Recent Progress in Applying Semidefinite Optimization to Satisfiability Problems

Miguel F. Anjos
Department of Management Sciences
University of Waterloo, Canada

Research partially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

MOPTA 05 – University of Windsor – 26 July 2005

The Satisfiability Problem (SAT)

- SAT instances arise from many areas: artificial intelligence, circuit testing, software verification, etc.
- It is known that SAT is in general NP-complete, although several important special cases can be solved in polynomial time.
- We consider the SAT problem in conjunctive normal form (CNF).

Formal Definition of SAT

An instance of SAT is specified by a set of n variables $x_1, \ldots, x_n$ and a propositional formula

$$\Phi = \bigwedge_{j=1}^{m} C_j$$

with m clauses, each of the form

$$C_j = \bigvee_{k \in I_j} x_k \vee \bigvee_{k \in \bar{I}_j} \bar{x}_k,$$

where $I_j, \bar{I}_j \subseteq \{1, \ldots, n\}$, $I_j \cap \bar{I}_j = \emptyset$, $|I_j \cup \bar{I}_j| \geq 2$, and $\bar{x}_i$ denotes the negation of x_i.

Formal Definition of SAT (ctd)

The SAT problem is :

Given an instance of SAT, is Φ satisfiable?

i.e.

is there a truth assignment to the variables $x_1, \ldots, x_n$ such that Φ evaluates to TRUE?

Example for this talk

$$(x_1 \vee x_2) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$$

Semidefinite Programming (SDP)

We assume that we have an efficient algorithm to solve semidefinite programming problems.

A primal-dual pair for SDP has the form:

$$\begin{array}{ll|ll} \max & C \cdot X & \min & b^T y \\ \text{s.t.} & A_i \cdot X = b_i,\ i = 1, \ldots, m & \text{s.t.} & Z = \sum_{i=1}^{m} y_i A_i - C \\ & X \succeq 0 & & Z \succeq 0 \end{array}$$

where $A \cdot B = \sum_{i,j} A_{i,j} B_{i,j} = \mathrm{trace}\,(B^T A)$.

Note the similarity with linear programming, except that the "non-negativity" constraint $X \succeq 0$ denotes that X is a real, symmetric, positive semidefinite matrix.

Previous Applications of SDP to SAT

- SDP-based approximation algorithms:
  - for MAX-2-SAT by Goemans and Williamson (1994)
  - for MAX-3-SAT by Karloff and Zwick (1997)
  - for MAX-4-SAT by Halperin and Zwick (1999)
- Gap relaxation of de Klerk, van Maaren and Warners (2000), which is based on the elliptic approximations of clauses by van Maaren (1999).
- Partial higher liftings of Anjos (2004,2005).

Useful Definitions

For clause $C_j = \bigvee_{k \in I_j} x_k \vee \bigvee_{k \in \bar{I}_j} \bar{x}_k$ and $k \in I_j \cup \bar{I}_j$, define

$$s_{j,k} := \begin{cases} 1, & \text{if} \quad k \in I_j \\ -1, & \text{if} \quad k \in \bar{I}_j \end{cases}$$

Let 1 denote TRUE and -1 denote FALSE.

Then

$$\begin{aligned} \text{clause } C_j \text{ is satisfied} \quad &\Leftrightarrow \quad s_{j,k} x_k = 1 \text{ for some } k \in I_j \cup \bar{I}_j \\ &\Leftrightarrow \quad \prod_{k \in I_j \cup \bar{I}_j} (1 - s_{j,k} x_k) = 0. \end{aligned}$$

Gap Relaxation for our Example

$$(x_1 \vee x_2) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$$

$$\begin{array}{ll} \text{find} & X \succeq 0 \\ \text{s.t.} & X_{1,2} - X_{0,1} - X_{0,2} + 1 = 0 \\ & X_{2,4} - X_{3,4} - X_{2,3} - X_{0,2} + X_{0,3} - X_{0,4} \leq 0 \end{array}$$

$$X = \begin{pmatrix} 1 & X_{0,1} & X_{0,2} & X_{0,3} & X_{0,4} \\ X_{0,1} & 1 & X_{1,2} & X_{1,3} & X_{1,4} \\ X_{0,2} & X_{1,2} & 1 & X_{2,3} & X_{2,4} \\ X_{0,3} & X_{1,3} & X_{2,3} & 1 & X_{3,4} \\ X_{0,4} & X_{1,4} & X_{2,4} & X_{3,4} & 1 \end{pmatrix}$$

Some Properties of the Gap Relaxation

- It characterizes satisfiability for 2-SAT problems, i.e. the Gap relaxation is feasible if and only if the corresponding 2-SAT instance is satisfiable. (2-SAT is known to be solvable in polynomial time.)
- It also characterizes satisfiability for certain classes of SAT problems, such as mutilated chessboard and pigeonhole instances.
- The authors also analyzed rounding schemes and approximation guarantees for the Gap relaxation, as well as its behaviour on so-called $(2+p)$-SAT problems.

However, the Gap relaxation is always feasible when the SAT instance has no clauses of length less than three, and hence is unable to detect unsatisfiability for such instances.

How Can We Do Better?

Note that the rows and columns of the matrix variable in the Gap relaxation are indexed by the binary variables themselves:

$$X = \begin{pmatrix} 1 & X_{0,1} & \cdots & X_{0,n} \\ & 1 & & \\ & & \ddots & X_{i,j} \\ & & & 1 \end{pmatrix}$$

Tighter SDP relaxations can be obtained by using ideas from a *higher liftings* paradigm for constructing SDP relaxations of polynomial optimization problems:
consider semidefinite relaxations with the rows and columns of the matrix variable indexed by subsets of the set of variables.

Properties of Higher Liftings

These higher liftings

- have strong theoretical properties: Anjos-Wolkowicz (1999), Lasserre (2000,2001), Parrillo (2000), Laurent (2001,2002), Bienstock-Zuckerberg (2002).

  In particular, using all 2^n subsets means that we are optimizing over the convex hull of the ± 1 feasible solutions.

- but the size of the liftings grows very rapidly with the number of binary variables.

  As a consequence, only second liftings for Maximum-Cut problems with only up to, say, 30 binary variables can be solved in practice.

Partial Higher Liftings

We therefore consider partial higher liftings for which

1. both the dimension of the matrix variable and the number of linear constraints depend linearly on the size of the SAT instance; and
2. the structure of the SDP relaxation reflects the structure of the SAT instance.

The approach proposed in Anjos (2005) expresses satisfiability using

$$\prod_{i \in I_j \cup \bar{I}_j} (1 - s_{j,i} x_i) = 0$$

which, upon expanding and rearranging terms, equals

$$\sum_{t=1}^{l(C_j)} (-1)^{t-1} \left[\sum_{I \subseteq I_j \cup \bar{I}_j, |I|=t} \left(\prod_{i \in I} s_{j,i} \right) \left(\prod_{i \in I} x_i \right) \right] = 1.$$

Improved SDP Relaxation for our Example

$$(x_1 \vee x_2) \wedge (x_2 \vee \bar{x}_3 \vee x_4)$$

From the first clause, we have $x_1 + x_2 - x_1x_2 = 1$ which involves the terms x_1, x_2, and x_1x_2.

Similarly, from the second clause, we have
$x_2 - x_3 + x_4 + x_2x_3 - x_2x_4 + x_3x_4 - x_2x_3x_4 = 1$.

The resulting matrix variable Y has dimension 10, and conceptually its first row will be:

$$\begin{pmatrix} 1 & x_1 & x_2 & x_1x_2 & x_3 & x_4 & x_2x_3 & x_2x_4 & x_3x_4 & x_2x_3x_4 \end{pmatrix}$$

We index the first column by $\emptyset$, and the others by an appropriate subset of variables, so the off-diagonal elements of the first row are:

$Y_{\emptyset,\{1\}}, Y_{\emptyset,\{2\}}, Y_{\emptyset,\{1,2\}}, Y_{\emptyset,\{3\}}, Y_{\emptyset,\{4\}}, Y_{\emptyset,\{2,3\}}, Y_{\emptyset,\{2,4\}}, Y_{\emptyset,\{3,4\}}, Y_{\emptyset,\{2,3,4\}}$.

Improved SDP Relaxation for our Example (ctd)

For each clause, we add one equality constraint:

$$(x_1 \vee x_2) \quad \Leftrightarrow \quad x_1 + x_2 - x_1 x_2 = 1 \quad \Rightarrow \quad Y_{\emptyset,x_1} + Y_{\emptyset,x_2} - Y_{\emptyset,x_{12}} = 1$$

$$(x_2 \vee \bar{x}_3 \vee x_4) \quad \Leftrightarrow$$

$$x_2 - x_3 + x_4 + x_2x_3 - x_2x_4 + x_3x_4 - x_2x_3x_4 = 1 \quad \Rightarrow$$

$$Y_{\emptyset,x_2} - Y_{\emptyset,x_3} + Y_{\emptyset,x_4} + Y_{\emptyset,x_{23}} - Y_{\emptyset,x_{24}} + Y_{\emptyset,x_{34}} - Y_{\emptyset,x_{234}} = 1.$$

Finally, we also add some of the constraints equating first row entries to other entries of Y which should be equal to them, for example

$$Y_{\emptyset,\{2,3,4\}} = Y_{\{4\},\{2,3\}}.$$

We do this for all entries at the intersection of two subsets which arise from the same clause.

Conceptual Structure of the (Symmetric) Matrix Y

$$
Y = \begin{array}{c}
\begin{array}{cccccccccc}
\emptyset & \{1\} & \{2\} & \{1,2\} & \{3\} & \{4\} & \{2,3\} & \{2,4\} & \{3,4\} & \{2,3,4\}
\end{array} \\
\begin{pmatrix}
1 & x_1 & x_2 & x_{12} & x_3 & x_4 & x_{23} & x_{24} & x_{34} & x_{234} \\
 & 1 & x_{12} & x_2 & * & * & * & * & * & * \\
 & & 1 & x_1 & x_{23} & x_{24} & x_3 & x_4 & x_{234} & x_{34} \\
 & & & 1 & * & * & * & * & * & * \\
 & & & & 1 & x_{34} & x_2 & x_{234} & x_4 & x_{24} \\
 & & & & & 1 & x_{234} & x_2 & x_3 & x_{23} \\
 & & & & & & 1 & x_{34} & x_{24} & x_4 \\
 & & & & & & & 1 & x_{23} & x_3 \\
 & & & & & & & & 1 & x_2 \\
 & & & & & & & & & 1
\end{pmatrix}
\end{array}
$$

The asterisk elements are not involved in any of the linear equality constraints (but they are constrained by positive semidefiniteness).

Improved SDP Relaxation for our Example

find $\quad Y \in \mathcal{S}^{10}$

s.t.

$$Y_{\emptyset,x_1} + Y_{\emptyset,x_2} - Y_{\emptyset,x_{12}} = 1$$

$$Y_{\emptyset,x_2} - Y_{\emptyset,x_3} + Y_{\emptyset,x_4} + Y_{\emptyset,x_{23}} - Y_{\emptyset,x_{24}} + Y_{\emptyset,x_{34}} - Y_{\emptyset,x_{234}} = 1$$

Y with prescribed structure

$Y \succeq 0.$

Properties of the Improved SDP Relaxation

Theorem 1 *Given any propositional formula in CNF,*

- *If the SDP problem is infeasible, then the formula is unsatisfiable.*
- *If the SDP problem is feasible, and Y is a feasible matrix such that* $\operatorname{rank} Y \leq 3$, *then a truth assignment satisfying the formula can be obtained from Y.*

Computationally, the effort required by an SDP-based algorithm is often still too large for practical use.

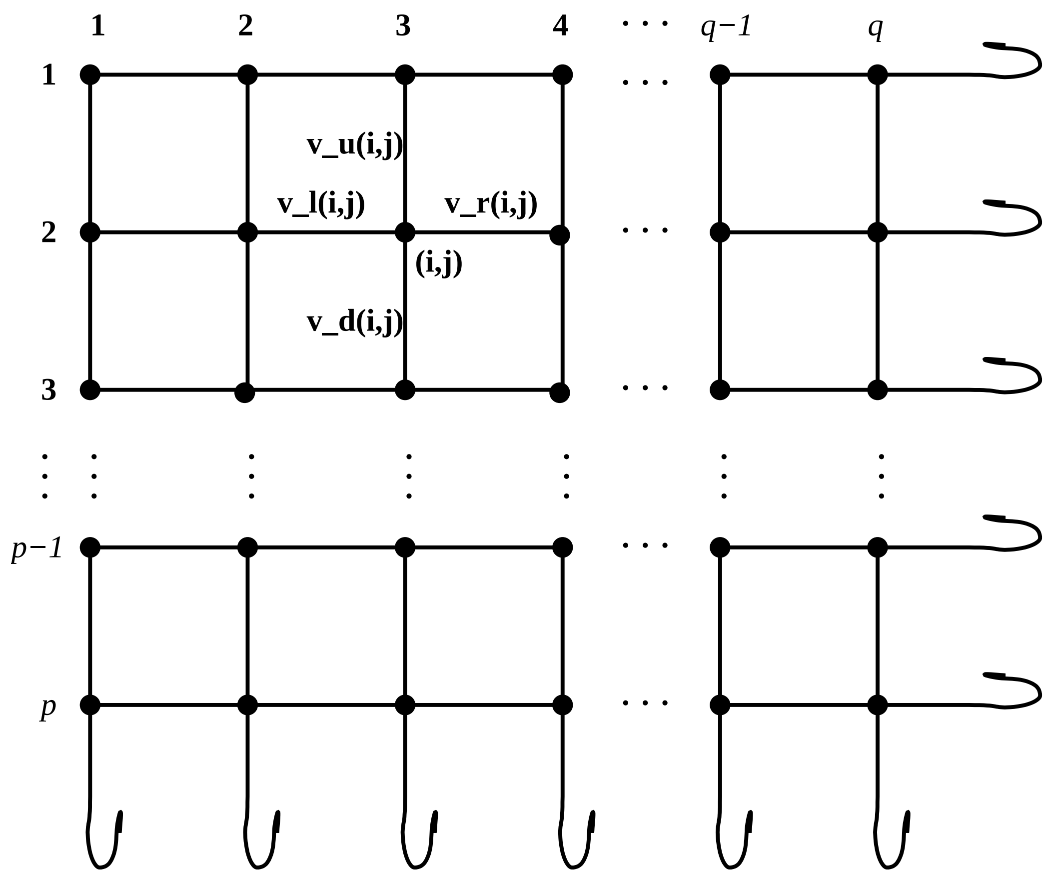
However, for some instances of SAT, the SDP approach was competitive with the top solvers in the SAT 2003 and SAT 2004 competitions.

Thus this approach has the potential to complement existing techniques for SAT.

The Tseitin Instances of SAT

They were introduced by Tseitin in 1968, and are known to be hard for many proof systems.

Consider a toroidal grid graph of the form



The Tseitin Instances of SAT (ctd)

For each node (i, j), set the parameter $t(i, j) = 0$ or 1.

Then, for each (i, j), define 8 clauses as follows:

If $t(i, j) = 0$:

$$
\begin{array}{lll}
\bar{v^l} \vee v^r \vee v^u \vee v^d & v^l \vee v^r \vee v^u \vee \bar{v^d} & \bar{v^l} \vee v^r \vee \bar{v^u} \vee \bar{v^d} \\
v^l \vee \bar{v^r} \vee v^u \vee v^d & \bar{v^l} \vee \bar{v^r} \vee \bar{v^u} \vee v^d & v^l \vee \bar{v^r} \vee \bar{v^u} \vee \bar{v^d} \\
v^l \vee v^r \vee \bar{v^u} \vee v^d & \bar{v^l} \vee \bar{v^r} \vee v^u \vee \bar{v^d} &
\end{array}
$$

If $t(i, j) = 1$:

$$
\begin{array}{lll}
v^l \vee v^r \vee v^u \vee v^d & \bar{v^l} \vee v^r \vee v^u \vee \bar{v^d} & v^l \vee v^r \vee \bar{v^u} \vee \bar{v^d} \\
\bar{v^l} \vee \bar{v^r} \vee v^u \vee v^d & v^l \vee \bar{v^r} \vee \bar{v^u} \vee v^d & \bar{v^l} \vee \bar{v^r} \vee \bar{v^u} \vee \bar{v^d} \\
\bar{v^l} \vee v^r \vee \bar{v^u} \vee v^d & v^l \vee \bar{v^r} \vee v^u \vee \bar{v^d} &
\end{array}
$$

Properties of the Tseitin Instances

It is not difficult to show that

the SAT instance is unsatisfiable

if and only if

$$\sum_{(i,j)} t(i,j) \text{ is odd.}$$

For a $p \times q$ grid and given values of $t(i,j)$, the SAT instance has

- $n = 2pq$ Boolean variables, and
- $m = 8pq$ clauses, all of length 4.

New Result

Theorem 2 *Given a Tseitin instance, we can write down an explicit SDP problem with matrix variable of dimension* $14pq$ *and* $23pq - 1$ *linear equality constraints such that*

the SDP problem is infeasible
if and only if
the SAT instance is unsatisfiable.

This is not the first proof that Tseitin instances can be solved in polynomial-time (see e.g. Grigoriev et al. (2002)).

However, the SDP approach not only proves this, but its construction is *explicitly* given, which is very useful from a practical point of view.

Furthermore, it also provides an explicit certificate of unsatisfiability (or satisfiability) in polynomial-time.

Ongoing Research

The ultimate goal of this research is a practical SDP-based algorithm for solving general satisfiability problems.

Ongoing research is

- from a theoretical point of view, considering extensions to other types of well-known hard instances, such as graph-colouring instances; and
- from a computational point of view, testing algorithmic heuristics suggested by insights from this new result.

For all the Details...

Miguel F. Anjos. An Improved Semidefinite Programming Relaxation for the Satisfiability Problem. *Mathematical Programming* 102(3):589–608, 2005.

Miguel F. Anjos. On Semidefinite Programming Relaxations for the Satisfiability Problem. *Mathematical Methods of Operations Research* 60(3):349–367, 2004.

Miguel F. Anjos. Semidefinite Programming Approaches for Satisfiability and Maximum-Satisfiability Problems.
To appear in the *Journal of Satisfiability, Boolean Modeling and Computation*, August 2005.

Miguel F. Anjos. An Explicit Semidefinite Characterization of Satisfiability for Tseitin Instances, May 2005.
Available (until publication) on Optimization Online:

http://www.optimization-online.org