Discrete Applied Mathematics 161 (2013) 558–572

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

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

Fast algorithms for MIN INDEPENDENT DOMINATING SET[☆]

N. Bourgeois [a], F. Della Croce [b], B. Escoffier [a,*], V.Th. Paschos [a,c]

[a] LAMSADE, CNRS and Université Paris-Dauphine, France

[b] D.A.I., Politecnico di Torino, Italy

[c] Institut Universitaire de France, France

ARTICLE INFO

Article history:
Received 15 November 2010
Received in revised form 24 October 2011
Accepted 9 January 2012
Available online 9 February 2012

Keywords:
MIN INDEPENDENT DOMINATING SET
Exponential algorithms
Exact algorithms
Approximation algorithms

ABSTRACT

We first devise a branching algorithm that computes a minimum independent dominating set with running time $O^*(1.3351^n) = O^*(2^{0.417n})$ and polynomial space. This improves upon the best state of the art algorithms for this problem. We then study approximation of the problem by moderately exponential time algorithms and show that it can be approximated within ratio $1+\epsilon$, for any $\epsilon > 0$, in a time smaller than the one of exact computation and exponentially decreasing with ϵ. We also propose approximation algorithms with better running times for ratios greater than 3 in general graphs and give improved moderately exponential time approximation results in triangle-free and bipartite graphs. These latter results are based upon a new bound on the number of maximal independent sets of a given size in these graphs, which is a result interesting per se.

1. Introduction

Given a graph $G(V, E)$, an independent set of G is a subset $S \subseteq V$ such that, for any $(v_i, v_j) \in S \times S$, $(v_i, v_j) \notin E$. A dominating set is a subset $D \subseteq V$ such that every vertex in $V \setminus D$ has at least one neighbor in D. The problems of determining a maximum size independent set and a minimum size dominating set, denoted by MAX INDEPENDENT SET and MIN DOMINATING SET, respectively, are two paradigmatic problems in complexity theory and in combinatorial optimization. An independent dominating set is an independent set that is maximal for inclusion, i.e., a set that is both an independent set and a dominating set. The MIN INDEPENDENT DOMINATING SET problem consists of determining a minimum size independent dominating set. In the literature, MIN INDEPENDENT DOMINATING SET is frequently referred to as the minimum maximal independent set problem.

MIN INDEPENDENT DOMINATING SET is known to be **NP**-hard [18]. It is very extensively studied in the framework of polynomial time approximation and turns out to be one of the hardest problems to approximate. For instance, it is shown to be **min PB**-complete, in [23,24] (where **min PB** is the class of minimization problems whose objective functions are bounded by a polynomial in the size of the input) and even inapproximable (in the standard approximation framework) within ratio $|V|^{1-\epsilon}$, for any $\epsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$ [20]. Also, in the differential approximation framework[1],it is shown to be

☆ Research supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010. An extended abstract of this article has been presented in Bourgeois et al. (2010) [7]. This full version contains improvements both for the exact and the approximation algorithms.

* Correspondence to: Lamsade, Université Paris-Dauphine, Place du Maréchal de Lattres de Tassigny 75775 Paris Cédex 16, France. Fax: +33 144054091.
E-mail addresses: bourgeois@lamsade.dauphine.fr (N. Bourgeois), federico.dellacroce@polito.it (F. Della Croce), escoffier@lamsade.dauphine.fr (B. Escoffier), paschos@lamsade.dauphine.fr (V.Th. Paschos).

[1] Here, the approximation performance of an algorithm is measured with respect to the so-called differential approximation ratio [14], defined by $(\omega(I) - m_{\mathtt{A}}(I, S))/(\omega(I) - \mathrm{opt}(I))$ where, for an instance I of a problem Π approximately solved by an algorithm $\mathtt{A}$, $m_{\mathtt{A}}(I, S)$ is the value of a solution S computed by $\mathtt{A}$ on I and $\omega(I)$ and $\mathrm{opt}(I)$ are the values of an optimal solution and a worst solution on I, respectively. In other words, the differential approximation ratio measures the relative position of the value of an approximate solution in the interval [optimal-value solution of I, worst-value feasible solution of I].

0166-218X/$ – see front matter
doi:10.1016/j.dam.2012.01.003

0-DAPX-complete [2], where **0-DAPX** is the class of optimization problems that are not differentially approximable within any ratio strictly greater than 0, in other words, for any problem of this class, any approximation algorithm has worst-case approximation ratio equal to 0.

Except polynomial time approximation, another way, that recently has received great attention from the theoretical computer science community, to cope with intractability of **NP**-hard problems is to design algorithms able to solve them to optimality with worst-case exponential running time that is provably as low as possible. Here, we study solution of MIN INDEPENDENT DOMINATING SET by such algorithms. For MIN INDEPENDENT DOMINATING SET, the trivial $O^*(2^{|V|})$ bound has been initially (see [22]) broken down to $O^*(3^{|V|/3}) = O^*(2^{0.529|V|})$ (notation $O^*(\cdot)$ is used to measure complexity of an algorithm ignoring polynomial factors) using a result presented in [25], namely that the number of maximal (for inclusion) independent sets in a graph is at most $3^{|V|/3}$. This result has been improved in [19] where, using a branch & reduce technique, an algorithm optimally solving MIN INDEPENDENT DOMINATING SET with running time $O^*(2^{0.441|V|}) = O^*(1.358^{|V|})$ is proposed. Let us note that the best known exact algorithms for MAX INDEPENDENT SET and MIN DOMINATING SET have worst-case running times $O^*(1.2125^{|V|})$ [8] and $O^*(1.4969^{|V|})$ [27], respectively.

In this paper, we first devise a branching algorithm that computes a minimum independent dominating set in general graphs with running time $O^*(1.3351^{|V|}) = O^*(2^{0.417|V|})$ and polynomial space. More precisely, in Section 2 the general recurrence of the exact branching algorithm is presented, while Sections 3 and 4 deal with the possible branchings that may occur in the proposed exact algorithm and discuss the related complexities. Section 5 proposes another way to handle **NP**-hard problems, that lays in the midway between polynomial time approximation and exact computation, the so-called moderately exponential time approximation. The goal of the issue moderately exponential time approximation algorithms is to explore approximability of highly inapproximable (in polynomial time) problems in superpolynomial or moderately exponential time. Roughly speaking, if a given problem is solvable in time say $O^*(\gamma^n)$ but it is **NP**-hard to approximate within some ratio r, we seek r-approximation algorithms with complexity – significantly – lower than $O^*(\gamma^n)$. This issue has already been considered for several other problems such as MINIMUM SET COVER [11,4], MIN COLORING [5], MAX INDEPENDENT SET and MIN VERTEX COVER [6], MIN BANDWIDTH [12,17], …Similar issues arise in the field of fixed-parameter complexity, where approximation notions have been introduced, for instance, in [15,10]. In this article, we tackle approximation of MIN INDEPENDENT DOMINATING SET by moderately exponential time algorithms and show that there exist $(1+\epsilon)$-approximations obtained in time $O^*(2^{0.417(1-\epsilon/168)n})$ for every $\epsilon \leqslant 5$. We also propose algorithms with significantly better running times for ratios greater than 3. We finally study the problem in triangle-free and bipartite graphs. We first give a general bound on the number of maximal independent sets of size at most k in these graphs, result of independent interest. Combining this result with the previous techniques leads to approximation algorithms with better running times.

In what follows, given a graph $G(V, E)$ and a vertex $v \in V$, we denote by $\text{opt}(G)$ the value of the optimum in G. Also, we denote by $N(v)$ the neighborhood of v, by $N[v] = N(v) \cup \{v\}$ the closed neighborhood of v, by $d(v) = |N(v)|$ the degree of v. For a subset $H \subset V$, we denote by $G[H]$ the subgraph of G induced by H. For $v \in H$, for some $H \subset V$, we denote by $d'_H(v)$ the degree of v in $G[H]$; when no confusion arises, we simplify notations using $d'(v)$ instead. For convenience, we set $N[H] = \{N[v] : v \in H\}$. We use δ and Δ to denote the minimum and maximum degree of G, respectively. For simplicity, we set $n = |V|$ and $m = |E|$; $T(n)$ stands for the maximum running time an algorithm requires to solve MIN INDEPENDENT DOMINATING SET in a graph containing at most n vertices.

Branch & reduce-based algorithms have been used for decades, and a classical analysis of their running times leading to worst case complexity upper bounds is now well-known. If one knows that computing a solution on an instance of size n amounts to computation of a solution on a sequence of p instances of respective sizes $n - k_1, \ldots, n - k_p$, one can write:

$$T(n) \leqslant \sum_{i \leqslant p} T(n - k_i) + q(n) \tag{1}$$

for some polynomial q. The running time $T(n)$ is bounded by $O^*(c^n)$, where c is the greatest real root of $1 = \sum_{i \leqslant p} x^{-k_i}$. This root is often called the contribution of the branching to overall complexity factor, or the branching factor. In the sequel, we will omit for simplicity to precise the additive polynomial term $q(n)$ in recurrence relations. Of course, it is possible that there does not exist only one single recurrence as in (1), but several ones, depending on the instance. In this case, the running time is never greater than what is needed to solve an instance where at every step we make a branching that has the highest possible branching factor, i.e., the largest solution of (1). This is actually not true anymore when doing *multiple branchings* such as "either we take a or not, and if we add a to the solution then we know that b has degree 3 in the remaining graph and one can make a very good branching on it …" Indeed, it might be the case that the global worst case of the multiple branching does not correspond to the worst cases of the single branchings it involves. In the sequel, we keep this remark in mind in order to compute worst case running times involving multiple branchings.

2. General recurrence

Following an idea proposed in [19], we partition the graph into "marked" and "free" vertices. Initially, all the vertices are free. Then, in a remaining instance (after some branching steps), a marked vertex is a vertex not yet dominated (by a vertex already chosen to be in the solution under construction) but for which we have already decided *not* to take it in the solution. Indeed, we generalize the problem at hand in the following way: "given a subset $W \subseteq V$ (W is the set of free