Contents lists available at ScienceDirect
Discrete Applied Mathematics
journal homepage: www.elsevier.com/locate/dam

On semidefinite least squares and minimal unsatisfiability

Miguel F. Anjos[a,*], Manuel V.C. Vieira[b]

[a] GERAD & École Polytechnique de Montréal, Montréal, QC, Canada H3C 3A7
[b] Departamento de Matemática, Faculdade de Ciências e Tecnologia & CMA, Universidade Nova de Lisboa, Portugal

ARTICLE INFO

Article history:
Received 23 May 2014
Received in revised form 7 July 2016
Accepted 4 September 2016
Available online xxxx

Keywords:
Semidefinite programming
Satisfiability
Minimal unsatisfiability
Farkas' Lemma
Semidefinite least squares

ABSTRACT

This paper provides new results on the application of semidefinite optimization to satisfiability by studying the connection between semidefinite optimization and minimal unsatisfiability. We use a semidefinite least squares problem to assign weights to the clauses of a propositional formula in conjunctive normal form. We then show that these weights are a measure of the necessity of each clause in rendering the formula unsatisfiable, the weight of a necessary clause is strictly greater than the weight of any unnecessary clause. In particular, we show the following results: first, if a formula is minimal unsatisfiable, then all of its clauses have the same weight; second, if a clause does not belong to any minimal unsatisfiable subformula, then its weight is zero. An additional contribution of this paper is a demonstration of how the infeasibility of a semidefinite optimization problem can be tested using a semidefinite least squares problem by extending an earlier result for linear optimization. The connection between the semidefinite least squares problem and Farkas' Lemma for semidefinite optimization is also discussed.

1. Introduction

The Boolean satisfiability (SAT) problem is at the crossroads of several important areas, including logic, computer science, graph theory, and operations research. It has numerous practical applications in these fields and others, as documented in the Handbook [9]. The problem consists of determining whether or not it is possible to satisfy a given propositional formula by at least one assignment of the values true/false to the Boolean variables appearing in the formula. It is a famous result that SAT is in general NP-complete [16], and it is in general a challenging problem to detect that a SAT instance is unsatisfiable and to provide insight into its unsatisfiability.

Unsatisfiability can occur for multiple reasons, and explaining its causes is a key requirement in a number of practical applications. There is an important body of literature concerned with, given a propositional formula that is unsatisfiable, obtaining an unsatisfiable subformula, and proving guarantees on the size of computed subformulas, see e.g. [23,31,30,25]. Most of this work has focused on computing one or all minimal unsatisfiable subformulas (MUSs). In particular, Kullmann et al. [28] provided a differentiated analysis of the causes of unsatisfiability through a classification of single clauses based on the contribution of each clause to the causes of unsatisfiability. Their classification varies from clauses that are necessary to prove unsatisfiability to unusable clauses. The highest degree of necessity corresponds to necessary clauses, where a clause is said to be necessary if every resolution refutation of the given formula must use this clause.

Unsatisfiability can also be expressed using optimization. This is known at least since the pioneering work of Williams [37] and Blair et al. [10] on the connections between inference in propositional logic and integer linear programming. The

[*] Correspondence to: Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal, QC, Canada H3C 3A7.
E-mail addresses: anjos@stanfordalumni.org (M.F. Anjos), mvcv@fct.unl.pt (M.V.C. Vieira).

http://dx.doi.org/10.1016/j.dam.2016.09.008
0166-218X/© 2016 Elsevier B.V. All rights reserved.

first optimization-based approaches to SAT focused mostly on formulating SAT and maximum satisfiability (MAX-SAT) as 0/1 integer linear programming problems whose linear programming relaxations can then be solved efficiently [15]. For certain classes of instances, including Horn formulas and their generalizations, the exactness of the linear programming relaxation has been established, see e.g. [11].

Semidefinite optimization, or semidefinite programming (SDP) is the problem of optimizing a linear function of a matrix variable subject to linear constraints on its elements and the additional constraint that the matrix be positive semidefinite. The Handbooks [38,6] provide a wide coverage of SDP theory, algorithms, software, and application areas in which SDP has had a major impact. One of the best-known results in SDP is due to Goemans and Williamson who proposed SDP-based polynomial-time approximation schemes for MAX-CUT and MAX-2-SAT [24]. Further research has deepened the connections between SDP and the SAT and MAX-SAT problems, see the recent survey chapter [5].

The SAT problem can be formulated as an SDP problem with a rank-one constraint, and removing the rank constraint yields a convex optimization problem that is an SDP relaxation of SAT, see e.g. [2]. This can be done in different ways. In [20,21], de Klerk, van Maaren, and Warners introduced the Gap SDP relaxation and showed that it is exact for some well-known classes of SAT instances, in the sense that the Gap SDP is infeasible if and only if the SAT instance is unsatisfiable. Subsequent papers by Anjos [1–4] and van Maaren et al. [35,36] proposed several SDP relaxations for different versions of SAT, and some exactness results for particularly structured SAT formulas. An exact SDP relaxation for general SAT is obtained by formulating the SAT instance as a binary optimization problem and then constructing the corresponding Lasserre SDP relaxation [29]. Lasserre's theory proves that this SDP relaxation is always exact. However, because the size of the relaxation is exponential in the number of Boolean variables in the instance, it is computationally impractical for all but very small instances of SAT.

A more direct connection between SAT and SDP was given in our recent paper [8] where we proved that the process of resolution in SAT is equivalent to a linear transformation between the feasible sets of SDP relaxations. This equivalence between resolution and SDP, called *SDP resolution*, makes it possible to write a direct proof of the exactness of Lasserre's SDP relaxation in the specific context of SAT without recourse to Lasserre's general theory. The exactness proof in [8] shows that the exact relaxation implicitly deduces whether the empty clause can be derived by a finite sequence of resolution steps starting from the SAT formula. It then follows that the SDP relaxation is infeasible precisely when such a sequence of steps exists.

This paper provides new results on the application of SDP in the context of unsatisfiability. Specifically we focus here is on the connection between SDP and minimal unsatisfiability. A CNF formula (formally defined in Section 1.1) is minimal unsatisfiable (MU) if it is unsatisfiable but if any clause is removed then the resulting formula is satisfiable. We establish a connection between SDP and minimal unsatisfiability by using a semidefinite least squares (SLS) problem to associate non-negative weights to the clauses in a formula. We then argue that these weights are a measure of the importance of each clause in rendering the formula unsatisfiable. In particular, we prove that this approach identifies two important cases: first, if an unsatisfiable formula is MU, then all of its clauses have the same weight; second, if a clause does not belong to any minimal unsatisfiable subformula (MUS) of an unsatisfiable formula, then its weight is zero.

The computation of these weights is in general a hard problem. To show this, let F denote the set of clauses in a CNF formula such that $|F| \geq 2$, and for $C \subseteq F$, let $w(C)$ be a assignment of non-negative weights to the clauses in C. Consider the following decision problem (CW):

Given F, is $w(C)$ constant for $C \subseteq F$?

We argue that (CW) is NP-hard by following the idea of the reduction from SAT to minimal unsatisfiability in Lemma 2 of [34]. Specifically we build an unsatisfiable set of clauses G' by adding the clause Y to the clause-set G from the proof of Lemma 1 of [34], where G is MU if and only if F is unsatisfiable. Because Y has a clash with at least one clause of G, it follows that G' contains a clause that cannot be removed without destroying unsatisfiability. Thus, there is a polytime reduction that from F constructs G' such that:

- if F is satisfiable then G' is MU, and hence $w(C)$ is constant (Theorem 5.2 in this paper).
- if F is unsatisfiable then G' contains a redundant and a necessary clause, and hence $w(C)$ is not constant (Theorem 5.1 in this paper).

It follows that (CW) is NP-hard.

The motivation for this work is that, in spite of the hardness of the problem, the identification of MUSs remains a need in practice. The optimization problem (14) is a convex optimization problem that is in principle solvable in polynomial time except for the fact that the number of variables is exponential in the number of Boolean variables. Moreover, problem (14) has a structure that may be exploited in future to design a practical algorithm.

An additional contribution of this paper is an exploration of the use of SLS to test the infeasibility of an SDP problem. This is an extension of an earlier result of Dax [18] for linear optimization. We also explain the connection between the SLS problem and a semidefinite version of the well-known Farkas' Lemma.

Farkas' Lemma was used in the SAT context in [17] where a connection is made between a non-trivial solution of an homogeneous system and CNF formulas that are tautologies. The problem of efficiently deleting clauses that do not contribute to any proof of unsatisfiability was studied in [27]. Using a Farkas' Lemma variant, classes of formulas where selecting a MUS is easy were investigated in [13]. These and other results on unsatisfiability can be found in the major