Theoretical Computer Science 526 (2014) 18–27

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

New exact algorithms for the 2-constraint satisfaction problem

Alexander Golovnev [a,1], Konstantin Kutzkov [b,*]

[a] New York University, United States
[b] IT University of Copenhagen, Denmark

ARTICLE INFO

Article history:
Received 25 August 2012
Received in revised form 20 October 2013
Accepted 11 January 2014
Communicated by V.Th. Paschos

Keywords:
Exact exponential time algorithms
Constraint satisfaction
Maximum satisfiability

ABSTRACT

Many optimization problems can be phrased in terms of constraint satisfaction. In particular MAX-2-SAT and MAX-2-CSP are known to generalize many hard combinatorial problems on graphs. Algorithms solving the problem exactly have been designed but the running time is improved over trivial brute-force solutions only for very sparse instances. Despite many efforts, the only known algorithm [28] solving MAX-2-CSP over n variables in less than $O^*(2^n)$ steps uses exponential space.
Several authors have designed algorithms with running time $O^*(2^{nf(d)})$ where $f:\mathbb{R}^+ \to (0,1)$ is a slowly growing function and d is the average variable degree of the input formula. The current best known algorithm for MAX-2-CSP [25] runs in time $O^*(2^{n(1-\frac{2}{d+1})})$ and polynomial space. In this paper we continue this line of research and design new algorithms for the MAX-2-SAT and MAX-2-CSP problems.
First, we present a general technique for obtaining new bounds on the running time of a simple algorithm for MAX-2-CSP analyzed with respect to the number of vertices from algorithms that are analyzed with respect to the number of constraints. The best known bound for the problem is improved to $O^*(2^{n(1-\frac{3}{d+1})})$ for $d \geqslant 3$. We further improve the bound for MAX-2-SAT, in particular for $d \geqslant 6$ we achieve $O^*(2^{n(1-\frac{3.677}{d+1})})$.
As a second result we present an algorithm with asymptotically better running time for the case when the input instance is not very sparse. Building on recent work of Feige and Kogan we derive an upper bound on the size of a vertex separator for graphs in terms of the average degree of the graph. We then design a simple algorithm solving MAX-2-CSP in time $O^*(2^{c_d n})$, $c_d = 1 - \frac{2\alpha \ln d}{d}$ for some $\alpha < 1$ and $d = o(n)$.

1. Introduction

1.1. Exponential time algorithms

A paradigm in computational complexity was to divide problems into efficiently solvable and hard problems, in more precise terms NP-hard or #P-hard. Since hard problems are unlikely to be solvable exactly in polynomial time, researchers started to design *approximation* algorithms such that one can efficiently find a solution which is reasonably good. However, for many hard problems, approximation algorithms are either inadequate or do not yield satisfactory results. For example, unless P = NP no polynomial time algorithm can approximate MAX-2-SAT with a factor better than 0.9546 [13].

* Corresponding author.
1 Part of this work was done when the first author was at St. Petersburg University of the Russian Academy of Sciences.

0304-3975/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.tcs.2014.01.010

Hard problems were deemed intractable and except for a few results [14,16,21,26], not much work was invested in the design of exponential time algorithms improving upon the naïve solution. However, the increase in computational power and available memory led to a shift in these attitudes. The last two decades have witnessed a growing interest in design and analysis of deterministic and randomized exponential time algorithms. New algorithms with considerably improved running times have increased the size of efficiently computable instances of hard problems. Notable examples include solving k-COLORABILITY for graphs on n vertices in time and space $O^*(2^n)$[2] independent of k [3] and k-SATISFIABILITY on n variables in time $O^*((\frac{2(k-1)}{k})^n)$ and polynomial space [23,24].

1.2. Problem statement

The maximum satisfiability problem (MAX-SAT) is: given a Boolean formula in conjunctive normal form (CNF), i.e. a conjunction of literal disjunctions, find the maximum number of simultaneously satisfiable clauses of this formula. MAX-2-SAT is a restricted version of MAX-SAT, where each clause contains at most two literals.

MAX-2-SAT is a special case of the maximum 2-constraint satisfaction problem (MAX-2-CSP). In the MAX-2-CSP problem one is given a graph $G=(V,E)$ along with sets of "weight" functions $S_v:\{0,1\}\to\mathbb{Z}$ for each vertex $v\in V$ and $S_e:\{0,1\}^2\to\mathbb{Z}$ for each edge $e\in E$. (We will assume that the weight functions can be evaluated and represented in polynomial time and space.) The goal is to find an assignment $\phi:V\to\{0,1\}$ maximizing the sum

$$\sum_{e=(v_1,v_2)\in E} S_e\bigl(\phi(v_1),\phi(v_2)\bigr)+\sum_{v\in V} S_v\bigl(\phi(v)\bigr). \tag{1}$$

It is easy to see that by associating variables with vertices and clauses with edges, MAX-2-SAT corresponds to the case when all functions from S_e are disjunctions and each vertex $v\in V$ is assigned a weight of 0 by S_v. In the following an *instance* of MAX-2-CSP will be described by the graph G and the set of weight functions $S=S_v\cup S_e$, and an instance of MAX-2-SAT will be given only by the Boolean formula F since the weight function does not need to be explicitly defined.

MAX-SAT and MAX-2-SAT are among the most famous NP-hard optimization problems generalizing many graph problems. As already mentioned, the problem is hard to approximate with a factor better than 0.9546. Moreover, there is no known algorithm for either problem with polynomial space and running time less than $O^*(2^n)$, i.e. a running time of the form $O^*((2-\varepsilon)^n)$ for a constant $\varepsilon>0$. The strong exponential time hypothesis [4,17] implies that MAX-SAT cannot be solved in time less than $O^*(2^n)$.

1.3. The main definitions

Let $G=(V,E)$ be an undirected graph.[3] In the following we give definitions only for MAX-2-CSP in terms of graph terminology. By associating vertices with Boolean variables and edges with 2-clauses, i.e. disjunctions of two variables or their negations, the definitions trivially extend to Boolean formulas in CNF form.

By $n(G), m(G)$ we denote, respectively, the number of vertices and the number of edges in G. The size of G is defined as $|G|=m(G)+n(G)$. By the degree $deg(x)$ of a vertex x we mean the number of edges incident to x. We say that a vertex y is the neighbor of a vertex x if there is an edge $(x,y)\in G$. By $\Delta(G)$ we denote the maximum vertex degree in G. $d(G)=2m/n$ is the average vertex degree. We omit G if it is clear from the context.

Note that in the case of MAX-2-CSP one can assume without loss of generality that the corresponding graph does not contain multiple edges (as any two parallel edges can be replaced by their "sum"). At the same time one cannot exclude multiple edges from a MAX-2-SAT graph by the same argument (e.g., the graph of a formula $(x\vee y)(\neg x\vee y)(y\vee z)$ has two edges between x and y).

By (n,Δ)-MAX-2-CSP we denote MAX-2-CSP problems restricted to instances in which each vertex has degree at most Δ. By $Opt(G,S)$ we denote the maximal value of (1) for (G,S) over all possible assignments $\phi:V\to\{0,1\}$. Similarly we define (n,Δ)-MAX-2-SAT for a Boolean formula F, and define $Opt(F)$ to be the maximal number of simultaneously satisfiable clauses of the formula F.

Let (G,S) be an instance of MAX-2-CSP, and v be a vertex in G. By $(G,S)[v]$, or simply $G[v]$ when clear from the context, we denote the instance resulting from replacing all occurrences of v by 1 in (G,S). Similarly, for $G[\neg v]$ we replace v by 0. The definition naturally transfers to MAX-2-SAT, clauses containing a given literal l or $\neg l$ are either satisfied or shortened, i.e., 2-clauses become 1-clauses and 1-clauses are removed. Recursively solving a MAX-2-CSP problem instance by considering the cases $G[v]$ and $G[\neg v]$ for a vertex v, is referred to as *splitting* or *branching* on v.

A *vertex cover* of a graph $G=(V,E)$ is a subset of vertices $C\subseteq V$ such that for each $(u,v)\in E$, $u\in C$ or $v\in C$ holds. The complement set $V\backslash C$ is an *independent set* in G. Two sets of vertices $U\subset V$, $W\subset V$ are called *pairwise independent* if $U\cap W=\emptyset$ and there exist no edges $(u,w)\in E$ with $u\in U$, $w\in W$. We will call a set $\mathcal{S}\subseteq V$ of vertices *balanced*

[2] The O^* notation ignores polynomial factors.
[3] Note that we allow loops and multiple edges.