Discrete Applied Mathematics 160 (2012) 61–80

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Estimating satisfiability

Yacine Boufkhad *, Thomas Hugel

LIAFA, CNRS UMR 7089, Université Denis Diderot Paris 7, Case 7014, F-75205 Paris Cedex 13, France

ARTICLE INFO

Article history:
Received 16 March 2011
Received in revised form 9 September 2011
Accepted 8 October 2011
Available online 27 October 2011

Keywords:
Constraint satisfaction problem
Satisfiability
First moment method

ABSTRACT

The problem of estimating the proportion of satisfiable instances of a given CSP (constraint satisfaction problem) can be tackled through weighting. It consists in putting onto each solution a non-negative real value based on its neighborhood in a way that the total weight is at least 1 for each satisfiable instance. We define in this paper a general weighting scheme for the estimation of satisfiability of general CSPs. First we give some sufficient conditions for a weighting system to be correct. Then we show that this scheme allows for an improvement on the upper bound on the existence of non-trivial cores in 3-SAT obtained by Maneva and Sinclair (2008) [17] to 4.419. Another more common way of estimating satisfiability is ordering. This consists in putting a total order on the domain, which induces an orientation between neighboring solutions in a way that prevents circuits from appearing, and then counting only minimal elements. We compare ordering and weighting under various conditions.

1. Introduction

Constraint satisfaction problems cover a large variety of problems that arise in many areas of combinatorial optimization. They are central in complexity theory because they are *NP*-complete and also because one particular case – satisfiability of Boolean formulas – was the first problem to be identified in this class. In general, they consist in defining constraints on a set of *variables* taking their *values* in a given finite domain. *Constraints* specify which combinations of values assigned to subsets of variables are allowed (or dually are forbidden). A *solution* is a valuation (i.e. the assignment of a value to each variable) that does not violate any constraint. The satisfiability problem is the following: given an instance, decide the existence of a solution for it.

Besides the design of algorithms for solving these problems, the research of structural properties for these problems has attracted much attention in the recent years. In particular, the empirical evidence of the existence of a threshold (rigorously established in some particular cases) in the satisfiability of some classes of CSPs has opened a field of research: attempts are made to rigorously establish the existence and the location of this threshold. This involves estimating the proportion of satisfiable instances in a given set of instances. The *NP*-completeness of these problems in general makes it difficult to determine whether a given instance is satisfiable; that may explain why direct counting of satisfiable instances is currently unfeasible. However, precisely because these problems are in *NP*, it is easy to determine whether some instance is satisfied by a given valuation and then to count the formulas satisfied by this valuation. Thus counting couples (formulas, solutions) is only accessible starting from a solution; moreover, given a solution, it is not complicated to investigate also its immediate neighborhood. But even at a distance of 2, i.e. with neighbors of neighbors, calculations become quite complicated (see [15]). This fact imposes a strong restriction on the design of both estimation techniques studied: they can only make use of local information. We shall refer to this as *the locality condition*.

* Corresponding author. Tel.: +33 1 44 27 54 19; fax: +33 1 44 27 68 49.
E-mail addresses: boufkhad@liafa.jussieu.fr (Y. Boufkhad), thomas.hugel@liafa.jussieu.fr (T. Hugel).

0166-218X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2011.10.005

Using one of the most popular techniques in the probabilistic method (cf. [5]), namely the first moment method, it is possible to bound from above the probability of satisfiability. The implementation of the first moment method makes use of Markov's inequality; one needs to define a non-negative random variable X that must be at least 1 for a satisfiable formula (we call that a *correct* random variable). Ideally, X should be as small as possible; in other words, it should be 0 for unsatisfiable instances and as close to 1 as possible for satisfiable ones (if X is 1 for every satisfiable instance and 0 for every unsatisfiable instance then we get the exact probability of satisfiability). The most straightforward candidate for X is simply the number of solutions. In order to compute the expected number of solutions for a random formula, it suffices to count for each valuation the number of instances that are satisfied by it and then to sum up over all valuations (by linearity of expectation). But since the number of solutions is generally too large, the method over-estimates the proportion of satisfiable instances.

Many techniques have been developed to overcome this difficulty in various types of CSPs: Satisfiability of CNF formulas [13,10,15,11,12,7,14,9,8]; 3-Coloring of graphs [1]; Binary CSPs [2,3] . . . . Most of these methods share a common point: they count minimal elements under some partial order over solutions. We will refer to this method as *solution selection through a partial ordering* or for short *ordering*. Due to the locality condition, the partial order must be locally computable (i.e. must depend only on the immediate neighbors of the considered solution). Two solutions of some instance are neighbors if they disagree only on the value taken by one variable. Both solutions may be ordered using a predetermined order on the values for this particular variable in this particular instance. Finally we count only those solutions having minimal values for all their variables with respect to their neighbors.

Recently, Maneva et al. [16] introduced a novel approach for the Boolean satisfiability problem consisting in weighting trivaluations (over $\{0, 1, *\}$) and solutions depending on their neighborhood. While not originally intended to estimate the proportion of satisfiable instances (but rather to analyze some properties of Belief Propagation algorithms), it was nevertheless specifically used by Maneva and Sinclair [17] to estimate the probability of existence of non-trivial cores in random 3-SAT instances. The existence of non-trivial cores contains important information on the structure of the space of solutions; moreover it is related to the clustering that has been proved to exist in k-SAT for $k \geq 9$ Achlioptas and Ricci-Tersenghi [4]. Maneva and Sinclair [17] show that in the 3-SAT instances, non-trivial cores do not exist for ratios of clauses to variables greater that 4.453. To do so they use *valid trivaluations* (i.e. satisfying some properties related to Boolean satisfiability) and weight them according to their values and their neighborhood.

2. Overview of results

Our first result consists in giving some sufficient conditions to make a weighting scheme correct for the estimation of satisfiability on general CSPs (Theorem 9, *Weight Conservation Theorem*). Then we propose a general weighting scheme obeying these conditions (Theorem 15). This scheme is based on:

1. a *weighting seed* that expresses the relative importance of each value with respect to a variable and an instance; the seed is such that if all valuations were solutions, then their total weight would be exactly 1;
2. a *dispatching function* expressing how the weights of forbidden valuations are dispatched among solutions to insure that counting weighted solutions will yield at least 1 for any satisfiable instance.

We will refer to this method as a *solution weighting* or for short *weighting*.

In Theorem 21, we show that the estimation of satisfiability used by Maneva and Sinclair [17] can be improved upon by using a weighting scheme based on a 3-valued CSP and obeying the conditions of our *Weight Conservation Theorem* (which shows that these conditions are somehow relevant). Thanks to this weighting system, we improve on the upper bound on the existence of non-trivial cores to 4.419. We completely reuse the proof of Maneva and Sinclair [17] for our new weighting system, showing that the improvement on the value of the bound is indeed due to a better weighting system.

Till now the only way to compare ordering and weighting was to compute the estimations of satisfiability obtained by each of them on a certain set of instances and to choose the best one. We give some results comparing these two ways of estimating satisfiability in the following cases:

- weighting and ordering can be instance dependent when such syntactic properties as the number of occurrences of variables and values etc. can guide the design of weighting functions and orderings. We show that in the general case where the weighting function is instance dependent and when the weighting is *homogeneous* (i.e. when weighting seeds and dispatching functions are equal), weighting cannot be better than a well chosen instance dependent ordering (Theorem 32);
- in the case where ordering and weighting are instance independent (which is the case of problems where the values are indistinguishable like graph coloring for example) and in the case of sets of instances closed under value renaming (which is the case of almost all sets of instances considered in the literature), we show that weighting and ordering are equivalent on average (Theorem 38).

3. Framework

A *CSP* (Constraint Satisfaction Problem) is a triple $F = \langle X, D, C\rangle$ where X is a set of *variables* taking their values in the same finite *domain* D of values, and C is a set of *constraints*. A *constraint* is a couple $\langle \boldsymbol{x}, R\rangle$ where $\boldsymbol{x} \in X^k$ and $R \subseteq D^k$ for some integer k. R is interpreted as the tuples of allowed values. A *valuation* is a vector $v \in D^X$; access to coordinate $x \in X$ of v will