obtain his approximation algorithm for Conjecture 3.0.1.

More work is needed to see if Raghavendra's techniques can be extended. Perhaps the Unique Games Conjecture implies tight algorithmic results for problems that are not CSPs, and thus Conjecture 3.0.1 holds for a larger class of problems than just CSPs. Raghavendra's work may also have implications for proving a constructive version of 3.0.3.

As a first step in an investigation of this conjecture, we will attempt to extend Austrin's work to a class of imbalanced Max 2-Sat problems parameterized by a measure of imbalance Δ, which we will define in the next chapter.

Chapter 4

Results for Max 2-Sat and Δ-Imbalanced Max 2-Sat

In this chapter, we motivate our work on Max 2-Sat, define the 2-Sat, Max 2-Sat, Balanced Max 2-Sat, and Δ-Imbalanced Max 2-Sat problems, restate Austrin's proof of an approximability result for Max 2-Sat which is based on the algorithm of Lewin, Livnat and Zwick [LLZ02], state our proof of an approximability result for Δ-Imbalanced Max 2-Sat, and present our calculations of inapproximability results for Δ-Imbalanced Max 2-Sat based on Austrin's results [Aus07a]. Together these results show that the approximation algorithm of Lewin, Livnat and Zwick [LLZ02] is optimal for Δ-Imbalanced Max 2-Sat, for all values of Δ, under a certain assumption Austrin makes which we will establish numerically for various values of Δ.

Khot *et al.* sought tight inapproximability results for MaxCut and other 2-CSPs including Max 2-Sat [KKMO04]. They succeeded under the UGC by showing that MaxCut is UG-hard to approximate to within $\alpha \approx .879$, the same approximation ratio which Goemans and Williamson's algorithm achieved [GW95]. Applying the same techniques to Max 2-Sat, they proved that it is UG-hard to approximate Max 2-Sat up to a factor of .9439. This left a small gap in the approximability of Max 2-Sat because Lewin, Livnat and Zwick's algorithm achieves an approximation ratio of .9401. Khot *et al.* observed that their hardness result only relies on instances of MAX 2-SAT which are strictly "balanced." We formalize these definitions before proceeding. We define the 2-SAT decision problem as follows:

Definition 4.0.1. *An instance I of* 2-SAT *on a set of n variables consists of a set of clauses, with each clause $\mathcal{E} \in I$ a disjunction of the form $\ell_1 \vee \ell_2$ where each ℓ_i is a literal, i.e., either a variable or its negation. The decision problem is to decide whether or not there is an assignment of* TRUE *and* FALSE *to the n variables that satisfies all of the clauses.*

We define the MAX 2-SAT combinatorial optimization similarly:

Definition 4.0.2. *An instance I of (weighted)* MAX 2-SAT *is an instance of* 2-SAT *in which each clause $\mathcal{E}$ has a nonnegative weight $w_{\mathcal{E}}$. The objective is to find an assignment of* TRUE *and* FALSE *to the n variables that maximizes the sum of the weights of the satisfied clauses.*

We note that by [CST01], weighted and unweighted instances of MAX 2-SAT are equally hard to approximate. This means that there is an approximation preserving reduction from weighted instances of MAX 2-SAT to unweighted instances, so any hardness result which holds for weighted instances must also hold for unweighted instances.

Khot *et al.* defined BALANCED MAX 2-SAT as follows:

Definition 4.0.3. *An instance of* BALANCED MAX 2-SAT *is an instance of* MAX 2-SAT *for which each variable appears positively and negatively with equal total weight.*

For these balanced instances, Khot *et al.* demonstrated a simple approximation algorithm based on Goemans and Williamson's original algorithm which achieves an approximation ratio of .9439. Reasoning about these balanced cases, Khot *et al.* write [KKMO04, p. 21]:

> We contend that the balanced versions of 2-bit CSPs ought to be equally hard as their general versions; the intuition is that if more constraints are expected to be satisfied if x_i is set to, say, 1 rather than -1, it is a "free hint" that the x_i should be set to TRUE.

Khot *et al.* actually define balanced 2-CSPs more generally, and show that an instance is balanced if the expected number of satisfied constraints when $x_i = $ TRUE and the other variables are assigned uniformly at random is equal to the expected number of satisfied constraints when $x_i = $ FALSE and the other variables are assigned uniformly at random. Thus, their conjecture is equivalent to the claim that instances of BALANCED MAX 2-SAT, in which no variable is more likely to be set to TRUE than any other, are at least as hard to approximate as instances in which the variables are not so strictly balanced.

Surprisingly, Austrin disproves this conjecture, assuming the Unique Games Conjecture [Aus07a]. Austrin proves that there are "imbalanced" instances of MAX 2-SAT, even for a very restricted notion of imbalance, which are harder to approximate than instances of BALANCED MAX 2-SAT.

The terminology Δ-IMBALANCED MAX 2-SAT is our own. Austrin refers to instances of MAX 2-SAT with Δ-mixed clauses [Aus07a]:

Definition 4.0.4. *Given variables x_i, x_j, weight $w_{ij} \geq 0$, and imbalance parameter $-1 \leq \Delta \leq 1$ define a Δ-mixed clause as a pair of clauses:*

1. *$(x_i \vee x_j)$ with weight $w_{ij} \cdot \frac{1+\Delta}{2}$*
2. *$(\neg x_i \vee \neg x_j)$ with weight $w_{ij} \cdot \frac{1-\Delta}{2}$*

Definition 4.0.5. *An instance of* Δ-IMBALANCED MAX 2-SAT *is an instance of* MAX 2-SAT *with variables $x_1, \ldots, x_n$ consisting only of Δ-mixed clauses for all pairs of variables x_i, x_j. Note that $w_{ij} = 0$ is allowed. Also note that clauses of the form $(x_i \vee \neg x_j)$ are not allowed.*

The cases of Δ-IMBALANCED MAX 2-SAT in which $\Delta = 1$ and $\Delta = -1$ can be solved optimally in polynomial time because variables only occur positively or negatively so there is a trivial solution that works. We observe that although the case of Δ-IMBALANCED MAX 2-SAT with $\Delta = 0$ satisfies the requirements of BALANCED MAX 2-SAT, instances of BALANCED MAX 2-SAT are not necessarily instances of Δ-IMBALANCED MAX 2-SAT because they may not be restricted to Δ-mixed clauses.

Using this notation, Austrin proves the following:

Theorem 4.0.6 (Equation (43) in [Aus07a]). *Assuming the UGC it is NP-hard to approximate* Δ-Imbalanced Max 2-Sat *to within a factor:*

$$\min_{\xi\in[-1,1]} \max_{\mu\in[-1,1]} \frac{2-(1+\Delta)\mu-2\Gamma_{\tilde{\rho}}(\mu)}{2-\Delta\xi-|\xi|} + O(\epsilon) \tag{4.1}$$

where $\tilde{\rho} = \frac{|\xi|-1}{|\xi|+1}$

Austrin minimizes this formula to find the hardest instances of Δ-Imbalanced Max 2-Sat. For $\Delta = .367$, which corresponds to instances of Max 2-Sat with an imbalance of 68%, this formula says that it is UG-hard to approximate these instances to within a factor of .9401, matching the approximation ratio of Lewin, Livnat and Zwick for general instances of Max 2-Sat.

We graph known results for the approximability and inapproximability of Δ-Imbalanced Max 2-Sat in Figure 4.1. As the graph shows, there are gaps in our knowledge. Austrin's formula for inapproximability holds for all values of Δ, but Austrin did not calculate hardness results for values of Δ other than .367 and −.367. We will calculate these values and fill in this part of the graph in Figure 4.7. Furthermore, Austrin did not investigate the tightness of his inapproximability results for other values of Δ. We will fill in the approximability curve in the graph based on our proof of the approximability of Δ-Imbalanced Max 2-Sat for all values of Δ. A priori, it is conceivable that our algorithmic results will not match the hardness results which Austrin already found and we will be left with two curves and a gap.

In Section 4.2.1 we will prove that a modification of Lewin, Livnat, and Zwick's algorithm for Max 2-Sat works for Δ-Imbalanced Max 2-Sat. Analyzing this algorithm under certain plausible assumptions, we obtain a formula for the approximation ratio of Δ-Imbalanced Max 2-Sat which exactly matches the formula given by Austrin for the inapproximability of Δ-Imbalanced Max 2-Sat under the Unique Games Conjecture:

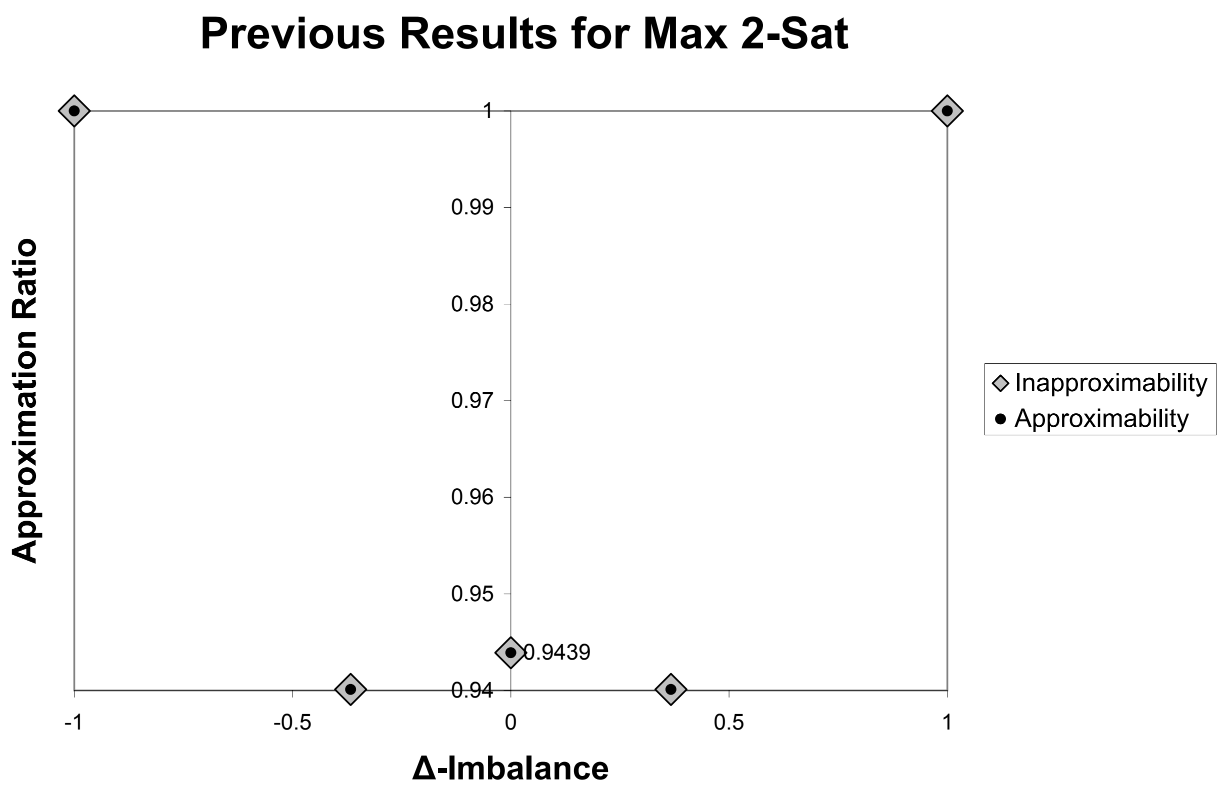

Figure 4.1: This graph shows known approximability (algorithmic) and inapproximability (hardness) results for Δ-Imbalanced Max 2-Sat. See Table 4.1 for the sources of these results. It should also be noted that the result in Lewin, Livnat and Zwick [LLZ02] gives a .9401-approximation for Max 2-Sat which means it is a lower-bound for all values of Δ, so this graph should contain a line for approximability along the bottom. This graph shows Austrin's interesting and surprising result: Balanced Max 2-Sat is easier to approximate than Max 2-Sat. Our project is to fill in the rest of the graph for other values of Δ.

Theorem 4.0.7. *There is an SDP-based rounding algorithm for* Δ-Imbalanced Max 2-Sat *with approximation ratio given by:*

$$\max_{\beta \in [0,1]} \min_{\xi \in [-1,1]} \frac{2 - (1+\Delta)\beta \cdot \xi - 2\Gamma_{\tilde{\rho}}(\beta\xi)}{2 - \Delta\xi - |\xi|}$$

where $\tilde{\rho} = \frac{|\xi|-1}{|\xi|+1}$

Because the formulas in Theorems 4.0.6 and 4.0.7 are equivalent, we will have successfully filled in the graph in Figure 4.7 with one tight curve. Note that if the UGC is false than these inapproximability results do not necessarily hold and a new approximation algorithm could be devised which achieves a better approximation ratio for imbalanced instances. So Khot *et al.*'s conjecture might still hold, but not under the UGC. We summarize the previous results for approximability and inapproximability and the results we will prove in Table 4.1.

Paper	Type of result	Approximation ratio
[LLZ02]	Approximability for Max 2-Sat	.9401
[KKMO04]	Inapproximability for Balanced Max 2-Sat	.9439
[KKMO04]	Approximability for Balanced Max 2-Sat	.9439
[Aus07a]	Inapproximability for Δ-Imbalanced Max 2-Sat	Theorem 4.0.6
[Aus07a]	Inapproximability for Max 2-Sat	.9401
This work	Approximability for Δ-Imbalanced Max 2-Sat	Theorem 4.0.7 (matches 4.0.6)

Table 4.1: Previous results for different cases of Max 2-Sat. Hardness results assume the Unique Games Conjecture.

4.1 Approximating Max 2-Sat

In this section we state and prove Austrin's algorithmic results for approximating Max 2-Sat [Aus07a]. We start with some definitions which we will need.

4.1.1 Preliminaries

Definition 4.1.1. *The standard normal density function $\phi(x)$ is given by:*

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \tag{4.2}$$

Definition 4.1.2. *For X a normally distributed random variable, the standard normal distribution function $\Phi(x)$ is given by:*

$$\Phi(x) = \Pr[X < x] = \int_{-\infty}^{x} \phi(t) dt = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2} dt \tag{4.3}$$

For X and Y normally distributed random variables with correlation ρ, the bivariate normal distribution of X and Y at a point (x, y) is $\Pr[X \leq x \text{ and } Y \leq y]$.

Definition 4.1.3. *We define a function* Γ_ρ *as the bivariate normal distribution with the following transformation on its input:*

$$\Gamma_\rho(\mu_1, \mu_2) = \Pr[X \le t_1 \textit{ and } Y \le t_2] \tag{4.4}$$

where $t_1 = \Phi^{-1}(\frac{1-\mu_1}{2})$ *and* $t_2 = \Phi^{-1}(\frac{1-\mu_2}{2})$.

Finally, we state a useful lemma. The proof can be found in Appendix D of [Aus07a].

Lemma 4.1.4. *For all* $\rho, \mu_1, \mu_2 \in [-1, 1]$ *the following holds:*

$$\Gamma_\rho(-\mu_1, -\mu_2) = \Gamma_\rho(\mu_1, \mu_2) + \frac{\mu_1}{2} + \frac{\mu_2}{2} \tag{4.5}$$

4.1.2 The Quadratic Program and Semidefinite Program for Max 2-Sat

Following [Aus07a], we present and analyze Lewin, Livnat, and Zwick's THRESH⁻ family of rounding algorithms for MAX 2-SAT [LLZ02]. In Section 4.2, we modify this algorithm for Δ-IMBALANCED MAX 2-SAT.

Lewin, Livnat and Zwick analyze the performance of a few different families of rounding algorithms applied to the same standard semidefinite program for MAX 2-SAT. They call their best family of rounding algorithms THRESH⁻ because it is a subset of the family of algorithms called THRESH, *i.e.,* threshold. The reason for this name is that the rounding is based on whether a certain value is above or below a certain threshold.

We present the standard quadratic program for MAX 2-SAT in Figure 4.2. We create variables $x_1, \ldots, x_n$ which take the value -1 for TRUE and 1 for FALSE. Following [LLZ02] we set $x_{n+i} = \neg x_i$, the so-called consistency requirement. Thus, the entire set of clauses is $x_i \vee x_j$ for $1 \le i, j \le 2n$. We call the weight of a clause w_{ij}. The arithmetization in 4.2 can be checked easily.

In Figure 4.3 we relax the quadratic program to create a semidefinite program for MAX 2-SAT. The variables x_i are relaxed into unit vectors $\mathbf{v_i}$ for $1 \le i \le 2n$. We create a fixed vector $\mathbf{v_0} = (1, 0, \ldots, 0)$ which corresponds to FALSE. The requirement that $x_{n+i} = \neg x_i$ now becomes $\mathbf{v_i} \cdot \mathbf{v_{n+1}} = -1$. Following [FG95] we have strengthened the program by

$$\max \frac{1}{4}\sum_{i,j} w_{ij}(3 - x_i - x_j - x_i x_j)$$
$$x_{n+i} = \neg x_i \text{ for } 1 \le i \le n$$
$$x_i \in \{-1, 1\}$$

Figure 4.2: The quadratic program for Max 2-Sat

adding the triangle constraints (note that there are only four because although there are eight combinations of signs for $\mathbf{v_0}$, $\mathbf{v_i}$, and $\mathbf{v_j}$, pairing them as in the triangle inequalities cuts the possibilities down to four.)

$$\max \frac{1}{4}\sum_{i,j} w_{ij}(3 - \mathbf{v_0}\cdot\mathbf{v_i} - \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j})$$
$$\mathbf{v_i}\cdot\mathbf{v_{n+i}} = -1 \text{ for } 1 \le i \le n$$
$$\mathbf{v_i} \in \mathcal{R}^{n+1} \text{ for } 1 \le i \le 2n$$
$$\mathbf{v_i}\cdot\mathbf{v_i} = 1 \text{ for } 1 \le i \le 2n$$
$$\mathbf{v_0} = (1, 0, \ldots, 0)$$
$$\text{The triangle inequalities, for } 1 \le i \le 2n:$$
$$\mathbf{v_0}\cdot\mathbf{v_i} + \mathbf{v_0}\cdot\mathbf{v_j} + \mathbf{v_i}\cdot\mathbf{v_j} \ge -1$$
$$-\mathbf{v_0}\cdot\mathbf{v_i} + \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j} \ge -1$$
$$\mathbf{v_0}\cdot\mathbf{v_i} - \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j} \ge -1$$
$$-\mathbf{v_0}\cdot\mathbf{v_i} - \mathbf{v_0}\cdot\mathbf{v_j} + \mathbf{v_i}\cdot\mathbf{v_j} \ge -1$$

Figure 4.3: The semidefinite program for Max 2-Sat

4.1.3 Rounding the Semidefinite Program

Following Austrin (and the prevailing practice) we note that while it is possible to find the optimum of a semidefinite program to within an arbitrarily small additive term, it is not known how to efficiently find the exact optimum. To make the notation simpler, we do not include this term in what follows, speaking instead of finding the optimal solution to the SDP.

Given an optimal solution $(\mathbf{v_0}, \ldots, \mathbf{v_n})$ we want to round these vectors so that we can assign boolean values to the $(x_0, \ldots, x_n)$.

For each vector v_i, we calculate the scalar projection of $\mathbf{v_i}$ onto $\mathbf{v_0}$ for all i:

$$\xi_i = \mathbf{v_0} \cdot \mathbf{v_i} \tag{4.6}$$

Now we calculate the component of $\mathbf{v_i}$ orthogonal to $\mathbf{v_0}$ as $\mathbf{v_i} - \xi_i \mathbf{v_0}$.

Normalizing, we call this vector $\widetilde{\mathbf{v_i}}$, the unit vector in the direction of the component of $\mathbf{v_i}$ orthogonal to $\mathbf{v_0}$:

$$\widetilde{\mathbf{v_i}} = \frac{\mathbf{v_i} - \xi_i \mathbf{v_0}}{\sqrt{1 - \xi_i^2}} \tag{4.7}$$

In Figure 4.4, we show the rounding algorithm THRESH⁻:

Step 1: Choose a standard normal random vector $\mathbf{r}$ in the n-dimensional subspace of $\mathcal{R}^{n+1}$ orthogonal to $\mathbf{v_0}$.

Step 2: For each vector $\mathbf{v_i}$, round it as follows, where $T : [-1, 1] \to \mathcal{R}$:

$$x_i = \begin{cases} \text{TRUE} & \text{if } \widetilde{\mathbf{v_i}} \cdot \mathbf{r} \leq T(\xi_i) \\ \text{FALSE} & \text{if } \widetilde{\mathbf{v_i}} \cdot \mathbf{r} > T(\xi_i) \end{cases}$$

Figure 4.4: Lewin Livnat and Zwick's THRESH⁻ family of rounding algorithms

Though there are certainly more complicated choices, [LLZ02] found a very simple function for T which performs well. Following Austrin's notation, we use:

$$T(x) = \Phi^{-1}\left(\frac{1 - a(x)}{2}\right) \tag{4.8}$$

See the Preliminaries section (4.1.1) for the definition of Φ.

Since $\mathbf{v_i} = -\mathbf{v_{n+i}}$ we have that $\xi_i = -\xi_{n+i}$, so to satisfy the consistency requirement, $T(-x) = -T(x)$ *i.e.,* $a(x)$ must be an odd function.

4.1.4 Analysis of THRESH$^-$

To find the approximation ratio α of this rounding algorithm we follow [GW95] in looking at a single clause $(x_i \vee x_j)$. For a single clause, the expected contribution to the integer linear program objective function (where the expectation is over the randomness in the rounding algorithm) is:

$$\frac{1}{4} w_{ij} \, \mathrm{E}[3 - x_i - x_j - x_i x_j] \tag{4.9}$$

For this same clause, the contribution to the SDP objective function is:

$$\frac{1}{4} w_{ij} (3 - \mathbf{v_0} \cdot \mathbf{v_i} - \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}) \tag{4.10}$$

We are looking for a lower bound on α so we minimize the ratio of Equations 4.9 and 4.10 over all feasible vector solutions to the SDP:

$$\min_{\substack{v \in (S^n)^{n+1} \text{ and} \\ v \text{ is a feasible solution to the SDP}}} \frac{\mathrm{E}[3 - x_i - x_j - x_i x_j]}{3 - \mathbf{v_0} \cdot \mathbf{v_i} - \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}} \tag{4.11}$$

Note that in calculating the approximation ratio we have used the standard trick in analyzing approximation algorithms based on rounding: because the value of the relaxed solution is at least as good as the value of the optimal solution, the real approximation ratio (*i.e.,* the ratio between the value of the approximation algorithm and the value of the optimal solution) is at least as good as the ratio we are calculating, between the value of the approximation algorithm and the value of the relaxed solution. This mode of analysis was described in Section 2.1 and illustrated in Figure 2.1.

To calculate Equation 4.11 we use the linearity of expectations. We will first analyze the linear terms and then the quadratic terms. First, we prove two lemmas which will aid our calculations:

Lemma 4.1.5. $\widetilde{\mathbf{v_i}} \cdot \mathbf{r}$ *is a standard* $N(0,1)$ *random variable*

Proof. $\mathbf{r}$ was chosen as a standard normal random vector in the n-dimensional subspace of $\mathcal{R}^{n+1}$ orthogonal to $\mathbf{v_0}$, the same subspace that contains $\widetilde{\mathbf{v_i}}$. Thus $\widetilde{\mathbf{v_i}} \cdot \mathbf{r}$ is a linear combination of n standard $N(0,1)$ random variables, which is a standard $N(0,1)$ random variable [Fel71, p. 87]. □

Lemma 4.1.6. *x_i is set to* TRUE *with probability* $\frac{1-a(\xi_i)}{2}$

Proof.

$$\Pr[x_i = \text{TRUE}] = \Pr\left[\widetilde{vi} \cdot \mathbf{r} \leq \Phi^{-1}\left(\frac{1-a(\xi_i)}{2}\right)\right] \tag{4.12}$$

$$= \Pr\left[\Phi(\widetilde{vi} \cdot \mathbf{r}) \leq \frac{1-a(\xi_i)}{2}\right] \tag{4.13}$$

$$= \frac{1-a(\xi_i)}{2} \tag{4.14}$$

Equation 4.14 follows from Lemma 4.1.5. □

Applying Lemma 4.1.6, a simple calculation shows that for all i: $\mathrm{E}[x_i] = a(\xi_i)$.

We now turn to the quadratic terms. Let $\rho = \mathbf{v_i} \cdot \mathbf{v_j}$. We calculate the covariance of $\widetilde{\mathbf{v_i}} \cdot \mathbf{r}$ and $\widetilde{\mathbf{v_j}} \cdot \mathbf{r}$:

$$Cov(\widetilde{\mathbf{v_i}} \cdot \mathbf{r}, \widetilde{\mathbf{v_j}} \cdot \mathbf{r}) = \mathrm{E}\left[(\widetilde{\mathbf{v_i}} \cdot \mathbf{r} - \mathrm{E}[\widetilde{\mathbf{v_i}} \cdot \mathbf{r}])(\widetilde{\mathbf{v_j}} \cdot \mathbf{r} - \mathrm{E}[\widetilde{\mathbf{v_j}} \cdot \mathbf{r}])\right] \tag{4.15}$$

$$= \mathrm{E}[(\widetilde{\mathbf{v_i}} \cdot \mathbf{r})(\widetilde{\mathbf{v_j}} \cdot \mathbf{r})] \tag{4.16}$$

$$= \mathrm{E}\left[\sum_p \widetilde{v_{ip}} r_p \sum_q \widetilde{v_{jq}} r_q\right] \tag{4.17}$$

$$= \sum_{p,q} \left(\widetilde{v_{ip}}\widetilde{v_{jq}}\, \mathrm{E}[r_p r_q]\right) \tag{4.18}$$

$$= \sum_p \widetilde{v_{ip}}\widetilde{v_{jp}} \tag{4.19}$$

$$= \widetilde{\mathbf{v_i}} \cdot \widetilde{\mathbf{v_j}} \tag{4.20}$$

Equation 4.19 holds because each component of $\mathbf{r}$ is an independent standard $N(0,1)$ random variable, so $\mathrm{E}[r_p r_q] = 0$ for $p \neq q$ and 1 for $p = q$.

Let $\widetilde{\rho} = \widetilde{\mathbf{v_i}} \cdot \widetilde{\mathbf{v_j}}$. We calculate:

$$\widetilde{\rho} = \widetilde{\mathbf{v_i}} \cdot \widetilde{\mathbf{v_j}} = \frac{\rho - \xi_i \mathbf{v_0}}{\sqrt{1-\xi_i^2}} \frac{\rho - \xi_j \mathbf{v_0}}{\sqrt{1-\xi_j^2}} \tag{4.21}$$

$$= \frac{\rho - \xi_i \xi_j}{\sqrt{(1-\xi_i^2)(1-\xi_j^2)}} \tag{4.22}$$

We want to know the probability that both x_i and x_j are set to true:

$$\Pr[\widetilde{\mathbf{v_i}} \cdot \mathbf{r} \leq T(\xi_i) \text{ and } \widetilde{\mathbf{v_j}} \cdot \mathbf{r} \leq T(\xi_j)] = \Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) \tag{4.23}$$

The Γ_ρ function was defined in the Preliminaries section (4.1.1).

By the consistency requirement, the probability that both x_i and x_j are set to false must be:

$$\Gamma_{\widetilde{\rho}}(a(-\xi_i), a(-\xi_j)) \tag{4.24}$$

Finally, we calculate:

$$\mathrm{E}[x_i x_j] = \Pr[x_i = x_j] - (1 - \Pr[x_i = x_j]) \tag{4.25}$$

$$= 2\Pr[x_i = x_j] - 1 \tag{4.26}$$

$$= 2\left[\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + \Gamma_{\widetilde{\rho}}(a(-\xi_i), a(-\xi_j))\right] - 1 \tag{4.27}$$

$$= 2\left[2\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + \frac{a(\xi_i)}{2} + \frac{a(\xi_j)}{2}\right] - 1 \tag{4.28}$$

$$= 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + a(\xi_i) + a(\xi_j) - 1 \tag{4.29}$$

Equation 4.28 follows from Proposition 4.1.4 in the Preliminaries section (4.1.1).

Putting all of these calculations together we find:

$$\mathrm{E}[3 - x_i - x_j - x_i x_j] \tag{4.30}$$

$$= 3 - a(\xi_i) - a(\xi_j) - (4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + a(\xi_i) + a(\xi_j) - 1) \tag{4.31}$$

$$= 4 - 2a(\xi_i) - 2a(\xi_j) - 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) \tag{4.32}$$

Thus:

$$\min_{\substack{v \in (S^n)^{n+1} \text{ and} \\ v \text{ is a feasible solution to the SDP}}} \frac{\mathrm{E}[3 - x_i - x_j - x_i x_j]}{3 - \mathbf{v_0} \cdot \mathbf{v_i} - \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}} \tag{4.33}$$

$$= \min_{(\xi_i, \xi_j, \rho) \text{ satisfy triangle constraints}} \frac{4 - 2a(\xi_i) - 2a(\xi_j) - 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j))}{3 - \xi_i - \xi_j - \rho} \tag{4.34}$$

4.1.5 Conclusion

Following [Aus07a] we let $a(x) = \beta \cdot x$ so that this becomes:

$$\min_{(\xi_i, \xi_j, \rho) \text{ satisfy triangle constraints}} \frac{4 - 2\beta(\xi_i + \xi_j) - 4\Gamma_{\widetilde{\rho}}(\beta\xi_i, \beta\xi_j)}{3 - \xi_i - \xi_j - \rho} \tag{4.35}$$

According to Lewin, Livnat and Zwick the minima for the above expression, which are the worst-case configurations (ξ_i, ξ_j, ρ), all take the form of the "simple" configuration: $(\xi, \xi, -1 + 2|\xi|)$ [LLZ02]. Lewin, Livnat and Zwick do not give an analytic proof of this fact, instead providing convincing numeric evidence. We will need to make this same assumption, that worst-case configurations have a simple form, when we extend Austrin's work in the next section.

Under this assumption, to find the approximation ratio α we minimize over a single parameter ξ and maximize over the parameter of the threshold function, β. Recalculating we find $\tilde{\rho} = \frac{\rho - \xi_i \xi_j}{\sqrt{(1-\xi_i^2)(1-\xi_j^2)}} = \frac{|\xi| - 1}{|\xi| + 1}$:

$$\alpha = \max_{\beta \in [0,1]} \min_{\xi \in [-1,1]} \frac{4 - 4\beta\xi - 4\Gamma_{\widetilde{\rho}}(\beta\xi, \beta\xi)}{3 - 2\xi - (-1 + 2|\xi|)} = \max_{\beta \in [0,1]} \min_{\xi \in [-1,1]} \frac{2 - 2\beta\xi - 2\Gamma_{\widetilde{\rho}}(\beta\xi, \beta\xi)}{2 - \xi - |\xi|} \tag{4.36}$$

Numerical calculations in [LLZ02] and done more explicitly in [Aus07a] show that $\alpha \approx .9401$. As we shall see, Austrin finds the same expression for the inapproximability of Max 2-Sat, thus proving that this approximation ratio is tight.

4.2 Approximating Δ-Imbalanced Max 2-Sat

In this section, we turn our attention to a problem formulated by Austrin, for which no previous work on approximability exists. We prove a tight result for Δ-Imbalanced Max 2-Sat which relies on the same assumption described at the end of the previous section, that worst-case configurations have a simple form.

4.2.1 The Quadratic Program and Semidefinite Program for Δ-Imbalanced Max 2-Sat

We consider how the quadratic program for Max 2-Sat in Figure 4.2 must be changed. This is the objective function from the program in Figure 4.2:

$$\max \frac{1}{4} \sum_{i,j} (3 - x_i - x_j - x_i \cdot x_j) \tag{4.37}$$

We consider this summation for all pairs $1 \leq i < j \leq n$ consisting of $(x_i \vee x_j)$ and $(\neg x_i \vee \neg x_j)$. Note that we are no longer considering clauses of the form $(x_i \vee \neg x_j)$, so $i, j \leq n$ instead of $2n$ and we will omit the consistency requirement $x_{n+i} = -x_i$. The definition of Δ-Imbalanced Max 2-Sat says these two clauses have weight $w_{ij}\frac{1+\Delta}{2}$ and $w_{ij}\frac{1-\Delta}{2}$ respectively. The sum of this pair in the ILP is thus:

$$w_{ij}\frac{1+\Delta}{2}(3 - x_i - x_j - x_i x_j) + w_{ij}\frac{1-\Delta}{2}(3 - \neg x_i - \neg x_j - \neg x_i \neg x_j) \tag{4.38}$$

$$= w_{ij}\frac{1+\Delta}{2}(3 - x_i - x_j - x_i x_j) + w_{ij}\frac{1-\Delta}{2}(3 + x_i + x_j - x_i x_j) \tag{4.39}$$

$$= w_{ij}(3 - \Delta x_i - \Delta x_j - x_i x_j) \tag{4.40}$$

Thus, we have the quadratic program for Δ-Imbalanced Max 2-Sat in Figure 4.5.

$$\max \frac{1}{4}\sum_{i<j} w_{ij}(3 - \Delta x_i - \Delta x_j - x_i x_j)$$
$$x_i \in \{-1, 1\}$$

Figure 4.5: The quadratic program for Δ-Imbalanced Max 2-Sat

$$\max \ \frac{1}{4}\sum_{i<j} w_{ij}(3 - \Delta \mathbf{v_0}\cdot\mathbf{v_i} - \Delta \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j})$$

$$\mathbf{v_0}\cdot\mathbf{v_i} + \mathbf{v_0}\cdot\mathbf{v_j} + \mathbf{v_i}\cdot\mathbf{v_j} \geq -1 \text{ for } 1 \leq i \leq n$$
$$\mathbf{v_i} \in \mathcal{R}^{n+1} \text{ for } 1 \leq i \leq n$$
$$\mathbf{v_i}\cdot\mathbf{v_i} = 1 \text{ for } 1 \leq i \leq n$$
$$\mathbf{v_0} = (1, 0, \ldots, 0)$$
$$\text{The triangle inequalities, for } 1 \leq i \leq n:$$
$$\mathbf{v_0}\cdot\mathbf{v_i} + \mathbf{v_0}\cdot\mathbf{v_j} + \mathbf{v_i}\cdot\mathbf{v_j} \geq -1$$
$$-\mathbf{v_0}\cdot\mathbf{v_i} + \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j} \geq -1$$
$$\mathbf{v_0}\cdot\mathbf{v_i} - \mathbf{v_0}\cdot\mathbf{v_j} - \mathbf{v_i}\cdot\mathbf{v_j} \geq -1$$
$$-\mathbf{v_0}\cdot\mathbf{v_i} - \mathbf{v_0}\cdot\mathbf{v_j} + \mathbf{v_i}\cdot\mathbf{v_j} \geq -1$$

Figure 4.6: The semidefinite program for Δ-Imbalanced Max 2-Sat

We can relax the ILP as before to get the program in Figure 4.6. Notice that in both the ILP and the SDP the Δ term is merely a dampening factor on the linear terms.

4.2.2 Rounding the Semidefinite Program

We round the SDP in Figure 4.6 as in Section 4.1.3, using the THRESH$^-$ family of rounding algorithms. However, unlike in the Max 2-Sat case, different values of Δ will almost certainly have different threshold functions. We continue working under the assumption that the simple function $a(x) = \beta \cdot x$ still achieves the optimal approximation ratio, though there is no a priori reason to believe this, until we show the tightness of the approximability results for this function $a(x)$ in Section 4.3. Thus, for different values of Δ we will be changing the value of β.

4.2.3 Analysis of THRESH$^-$

To find the approximation ratio α we must now consider a pair of clauses $(x_i \vee x_j)$ and $(\neg x_i \vee \neg x_j)$. For this pair, the expected contribution to the ILP objective function (where the expectation is over the randomness in the rounding algorithm) is:

$$\frac{1}{4} w_{ij} \, \mathrm{E}[3 - \Delta x_i - \Delta x_j - x_i x_j] \tag{4.41}$$

For this same clause, the contribution to the SDP objective function is:

$$\frac{1}{4} w_{ij} (3 - \Delta \mathbf{v_0} \cdot \mathbf{v_i} - \Delta \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}) \tag{4.42}$$

We are looking for a lower bound on α so we minimize over all feasible vector solutions to the SDP:

$$\min_{\substack{v \in (S^n)^{n+1} \text{ and} \\ v \text{ is a feasible solution to the SDP}}} \frac{\mathrm{E}[3 - \Delta x_i - \Delta x_j - x_i x_j]}{3 - \Delta \mathbf{v_0} \cdot \mathbf{v_i} - \Delta \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}} \tag{4.43}$$

We note that, as before, $\widetilde{\mathbf{v_i}} \cdot \mathbf{r}$ is still a standard $N(0,1)$ random variable and x_i is still set to TRUE with probability $\frac{1-a(\xi_i)}{2}$. The Δ passes outside the expectation by the linearity of expectations, so we don't have to worry about it and thus: $\mathrm{E}[x_i] = a(\xi_i)$. For the quadratic terms, nothing changes, so:

$$\mathrm{E}[x_i x_j] = 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + a(\xi_i) + a(\xi_j) - 1$$

Putting all of these calculations together we find:

$$\mathrm{E}[3 - \Delta x_i - \Delta x_j - x_i x_j] \tag{4.44}$$

$$= 3 - \Delta a(\xi_i) - \Delta a(\xi_j) - (4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) + a(\xi_i) + a(\xi_j) - 1) \tag{4.45}$$

$$= 4 - (1+\Delta) a(\xi_i) - (1+\Delta) a(\xi_j) - 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j)) \tag{4.46}$$

Thus:

$$\min_{\substack{v \in (S^n)^{n+1} \text{ and} \\ v \text{ is a feasible solution to the SDP}}} \frac{\mathrm{E}[3 - \Delta x_i - \Delta x_j - x_i x_j]}{3 - \Delta \mathbf{v_0} \cdot \mathbf{v_i} - \Delta \mathbf{v_0} \cdot \mathbf{v_j} - \mathbf{v_i} \cdot \mathbf{v_j}} \tag{4.47}$$

$$= \min_{(\xi_i, \xi_j, \rho) \text{ satisfy triangle constraints}} \frac{4 - (1+\Delta) a(\xi_i) - (1+\Delta) a(\xi_j) - 4\Gamma_{\widetilde{\rho}}(a(\xi_i), a(\xi_j))}{3 - \Delta\xi_i - \Delta\xi_j - \rho} \tag{4.48}$$

Now following [LLZ02] we let $a(x) = \beta \cdot x$ so that this becomes:

$$\min_{(\xi_i, \xi_j, \rho) \text{ satisfy triangle constraints}} \frac{4 - (1+\Delta)\beta(\xi_i + \xi_j) - 4\Gamma_{\widetilde{\rho}}(\beta\xi_i, \beta\xi_j)}{3 - \Delta\xi_i - \Delta\xi_j - \rho} \tag{4.49}$$

4.2.4 Conclusion

Austrin bases his approximability results on the assumption that the simple configurations are the worst ones, a plausible assumption that Lewin, Livnat, and Zwick arrived at through numerical evidence from MATLAB. We would like to make this same assumption. We state it formally as a conjecture:

Conjecture 4.2.1. *The minima of the following expression all have the form* $(\xi_i, \xi_j, \rho) = (\xi, \xi, -1 + 2|\xi|)$*:*

$$\min_{(\xi_i, \xi_j, \rho) \textit{ satisfy triangle constraints}} \frac{4 - (1+\Delta)\beta(\xi_i + \xi_j) - 4\Gamma_{\widetilde{\rho}}(\beta\xi_i, \beta\xi_j)}{3 - \Delta\xi_i - \Delta\xi_j - \rho} \tag{4.50}$$

The intuition for setting $\xi_i = \xi_j$, as explained by Austrin, is as follows: since the function we are minimizing is symmetric in the ξ_i and ξ_j terms, there is no advantage to $\mathbf{v_0}$ having different angles to $\mathbf{v_i}$ and $\mathbf{v_j}$, so we set $\xi_i = \xi_j$. Secondly, the simple configuration says that $\rho = -1 + 2|\xi|$ which is equivalent to $-2|\xi| + \rho = -1$. This is the same as saying that at least one of the triangle inequalities shown in Figure 4.3 is tight. The intuition is that making a triangle inequality tight means being on the edge of the feasible configuration space, as close as possible to the infeasible part of the configuration space which should contain very bad configurations.

For our case, we justify making the same assumption (while still noting that it is unproven) because the basic form of the expression we are minimizing is very similar to the expression minimized in [Aus07a], with the only change being the dampening effects of the Δ terms. The intuition still holds as well. But since our assumption is, at the end of the day, only based on numerical evidence from [LLZ02], we will provide some numerical evidence of our own that it is correct. We check Conjecture 4.2.1 for a few values of Δ, with results shown in Table 4.2.4. To find these configurations we have also minimized over β, though our investigations suggest that fixing β would have worked equally well, because the basic form of equation 4.50 does not change for different values of β.

Δ	Configuration
0	$(0.1846, 0.1846 - 0.6309)$
.25	$(0.1946, 0.1946, -0.6109)$
.367	$(0.2056, 0.2056, -0.5889)$
.5	$(0.2150, 0.2150, -0.5701)$
.75	$(0.2373, 0.2373, -0.5255)$

Table 4.2: Worst-case configurations for various values of Δ. In each case, they take a "simple" form: $(\xi, \xi, -1 + 2|\xi|)$.

The numerical evidence obtained with MATLAB for a few values of Δ supports our conjecture. If we assume Conjecture 4.2.1 for all values of Δ we arrive at the following expression for the approximation algorithm, where the configuration is $(\xi_i, \xi_j, \rho) = (\xi, \xi, -1+2|\xi|)$ and we want to pick the best threshold function parameterized by β:

$$\max_{\beta \in [0,1]} \min_{\xi \in [-1,1]} \frac{2 - (1+\Delta)\beta \cdot \xi - 2\Gamma_{\tilde{\rho}}(\beta\xi)}{2 - \Delta\xi - |\xi|} \tag{4.51}$$

As we will see in the next section, this expression exactly matches Austrin's inapproximability result for all values of Δ, so this result is tight, assuming our conjecture.

4.3 Inapproximability Results for Max 2-Sat and Δ-Imbalanced Max 2-Sat

In [Aus07a], Austrin proves the following general result:

Theorem 4.3.1 (Equation (43) in [Aus07a]). *Assuming the UGC it is NP-hard to approximate* Δ-Imbalanced Max 2-Sat *to within a factor:*

$$\min_{\xi \in [-1,1]} \max_{\mu \in [-1,1]} \frac{2 - (1+\Delta)\mu - 2\Gamma_{\tilde{\rho}}(\mu)}{2 - \Delta\xi - |\xi|} + O(\epsilon) \tag{4.52}$$

where $\tilde{\rho} = \frac{|\xi|-1}{|\xi|+1}$

Although Austrin proved that this formula holds for all values of Δ, he did not report calculations for values of Δ other than $\Delta = 0$ (the balanced case) and $\Delta \approx .3673$, the hardest case, which gives an approximation ratio $\alpha = .9401$.

We use MATLAB together with a package for calculating Bivariate Normal Distributions[1] (the function $\Gamma_{\tilde{\rho}}(\mu)$ in the formula) to graph this formula. Our graph is shown in Figure 4.7.

This expression matches the one given for approximability in Equation 4.51, so we conclude that under both the assumptions that led to Equation 4.51 and under the UGC, the algorithm of Lewin, Livnat and Zwick in [LLZ02] is the best possible, *i.e.,* the graph in Figure 4.7 is tight because it holds for both approximation and inapproximability.

[1]See BVNL, "A Matlab function for the computation of bivariate normal cdf probabilities" created by Alan Genz. Online at http://www.math.wsu.edu/faculty/genz/software/software.html

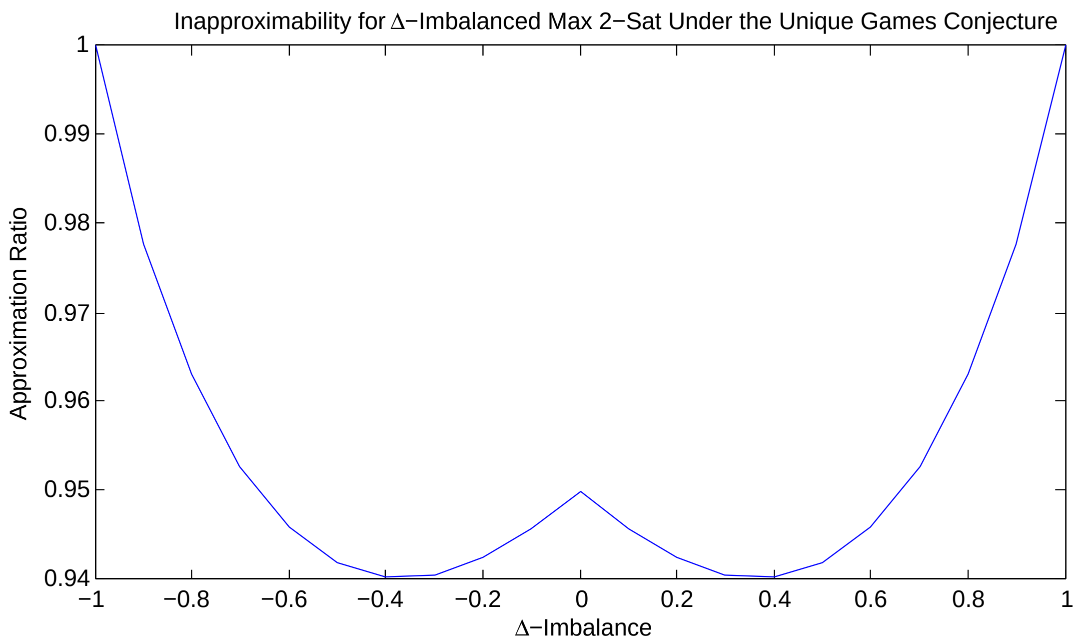

Figure 4.7: This graph was plotted based on Austrin's formula for the inapproximability of Δ-Imbalanced Max 2-Sat. By Equation 4.51 it is a tight curve for both approximability and inapproximability, filling out the curve in Figure 4.1

Chapter 5

Conclusion

Our work does not immediately get us closer to proving our conjectures for a general class of problems because the techniques we applied only served to demonstrate the utility of SDP-based approximation algorithms. However, as demonstrated by the graph in Figure 4.7, our work shows in even more stark terms the surprising conclusion of Austrin's work [Aus07a]: the hardest instances of Δ-Imbalanced Max 2-Sat are those in which it would seem we have a "free hint" (as [KKMO04] term it) about the solution. This conclusion is by no means startling enough to make us question the Unique Games Conjecture, but it does inspire us to push further.

Though we remarked earlier that the Unique Games Conjecture is stronger than $P \neq NP$, there is another likely possibility: proving that the Unique Games Conjecture is equivalent to $P \neq NP$, *i.e.,* showing that Unique Label Cover is NP-hard. Alternatively, a new approximation technique might be devised which disproves the UGC, perhaps by giving an improved approximation ratio for MaxCut, Max 2-Sat, or VertexCover. This would immediately invalidate much of the work which is based on the UGC.

One (non-mathematical) reason to believe $P \neq NP$ is that despite thousands of possible problems to tackle and years of trying (not to mention a real monetary incentive) no one has come across a polynomial time algorithm for any NP-hard problem. By contrast, the advances in approximation algorithms which started with Goemans and Williamson [GW95] are relatively recent. Perhaps a radically new technique for devising approximation algorithms will be devised. Work on the PCP theory is even newer, especially in light of the recent reformulation of the PCP theorems due to Dinur [Din06]. With these new techniques and other advances in hand, it is conceivable that someone could prove that the Unique Games Conjecture (assuming $P \neq NP$) follows from the PCP theorems.

In very recent, independent work, Raghavendra resolves some of our conjectures for CSPs [Rag08]. It would be very interesting to apply Raghavendra's techniques in investigating balanced and imbalanced instances of some of these CSPs to see if Austrin's conclusion—balanced instances are not the hardest assuming the Unique Games Conjecture—still holds.

The larger picture of a connection between semidefinite programming and the Unique Games Conjecture is that if the UGC fully captures the power of semidefinite programming than this tells us something very deep about semidefinite programming, even if the Unique Games Conjecture turns out to be false. Further exploring these implications, and in particular delving more deeply into the appearance of geometrically-derived constants from SDPs (such as the Goemans-Williamson constant) in UG-hardness results could prove very fruitful.

Our work on Δ-Imbalanced Max 2-Sat leaves many open questions. The definition of Δ-Imbalanced Max 2-Sat used in this work and in [Aus07a] is quite restrictive. Austrin suggests the following possible extensions [Aus]:

1. The total weight on positive clauses is Δ and the total weight on negative clauses is $1 - \Delta$
2. Each variable occurs positively a Δ-fraction of the time
3. The total fraction of positive literals is Δ

Finding algorithmic and hardness results for each of these cases is an interesting open problem. Another open problem is to find an analytic proof of the fact that the worst case configurations have a simple form for the Δ-Imbalanced Max 2-Sat semidefinite program, as conjectured in Section 4.2.4. Finally, the idea of setting up imbalanced and balanced instances of problems, not just in CSPs, might prove fruitful in other areas. A very rough analogue in graph theory might be algorithms on trees, balanced or imbalanced.

In moving from the theoretical world of computational complexity theory, where the PCP theory is very important, to the more applied parts of computer science, in which Max 2-Sat can be solved in practice for most instances with a powerful enough computer, approximation algorithms bridge an important divide. As we have described at great length, they are theoretically of great interest. But they are also often useful in practice. The untamed world of heuristics and the separate but related study of average-case analysis both present important opportunities for applying the techniques of the study of approximation algorithms. Does the Unique Games Conjecture have implications for the intractability of algorithms that do well in the average case? How well does a heuristic based on semidefinite programming perform? If a problem has a relatively inefficient but optimal algorithm, how good (and under what metrics?) does an approximation algorithm have to be to beat it? These questions and more point the way towards future work, inspired by the connections between semidefinite programming and the Unique Games Conjecture.

Chapter 6

Acknowledgments

I would like to thank Professor Alex Samorodnitsky, my adviser at the Hebrew University in Jerusalem where I started this work in the summer of 2007, and Professors Nati Linial and Irit Dinur for discussions during my time there. I would also like to thank Professor Madhu Sudan of the Massachusetts Institute of Technology, my thesis adviser during the fall of 2007 and winter of 2008. Thanks to Swastik Kopparty for helpful discussions and to Per Austrin for answering my questions about his work, and thanks to those who read drafts of my thesis: my brother Dr. Abraham Flaxman, Jie Tang, Jean Yang, Shira Mitchell, and Yakir Reshef. A special thanks to my most fervent supporter, Jackie Granick.

Bibliography

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[Aro02] Sanjeev Arora. How NP got a new definition: a survey of probabilistically checkable proofs. *BEIJING*, 3:637, 2002.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of np. *J. ACM*, 45(1):70–122, 1998.

[Aus] Per Austrin. Re: thinking about unique games, sdp, etc.

[Aus07a] Per Austrin. Balanced max 2-sat might not be the hardest. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 189–197, New York, NY, USA, 2007. ACM Press.

[Aus07b] Per Austrin. Towards sharp inapproximability for any 2-csp. In *FOCS '07: IEEE Symposium on Foundations of Computer Science*, 2007.

[CKK+05] Shuchi Chawla, Robert Krauthgamer, Ravi Kumar, Yuval Rabani, and D. Sivakumar. On the hardness of approximating multicut and sparsest-cut. In *CCC '05: Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, pages 144–153, Washington, DC, USA, 2005. IEEE Computer Society.

[CST01] Pierluigi Crescenzi, Riccardo Silvestri, and Luca Trevisan. On weighted vs unweighted versions of combinatorial optimization problems. *Information and Computation*, 167(1):10–26, 2001.

[Din06] Irit Dinur. The PCP theorem by gap amplification. In *Proc. 38th ACM Symp. on Theory of Computing*, pages 241–250, 2006.

[Fel71] William Feller. *An introduction to probability theory and its applications*, volume II. John Wiley and Sons, Inc., 1971.

[FG95] Uriel Feige and Michel X. Goemans. Approximating the value of two prover proof systems, with applications to Max 2Sat and Max DiCut. In *Proceedings of the Third Israel Symposium on Theory of Computing and Systems*, pages 182–189, 1995.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995.

[Has97] Johan Hastad. Some optimal inapproximability results. In *STOC '97: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 1–10, New York, NY, USA, 1997. ACM.

[Kho02] Subhash Khot. On the power of unique 2-prover 1-round games. In *STOC '02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, New York, NY, USA, 2002. ACM Press.

[KKMO04] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS'04)*, pages 146–154, Washington, DC, USA, 2004. IEEE Computer Society.

[KO06] Subhash Khot and Ryan O'Donnell. Sdp gaps and ugc-hardness for maxcutgain. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 217–226, Washington, DC, USA, 2006. IEEE Computer Society.

[KR03] Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-ϵ. In *Proc. of 18th IEEE Annual Conference on Computational Complexity (CCC)*, pages 379–386, 2003.

[KV05] Subhash Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into l_1. In *FOCS*, pages 53–62, 2005.

[KZ97] Howard Karloff and Uri Zwick. A 7/8-approximation algorithm for MAX 3SAT? In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science, Miami Beach, FL, USA*. IEEE Press, 1997.

[LLZ02] Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the max 2-sat and max di-cut problems, 2002.

[LS91] László Lovász and Alexander Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(2):166–190, 1991.

[O'D05] Ryan O'Donnell. A history of the pcp theorem. In *http://www.cs.washington.edu/education/courses/533/05au/pcp-history.pdf*, 2005.

[OW07] Ryan O'Donnell and Yi Wu. An optimal sdp algorithm for max-cut, and equally optimal long code tests. In *Manuscript*, 2007.

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *STOC '08: Proceedings of the fortieth annual ACM symposium on Theory of computing*, New York, NY, USA, 2008. ACM Press.

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.