Computational Study on Dominating Set Problem of Planar Graphs

Marjan Marzban[1], Qian-Ping Gu[1], and Xiaohua Jia[2]

[1] School of Computing Science, Simon Fraser University, Burnaby BC Canada
{mmarzban,qgu}@cs.sfu.ca
[2] Department of Computer Science, City University of Hong Kong
csjia@cityu.edu.hk

Abstract. Recently, there has been significant theoretical progress towards fixed-parameter algorithms for the DOMINATING SET problem of planar graphs. It is known that the problem on a planar graph with n vertices and dominating number k can be solved in $O(c^{\sqrt{k}}n)$ time using tree/branch-decomposition based algorithms, where c is some constant. However there has been no computational study report on the practical performances of the $O(c^{\sqrt{k}}n)$ time algorithms. In this paper, we report computational results of Fomin and Thilikos algorithm which uses the branch-decomposition based approach. The computational results show that the algorithm can solve the DOMINATING SET problem of large planar graphs in a practical time for the class of graphs with small branchwidth. For the class of graphs with large branchwidth, the size of instances that can be solved by the algorithm in a practical time is limited to a few hundreds edges. The practical performances of the algorithm coincide with the theoretical analysis of the algorithm. The results of this paper suggest that the branch-decomposition based algorithms can be practical for some applications on planar graphs.

Keywords: PLANAR DOMINATING SET, branch-decomposition, fixed-parameter algorithms, data reduction, computational study.

1 Introduction

Given an undirected graph $G(V, E)$, a *k-dominating set* D of G is a subset of k vertices of G such that for every vertex $v \in V(G)$, either $v \in D$ or v is adjacent to a vertex $u \in D$. The *dominating number* of G, denoted by $\gamma(G)$, is the minimum k such that G has a k-dominating set. Given G and an integer k, The DOMINATING SET problem is to decide if $\gamma(G) \leq k$. The optimization version of the problem is to find a dominating set D with $|D| = \gamma(G)$. The DOMINATING SET problem is a core NP-complete problem in combinatorial optimization and graph theory [17]. It also has wide practical applications such as resource allocations [21], domination problems in electric networks [19], and wireless ad hoc networks [33]. The books of Haynes et al. give a survey on the rich literature of algorithms and complexity of the DOMINATING SET

B. Yang, D.-Z. Du, and C.A. Wang (Eds.): COCOA 2008, LNCS 5165, pp. 89–102, 2008.

problem [20,21]. A recent experimental study on the heuristic algorithms for the DOMINATING SET problem can be found in [30].

The DOMINATING SET problem is NP-hard. Approximation algorithms and exact fixed-parameter algorithms have been extensively studied to tackle the intractability of the problem. A minimization problem P of size n is α-approximable if there is an algorithm which runs in polynomial time in n and produces a solution of P with value at most αOPT, where OPT is the value of the optimal solution of P and $\alpha \geq 1$. If P is $(1+\epsilon)$-approximable for every fixed $\epsilon > 0$, P is polynomial time approximable (i.e., has a PTAS). Problem P is fixed-parameter tractable if given a parameter k, OPT can be computed in $O(f(k)n^{O(1)})$ time, where $f(k)$ may be an exponentially fast (or faster) growing function in k. For arbitrary undirected graph G of n vertices, the DOMINATING SET problem is known $(1+\log n)$-approximable [22], but not approximable within a factor of $(1-\epsilon)\ln n$ for any $\epsilon > 0$ unless $NP \subseteq DTIME(n^{\log\log n})$ [15]. The problem is also known fixed-parameter intractable unless the parameterized complexity classes collapse [13,14]. If the problem is restricted to planar graphs, it is known as the PLANAR DOMINATING SET problem which is still NP-hard [17]. But the PLANAR DOMINATING SET problem is known polynomial time approximable [7] and fixed-parameter tractable [13].

In recent years, there have been significant improvements on the fixed-parameter algorithms for the PLANAR DOMINATING SET problem. Algorithms with running time $O(11^k n)$ [13] and $O(8^k n)$ [5] are known for graphs with $\gamma(G) = k$. The running time is further reduced and $O(c^{\sqrt{k}}n)$ time algorithms are known for a constant c [4,16,23]. Most of the sublinear exponent algorithms use a tree-decomposition based approach: First a tree decomposition of the given graph is computed and then a dynamic programming algorithm based on the tree-decomposition is used to compute a minimum dominating set. For a planar graph G with $\gamma(G) = k$, a tree decomposition of width $b\sqrt{k}$, b is a constant, can be computed and the dynamic programming part runs in $O(2^{2b\sqrt{k}}n)$ time [4]. One problem with those algorithms is that the constant $c = 2^{2b}$ is too large for solving the PLANAR DOMINATING SET problem in practice. In relation to treewidth and tree decompositions [27,28], Robertson and Seymour introduce branchwidth and branch decompositions [29]. Instead of a tree decomposition, a branch decomposition can be used in the above dynamic programming algorithms for the PLANAR DOMINATING SET problem. Fomin and Thilikos give such an algorithm (called FT Algorithm in what follows) which reduces the constant c to $2^{15.13}$ [16]. Dorn proposes an approach of applying the distance product of matrices to the dynamic programming step in branch/tree-decomposition based algorithms for the problem [11]. If the distance product of matrices is realized by the $O(n^\omega)$ ($\omega < 2.376$) time fast matrix multiplication method [10], the constant c in is improved to $2^{11.98}$. However the constant hidden in the Big-Oh may be huge. Dorn also proposes a tree-decomposition based algorithm for the problem [12]. Although expressed in terms of treewidth tw of G, the algorithm has time complexity $O(3^{tw} n^{O(1)})$, it has actually the same running time as that of FT Algorithm. An encouraging fact on branch decomposition is that an optimal branch decomposition of a planar graph can be computed in polynomial time [18,32]. This makes the branch-decomposition based algorithms receiving increasing attention for the problems on planar graphs.

Another important progress on the algorithmic tractability of the PLANAR DOMINATING SET problem is that the problem is shown having a linear size kernel [6]. More specifically, Alber et al. give an $O(n^3)$ time algorithm which, given a planar graph G with $\gamma(G) = k$, produces a reduced graph H (kernel) such that H has $O(k)$ vertices, $\gamma(H) = k' \leq k$, and a minimum dominating set of G can be constructed from a minimum dominating set of H in linear time [6]. In general, H and k' are smaller than G and k, respectively, since in the reduction process, a number of vertices in a minimum dominating set of H have been decided. This reduction process reduces the sublinear exponent from $c^{\sqrt{k}}$ to $c^{\sqrt{k'}}$ and thus improves the running time of the fixed-parameter algorithms for the PLANAR DOMINATING SET problem. This result is used in FT Algorithm which has three major steps [16]: Step I computes a kernel H of G by the data reduction process of [6] in $O(n^3)$ time. Step II finds an optimal branch decomposition of H with width $bw(H)$. This can be done by algorithms of [9,18] in $O(k^3)$ time. Step III computes a minimum dominating set D' of H using the dynamic programming method based on the branch decomposition in $O(2^{3\log_4 3bw(H)}k)$ time and constructs a minimum dominating set D of G from D' in linear time. It is proved in [16] that the branchwidth $bw(H) \leq 3\sqrt{4.5k'}$ and FT Algorithm has time complexity $O(2^{15.13k'}k + n^3)$. Alber et al. report that the data reduction computes a much smaller kernel in practice for a class of planar graphs [3,6]. Very recently, Bian et al. report that an optimal branch decomposition of a planar graph can be computed efficiently in practice [8,9]. These results provide the base for testing the practical efficiency of FT Algorithm for the PLANAR DOMINATING SET problem.

Although significant theoretical progresses have been made towards the fixed-parameter algorithms for the PLANAR DOMINATING SET problem, the authors are not aware of any report on the practical performances of these algorithms. In this paper, we report the computational study on FT Algorithm for the PLANAR DOMINATING SET problem. In our implementation of FT Algorithm, in addition to the data reduction rules of [3,6], we introduce new data reduction rules and use the recent works on planar branch decompositions. The new data reduction rules further reduce the kernel size and improve the running time of FT Algorithm. We have tested our implementation of FT Algorithm on several classes of planar graphs, including the maximal planar graphs and their subgraphs from LEDA [2,25], Delaunay triangulations of point sets taken from TSPLIB [26], triangulations and intersection graphs of segments from LEDA, Gabriel graphs, and planar graphs from PIGALE library [1]. The computational results show that the size of instances that can be solved in a practical time mainly depends on the branchwidth of the kernels. For example, the maximal planar graphs and their subgraphs have branchwidth at most four. This class of graphs are used as the test instances for the data reduction in previous studies [3,6]. Step I reduces the problem size significantly (often finds the solution already) and the PLANAR DOMINATING SET problem can be solved efficiently for very large instances in this class. On the other hand, for Delaunay triangulation and Gabriel graphs, because the branchwidth of kernels increases fast in instance size, the size of instances that can be solved in a practical time is limited to a few hundreds edges. For triangulation graphs, intersection graphs, and graphs from PIGALE library, the branchwidth of kernels increases slowly or does not grow in instance size, instances of size up to about ten thousands edges can be solved in a practical time. These results coincide with the theoretical analysis of FT Algorithm [16]: it runs exponentially in the branchwidth of the kernel and $k \geq b(bw(G))^2$ for some constant b. Because the kernel of G has $O(k)$ vertices, the analysis suggests that a large branchwidth of the instance implies a large kernel, Step I may not reduce the problem size much, and the kernel may have a large branchwidth. For a kernel H with large branchwidth, FT Algorithm is not practical because Step III of the algorithm runs exponentially in the branchwidth of H.

The results of this paper give a concrete example on using branch-decomposition based algorithms for solving important hard problems in planar graphs and show that the PLANAR DOMINATING SET problem can be solved in practice for some applications. This work may bring the theory of branch decomposition closer to practice.

The rest of the paper is organized as follows. In the next section, we review FT Algorithm. We introduce the data reduction rules in Section 3. Computational results of FT Algorithm are reported in Section 4. The final section concludes the paper.

2 Fomin and Thilikos Algorithm

We first introduce some definitions and terminology. Readers may refer to a textbook on graph theory (e.g., the one by West [34]) for basic definitions and terminology on graphs. In this paper, graphs are undirected unless otherwise stated. Let G be a graph with vertex set $V(G)$ and edge set $E(G)$. A *branch decomposition* of G is a tree T_B such that the set of leaves of T_B is $E(G)$ and each internal node of T_B has node degree three. For each link e of T_B, removing e separates T_B into two subtrees. Let E' and E'' be the sets of leaves of the subtrees. Let S_e be the set of vertices of G incident to both an edge of E' and an edge of E''. The width of e is $|S_e|$ and the width of T_B is the maximum width of all links of T_B. The *branchwidth* $bw(G)$ of G is the minimum width of all branch-decompositions. We call a link $e = \{x, y\}$ a leaf link if one of x and y is a leaf node of T_B, otherwise an internal link. Notice that S_e is a set which separates G into two subgraphs induced by edges of E' and E'', respectively.

We say a vertex u is dominated by a vertex v if u and v are adjacent. A vertex set U is dominated by a vertex set V if for every vertex $u \in U$ there is a vertex $v \in V$ such that u and v are adjacent or $u \in V$. Given two graphs G and H, we say $size(H) \leq size(G)$ if $|V(H)| \leq |V(G)|$ and $|E(H)| \leq |E(G)|$. In the rest of the paper, the PLANAR DOMINATING SET problem is used for the optimization version of the problem unless otherwise stated.

Now we briefly review FT Algorithm. Readers may refer to [16] for more details. FT Algorithm solves the PLANAR DOMINATING SET problem of G in three steps. Step I computes a kernel H of G by the data reduction process such that $size(H) \leq size(G)$, $\gamma(H) \leq \gamma(G)$, and a minimum dominating set D of G can be computed from a minimum dominating set D' of H in linear time. Step II finds an optimal branch decomposition T_B of H. Step III computes a minimum dominating set D' of H using the dynamic programming method based on T_B and constructs a minimum dominating set D of G from D'.

In Step I, the principle of data reduction introduced in [6] is that based on some rules we check the vertices of G to decide if some vertices can be included into D or excluded for computing D. More specifically, each vertex v of G is colored by black or grey as follows. Initially, every vertex v is colored grey, meaning that whether v should be included in D or not has not been decided. If v has been decided to be included in D, v is colored black. If v has been decided to be excluded for computing D in the future, v is removed from G. After the reduction process, we get a kernel $H(B \cup C, E)$, where B and C are the sets of black and grey vertices, respectively. The specific reduction rules will be introduced in the next section.

To compute an optimal branch decomposition T_B of H, either the edge-contraction algorithms [18,32] or the divide-and-conquer algorithms [9] can be used. The divide-and-conquer algorithms are faster for large graphs in practice.

In Step III, given a kernel $H = (B \cup C, E)$, we find a minimum $D' \subseteq (B \cup C)$ such that $D' \supseteq B$ and D' dominates all vertices of C. As shown later, a minimum dominating set D of G can be constructed from D' in linear time. To compute D', first the branch decomposition T_B of H is converted into a rooted binary tree by replacing a link $\{x, y\}$ of T_B by three links $\{x, z\}, \{z, y\}$, and $\{z, r\}$, where z and r are new nodes to T_B, r is the root, and $\{z, r\}$ is an internal link. For every internal link e of T_B, e has two children links incident to e. For every link e of T_B, let T_e be the subtree of T_B consisting of all descendant links of e. Let H_e be the subgraph of H induced by the edges at leave nodes of T_e. To compute a minimum dominating set D' of H, we find all dominating sets (solutions) of H_e from which D' may be constructed for every link e of T_B by a dynamic programming method: the solutions of H_e for each leaf link e is computed by enumeration and the solutions for an internal link e is computed by merging the solutions for the children links of e. To find a solution of H_e, each vertex of S_e is colored by one of the following colors.

Black. denoted by 1, meaning that the vertex is included in the dominating set.

White. denoted by 0, meaning that the vertex is dominated at the current step of the algorithm and is not in the dominating set.

Grey. denoted by $\hat{0}$, meaning that we have not decided to color the vertex into black or white yet at the current step.

A solution of H_e subject to a coloring $\lambda \in \{0, \hat{0}, 1\}^{|S_e|}$ is a minimum set $D_e(\lambda)$ satisfying

– for $u \in B \cap S_e$, $\lambda(u)$ is black;
– every vertex of $V(H_e) \setminus S_e$ is dominated by a vertex of $D_e(\lambda)$; and
– for every vertex $u \in S_e$ if $\lambda(u)$ is black then $u \in D_e(\lambda)$, if $\lambda(u)$ is white then $u \notin D_e(\lambda)$ and u is dominated by a vertex of $D_e(\lambda)$.

Intuitively, $D_e(\lambda)$ is a minimum set to dominate the vertices of H_e with grey vertices removed, subject to the condition that the vertices of S_e are colored by λ.

For a leaf link e, colorings λ and sets $D_e(\lambda)$ are computed by enumeration. An internal link e has children edges e_1 and e_2 in T_B. The colorings λ of S_e and sets $D_e(\lambda)$ are computed from the colorings λ_1 of S_{e_1}, sets $D_{e_1}(\lambda_1)$, colorings λ_2 of S_{e_2}, and sets $D_{e_2}(\lambda_2)$. A coloring λ of S_e is formed from λ_1 and λ_2 if:

– For $u \in S_e \setminus S_{e_2}$, $\lambda(u) = \lambda_1(u)$.
– For $u \in S_e \setminus S_{e_1}$, $\lambda(u) = \lambda_2(u)$.
– For $u \in S_e \cap S_{e_1} \cap S_{e_2}$, if $\lambda_1(u) = \lambda_2(u) = 1$ then $\lambda(u) = 1$; if $\lambda_1(u) = \lambda_2(u) = \hat{0}$ then $\lambda(u) = \hat{0}$; and if $\lambda_1(u) = 0$ and $\lambda_2(u) = \hat{0}$, or $\lambda_1(u) = \hat{0}$ and $\lambda_2(u) = 0$ then $\lambda(u) = 0$.
– For $u \in (S_{e_1} \cup S_{e_2}) \setminus S_e$, $\lambda_1(u) = \lambda_2(u) = 1$, or $\lambda_1(u) = 0$ and $\lambda_2(u) = \hat{0}$, or $\lambda_1(u) = \hat{0}$ and $\lambda_2(u) = 0$.

For a coloring λ of S_e formed from λ_1 and λ_2, the minimum dominating set $D_e(\lambda)$ is the minimum set among the sets of $D_{e_1}(\lambda_1) \cup D_{e_2}(\lambda_2)$. For $e = \{z, r\}$, a minimum set $D_e(\lambda)$ among all colorings λ of S_e is a minimum dominating set of H.

3 Data Reduction

In this section, we introduce the data reduction rules used in our implementation of FT Algorithm for Step I. All reduction rules of [3,6] are used. To enhance the data reduction effect, we also propose some new reduction rules. Following the convention of FT Algorithm, we color each vertex of G by black or grey, and may remove some vertices from G by those reduction rules. After the data reduction step, we get a kernel $H(B \cup C, E)$, recall that B and C are the sets of black and grey vertices, respectively. For a vertex v, let $N(v) = \{u | \{u, v\} \in E(G)\}$, $N[v] = N(v) \cup \{v\}$, $B(v) = B \cap N(v)$, and $C(v) = C \cap N(u)$. For a set U of vertices, let $N(U) = \cup_{v \in U} N(v)$. For a vertex u, if there is a black vertex $v \in N[u]$, we mark u *dominated*. Initially, every vertex of G is unmarked. In the data reduction step, some vertices are marked. Let X be the set of marked vertices and Y be the set of unmarked vertices. For $v \in V(G)$, the following is introduced in [6]:

$$N_1(v) = B(v) \cup \{u | u \in C(v), N(u) \setminus N[v] \neq \emptyset\},$$
$$N_2(v) = \{u | u \in N(v) \setminus N_1(v), N(u) \cap N_1(v) \neq \emptyset\}, \text{ and}$$
$$N_3(v) = N(v) \setminus (N_1(v) \cup N_2(v)).$$

Rule 1 [6]. For $v \in V(G)$, if $N_3(v) \cap Y \neq \emptyset$ then remove $N_2(v)$ and $N_3(v)$ from G, color v black, and mark $N[v]$ dominated.

For a pair of vertices $v, w \in V(G)$, let $N(v, w) = N(v) \cup N(w) \setminus \{v, w\}$, $B(v, w) = B \cap N(v, w)$, $C(v, w) = C \cap N(v, w)$, and $N[v, w] = N[v] \cup N[w]$. The following is introduced in [6]:

$$N_1(v, w) = B(v, w) \cup \{u | u \in C(v, w), N(u) \setminus N[v, w] \neq \emptyset\},$$
$$N_2(v, w) = \{u | u \in N(v, w) \setminus N_1(v, w), N(u) \cap N_1(v, w) \neq \emptyset\},$$
$$N_3(v, w) = N(v, w) \setminus (N_1(v, w) \cup N_2(v, w)).$$

Rule 2 [6]. For $v, w \in V(G)$ with both v and w grey, assume that $|N_3(v, w) \cap Y| \geq 2$ and $N_3(v, w) \cap Y$ can not be dominated by a single vertex of $N_2(v, w) \cup N_3(v, w)$.

Case 1: $N_3(v, w) \cap Y$ can be dominated by a single vertex of $\{v, w\}$.

- (1.1) If $N_3(v, w) \cap Y \subseteq N(v)$ and $N_3(v, w) \cap Y \subseteq N(w)$ then remove $N_3(v, w)$ and $N_2(v, w) \cap N(v) \cap N(w)$ from G and add new gadget vertices z and z' with edges $\{v, z\}, \{w, z\}, \{v, z'\}$, and $\{w, z'\}$ to G.
- (1.2) If $N_3(v, w) \cap Y \subseteq N(v)$ but $N_3(v, w) \cap Y \not\subseteq N(w)$ then remove $N_3(v, w)$ and $N_2(v, w) \cap N(v)$ from G, color v black, and mark $N[v]$ dominated.
- (1.3) If $N_3(v, w) \cap Y \subseteq N(w)$ but $N_3(v, w) \cap Y \not\subseteq N(v)$ then remove $N_3(v, w)$ and $N_2(v, w) \cap N(w)$ from G, color w black, and mark $N[w]$ dominated.

Case 2: If $N_3(v, w) \cap Y$ can not be dominated by a single vertex of $\{v, w\}$ then remove $N_2(v, w)$ and $N_3(v, w)$ from G, mark v and w black, and mark $N[v, w]$ dominated.

In Rule 1 and Rule 2 (Cases 1.2, 1.3, and 2) of [6], gadget vertices are used to guarantee some vertices to be included in the solution set. In [3] the rules are implemented in a way that the vertices to be included in the solution set are removed. Our descriptions are slightly different from the previous ones: we do not use gadget vertices nor remove the vertices to be included to the solution set but color them black. Our descriptions allow us to have new reduction rules given below that may further reduce the size of the kernel.

Rule 3

3.1: For $v, w \in V(G)$ with v black and w grey, if $(N_3(v, w) \cap Y) \setminus N(v) \neq \emptyset$ then remove $N_2(v, w) \cup N_3(v, w)$, color w black, and mark $N[w]$ dominated; otherwise remove $(N_2(v, w) \cup N_3(v, w)) \cap N(v)$.

3.2: For $v, w \in V(G)$ with v grey and w black, if $(N_3(v, w) \cap Y) \setminus N(w) \neq \emptyset$ then remove $N_2(v, w) \cup N_3(v, w)$, color v black, and mark $N[w]$ dominated; otherwise remove $(N_2(v, w) \cup N_3(v, w)) \cap N(w)$.

3.3: For $v, w \in V(G)$ with both v and w black, remove $N_2(v, w) \cup N_3(v, w)$.

Lemma 1. *Given a graph G, let G' be the graph obtained by applying Rule 3 for $v, w \in V(G)$. Then $size(G') \leq size(G)$, $\gamma(G') \leq \gamma(G)$, and a minimum dominating set D' of G' that contains all black vertices of G' is a minimum dominating set of G that contains all black vertices of G.*

Proof: For $v, w \in V(G)$ with v black and w grey, assume that $(N_3(v,w) \cap Y) \setminus N(v) \neq \emptyset$. For $u \in (N_3(v,w) \cap Y) \setminus N(v)$ and x which dominates u, $x \in \{w\} \cup N_2(v,w) \cup N_3(v,w)$. Since $N(N_2(v,w) \cup N_3(v,w)) \subseteq N[v] \cup N[w]$, we should include w into D to dominate $(N_3(v,w) \cap Y) \setminus N(v)$. Therefore, we can remove $N_2(v,w) \cup N_3(v,w)$ from G. Assume that $(N_3(v,w) \cap Y) \setminus N(v) = \emptyset$. For $u \in (N_2(v,w) \cup N_3(v,w)) \cap N(v)$, u is dominated by v and $N(v) \cup N(u) \subseteq N(v) \cup N(w)$. This implies that we can at least include w rather than u to get D. At this point, we can not decide if we should include w into D or not because there might be a vertex x with $N(w) \subseteq N(x)$ that should be included in D. But we can exclude $(N_2(v,w) \cup N_3(v,w)) \cap N(v)$ from D. Since $(N_2(v,w) \cup N_3(v,w)) \cap N(v)$ is dominated by v, we can remove $(N_2(v,w) \cup N_3(v,w)) \cap N(v)$ from G. This completes the proof for (3.1).

The proof for (3.2) is a symmetric argument of that for (3.1).

For $v, w \in V(G)$ with both v and w black, since $N(N_2(v,w) \cup N_3(v,w)) \subseteq N[v] \cup N[w]$, we can remove $N_2(v,w) \cup N_3(v,w)$ from G. □

Rule 4 [3]
- **4.1:** Delete edges between vertices of X (vertices marked dominated).
- **4.2** If $u \in X$ has $|C(u)| \leq 1$ then remove u.
- **4.3** For $u \in X$ with $C(u) \cap Y = \{u_1, u_2\}$, if u_1 and u_2 are connected by a path of length at most 2 then remove u.
- **4.4** For $u \in X$ with $C(u) \cap Y = \{u_1, u_2, u_3\}$, if $\{u_1, u_2\}, \{u_2, u_3\} \in E(G)$ then remove u.

To perform the data reduction, we first apply Rule 1 for every vertex of G. Next for every pair of vertices v and w of G, we apply either Rule 2 or Rule 3 depending on the colors of v and w. Then we apply Rule 4. We repeat the above until Rules 1-4 do not change the graph. From the results of [6,3] on Rules 1,2, and 4, and Lemma 1, we have the following result.

Theorem 1. *Given a planar graph G, let $H(B \cup C, E)$ be the kernel obtained by applying the reduction rules described above and D' be a minimum vertex set of $H(B \cup C, E)$ such that $D' \supseteq B$ and D' dominates C. Then a minimum dominating set D of G can be constructed from D' in linear time.*

Given a planar graph G, let $H(B \cup C, E)$ be the kernel obtained from Step I, T_B be an optimal branch decomposition of H, and $l(H) = \max\{|C \cap S_e|, e \in E(T_B)\}$. It is shown in [6] that $H(B \cup C, E)$ can be computed in $O(n^3)$ time. T_B can be computed by either the edge-contraction algorithm [18] or a divide-and-conquer algorithm [9] in $O(|E(H)|^3)$ time. It is shown in [16] that Step III has time complexity $O(2^{3 \log_4 3l(H)}|E(H)|)$. Therefore, FT Algorithm takes $O(2^{3 \log_4 3l(H)}|E(H)| + n^3)$ time to solve the PLANAR DOMINATING SET problem. In what follows, we use $l(H)$ for the branchwidth of kernel H.

4 Computational Results

We implemented FT Algorithm and tested our implementation on six classes of planar graphs from some libraries including LEDA [2,25] and PIGALE [1]. LEDA generates two types of planar graphs. One type of graphs are the random maximal planar graphs and their subgraphs and the other type of graphs are the planar graphs based on some geometric properties, including the Delaunay triangulations and triangulations of points, and the intersection graphs of segments, uniformly distributed in a two-dimensional plane. Instances of Class (1) are the random maximal graphs and their subgraphs generated by LEDA. This class of instances have been used by Alber et al. in their study on the data reduction rules used in Step I [3,6]. Instances of Class (2) are Delaunay triangulations of point sets taken from TSPLIB [26]. Instances of Classes (3) and (4) are the triangulations and intersection graphs generated by LEDA, respectively. Instances of Class (5) are Gabriel graphs of the points uniformly distributed in a two-dimensional plane. Instances of Classes (2)-(5) are graphs based some geometric properties. The DOMINATING SET problem on those graphs has important applications such as the virtual backbone design of wireless networks [24]. Instances of Class (6) are random planar graphs generated by the PIGALE library [1]. PIGALE provides a number of planar graph generators. We used a function in the PIGALE library that randomly generates one of all possible 2-connected planar graphs with a given number of edges based on the algorithms of [31].

Step I of FT Algorithm is implemented as described in the previous section. To compute an optimal branch decomposition T_B, we use the divide-and-conquer algorithm [9]. In Step III, to save memory, we compute the colorings λ and sets $D_e(\lambda)$ for each link e of T_B in the postorder. Once the colorings λ and sets $D_e(\lambda)$ are computed for a link e, the solutions for the children links of e are discarded. We sort the tables for the colorings to have an efficient implementation of Step III. The computer used for testing has an AMD Athlon(tm) 64 X2 Dual Core Processor 4600+ (2.4GHz) and 4Gbyte memory. The operating system is SUSE Linux 10.2 and the programming language used is C++.

We report the computational results of FT Algorithm in Table 1. For Step I, we give the number $|B|$ of vertices of an optimal dominating set decided in the data reduction and the running time of the step. For Step II, we give the size $|E(H)|$ and branchwidth $l(H) = \max\{|C \cap S_e|, e \in E(T_B)\}$ of kernel H, and the running time of the step. For Step III, we give the dominating number $\gamma(G)$ obtained by FT Algorithm and the running time of the step. The running time is in seconds, and Steps I, II, and III have time complexities $O(|E(G)|^3)$, $O(|E(H)|^3)$, and $O(2^{3 \log_4 3l(H)}|E(H)|)$, respectively. We use the number of edges to express the size of an instance or a kernel.

It is easy to show that the instances of Class (1) have branchwidth at most four. These instances have small kernels and Step I is very effective. For the instances included in the table, $|B|$ is very close to $\gamma(G)$ (i.e., Step I finds most vertices in an optimal dominating set) and the kernels are much smaller than the original instances. For some smaller instances not reported in the table, Step I already finds optimal dominating sets. Because the kernels have small size and branchwidth, FT Algorithm is efficient for the instances in this class, for example, an optimal dominating set can be computed for large instances of size up to about 40,000 edges in about 20 minutes.

Table 1. Computational results of FT Algorithm for instances of Classes (1)-(6)

Class	Graph	$\|E(G)\|$	$bw(G)$	Step I		Step II			Step III		total
	G			$\|B\|$	time	$\|E(H)\|$	$l(H)$	time	$\gamma(G)$	time	time
(1)	max1500	4047	4	209	4	23	2	< 1	211	< 1	4
	max6000	7480	4	2214	55	32	2	< 1	2219	< 1	55
	max8000	13395	4	2186	336	194	3	< 1	2211	< 1	337
	max11000	28537	4	1679	815	208	4	1	1695	< 1	816
	max13500	38067	4	1758	1203	302	3	1	1779	< 1	1204
(2)	pr144	393	9	2	< 1	291	6	1	20	1	3
	ch130	377	10	0	< 1	377	10	1	21	12734	12735
	kroB150	436	10	0	< 1	436	10	1	23	43094	43095
	pr226	586	7	12	1	126	6	< 1	21	< 1	2
	pr299	864	11	1	1	824	11	1	47	392931	392933
(3)	tri1000	2980	7	69	10	1657	7	4	163	26	40
	tri2000	5977	8	136	56	3192	7	146	321	120	322
	tri3000	8976	8	209	87	4805	7	379	489	190	656
	tri4000	11969	9	252	251	6888	7	1667	653	413	2331
	tri5000	14969	8	384	285	7271	8	1547	804	915	2747
(4)	rand2000	3247	8	371	8	1219	7	1	548	14	23
	rand3000	4943	10	514	19	2093	8	3	806	173	195
	rand4000	6676	11	678	35	2956	8	4	1068	217	256
	rand5000	8451	11	755	57	4177	8	13	1315	363	433
	rand6000	10293	11	839	93	5598	9	25	1563	2933	3051
(5)	Gab100	182	7	3	< 1	162	7	< 1	24	5	6
	Gab200	366	8	3	< 1	344	8	1	47	192	193
	Gab300	552	10	5	< 1	516	10	32	70	28014	28046
(6)	p1277	2128	9	116	8	1353	9	14	323	1953	1975
	p2518	4266	9	329	31	1876	5	26	621	3	60
	p4206	7101	6	596	75	2901	5	7	1039	2	84
	p5995	10092	7	708	181	5142	5	20	1504	6	207
	p7595	12691	6	998	259	5702	5	16	1893	7	272

For Class (2) and (5), the branchwidth of instance increases fast in instance size (e.g., Class (2) instances rd400 of 1,183 edges and u2152 of 6,312 edges have branchwidth 17 and 31, respectively, Class (5) instances Gab500 of 932 edges and Gab2000 of 3,911 edges have branchwidth 12 and 26, respectively). For the instances tested, the kernel H of an instance G has the same branchwidth as that of G ($l(H) = bw(G)$) and has the same size as or only slightly smaller than that of G. The size of those instances for which the PLANAR DOMINATING SET problem can be solved in a practical time is limited to a few instances of size up to only a few hundreds edges. The computation time for Instances ch130, kroB150, and pr299 in Class (2) is significantly larger than that for Instances pr144 and pr226 in the same class. As shown in the table, this huge difference comes from the difference between the branchwidthes of kernels (Step III), the kernels of Instances ch130 and kro150 have branchwidth 10 while those of pr144 and pr226 have branchwidth 6. This coincides with the theoretical time complexity of FT

Algorithm which runs exponentially in $l(H)$. Similar difference is observed for Instances Gab100 and Gab300 of Class (5) as well.

For Classes (3), (4), and (6), the branchwidth of instance increases slowly or does not grow in instance size. The data reduction is effective for instances in these classes. For most instances, the kernel size is at most half of the instance size and the branchwidth of the kernel is usually smaller than that of the instance as well. Our data show that the PLANAR DOMINATING SET problem can be solved for instances in these classes of size up to about 10,000 edges in a practical time. For large instances, the size $|E(H)|$ of kernel H is also important to the running time of Step III. For example, FT Algorithm takes more time to solve Instance rand6000 than that for rand2000. The time difference comes from the differences of both $l(H)$ and $|E(H)|$.

Due to the page limit, Table 1 only contains the instances well scaled within some size ranges. We have tested FT Algorithm on more instances. The results are similar to those in Table 1, the running time mainly depends on $l(H)$ and then $|E(H)|$. For a kernel H with large $l(H)$, Step III is time consuming, because this step runs exponentially in $l(H)$. Our computational results suggest that it may not be practical to use FT Algorithm to solve the PLANAR DOMINATING SET problem of instances with $l(H) > 10$ on a PC with a CPU of about 3GHz (e.g., it takes more than 100 hours to solve the instance pr299 with 864 edges and $l(H) = 11$).

Both the theoretical analysis and computational study suggest that computing a kernel H with smaller $l(H)$ and $|E(H)|$ is a most effective way to improve the efficiency of FT Algorithm. For this purpose, we proposed new reduction rules (Rule 3). Recall that H is the kernel obtained by new reduction rules (Rules 1,2,3, and 4) and let H' be the kernel obtained by applying only the previous known reduction rules (Rules 1,2, and 4). Since all nodes colored black (resp. nodes deleted) by previous rules are also colored (resp. deleted) by new rules, $l(H) \leq l(H')$ and $|E(H)| \leq |E(H')|$. For Classes (2) and (5), $l(H) = l(H') = bw(G)$ for all instances testes and $|E(H)| = |E(H')| = |E(G)|$ for most instances, that is, the effect of data reduction is very limited. However, for instances in other classes, data reduction is effective and our new rules improve the efficiency of FT Algorithm. For instances of Classes (1),(3),(4), and (6), Table 2 shows the computational results of FT Algorithm when previous rules and new rules are used. In the table, t_{old} and t_{new} (resp. $|B'|$ and $|B|$) are the total running times (resp. the numbers of vertices in an optimal dominating set decided in Step I) when previous rules and new rules are used, respectively. The data show that $l(H) = l(H')$ and $|E(H)| < |E(H')|$ for most instances. The total running time is improved when new rules are used: $t_{new} < t_{old}$ for all instances in the table. The improvement is instance dependent and t_{new}/t_{old} varies from 48% to 97%. The average of t_{new}/t_{old} over the five instances of Class (1) is about 90%. Similarly, the averages of t_{new}/t_{old} for Classes (3),(4), and (6) are about 70%, 85%, and 90%, respectively. The improvement of the total running time is obtained mainly from Step III. The running time of Step I when new rules are used is about the

Table 2. The results of using new data reduction rules and without using the new rules in Step I

Class	Graph G	$\|E(G)\|$	$bw(G)$	Results without new rules				Results with new rules			
				$\|B'\|$	$\|E(H')\|$	$l(H')$	time	$\|B\|$	$\|E(H)\|$	$l(H)$	time
(1)	max1500	4047	4	209	23	2	5	209	23	2	4
	max6000	7480	4	2212	41	2	58	2214	32	2	55
	max8000	13395	4	2183	218	3	357	2186	194	3	337
	max11000	28537	4	1671	287	4	893	1679	208	4	816
	max13500	38067	4	1752	362	4	1294	1758	302	3	1204
(3)	tri1000	2980	7	63	1752	7	84	69	1657	7	40
	tri2000	5977	8	102	3787	7	490	136	3192	7	322
	tri3000	8976	8	175	5442	7	877	209	4805	7	656
	tri4000	11969	9	214	7541	7	2499	252	6888	7	2331
	tri5000	14969	8	333	8201	8	4118	384	7271	8	2747
(4)	rand2000	3247	8	361	1293	7	25	371	1219	7	23
	rand3000	4943	10	512	2120	8	216	514	2093	8	195
	rand4000	6676	11	669	3043	8	263	678	2956	8	256
	rand5000	8451	11	748	4254	8	474	755	4177	8	433
	rand6000	10293	11	832	5675	9	5586	839	5598	9	3051
(6)	p1277	2128	9	112	1371	9	2134	116	1353	9	1975
	p2518	4266	9	291	2139	5	67	329	1876	5	60
	p4206	7101	6	555	3189	5	91	596	2901	5	84
	p5995	10092	7	652	5508	5	297	708	5142	5	207
	p7595	12691	6	925	6159	5	281	998	5702	5	272

same as that when previous rules are used (instance dependent) and we omit the details here due to the page limit.

5 Concluding Remarks

We tested the practical performances of FT Algorithm on a wide range of planar graphs. The computational results coincide with the theoretical analysis of the algorithm, it is efficient for graphs with small branchwidth but may not be practical for graphs with large branchwidth. By a PC with a CPU of about 3GHz, it is possible to solve the PLANAR DOMINATING SET problem for graphs with the branchwidth of their kernels at most 10 in a few hours. Since FT Algorithm runs exponentially in the branchwidth $l(H)$ of a kernel H for a given graph, it is worth to develop more powerful data reduction rules to reduce $l(H)$. Another research direction is to develop heuristics to reduce $l(H)$ to compute approximate solutions for the PLANAR DOMINATING SET problem by branch-decomposition based algorithms. Those heuristics should provide solutions very close to the optima but runs faster than FT Algorithm for graphs with large branchwidth. It is also interesting to find heuristics which are efficient in practice and have guaranteed performance for the Planar Dominating Set problem.

Acknowledgement

The authors thank anonymous reviewers for constructive comments. The work was partially supported by NSERC Research Grant of Canada and Research Grant Council of Hong Kong (Project No. CityU 114307).

References

1. Public Implementation of a Graph Algorithm Library and Editor (2008), `http://pigale.sourceforge.net/`
2. The LEDA User Manual, Algorithmic Solutions, Version 4.2.1 (2008), `http://www.mpi-inf.mpg.de/LEDA/MANUAL/MANUAL.html`
3. Alber, J., Betzler, N., Niedermeier, R.: Experiments on data reduction for optimal domination in networks. In: Proc. of the International Network Optimization Conference (INOC 2003), pp. 1–6 (2003)
4. Alber, J., Bodlaender, H.L., Fernau, H., Kloks, T., Niedermeier, R.: Fixed parameter algorithms for dominating set and related problems on planar graphs. Algorithmca 33, 461–493 (2002)
5. Alber, J., Fan, H., Fellows, M., Fernau, H., Niedermeier, R.: Refined search tree technique for dominating set on planar graphs. In: Sgall, J., Pultr, A., Kolman, P. (eds.) MFCS 2001. LNCS, vol. 2136, pp. 111–122. Springer, Heidelberg (2001)
6. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial time data reduction for dominating set. Journal of the ACM 51(3), 363–384 (2004)
7. Baker, B.S.: Approximation algorithms for np-complete problems on planar graphs. Journal of ACM 41, 153–180 (1994)
8. Bian, Z., Gu, Q., Marzban, M., Tamaki, H., Yoshitake, Y.: Empirical study on branchwidth and branch decomposition of planar graphs. In: Proc. of the 9th SIAM Workshop on Algorithm Engineering and Experiments (ALENEX 2008), pp. 152–165 (2008)
9. Bian, Z., Gu, Q.-P.: Computing branch decomposition of large planar graphs. Technical report, SFU-CMPT-TR 2008-04, School of Computing Science, Simon Fraser University (2008)
10. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation 9, 251–280 (1990)
11. Dorn, F.: Dynamic programming and fast matrix multiplication. In: Azar, Y., Erlebach, T. (eds.) ESA 2006. LNCS, vol. 4168, pp. 280–291. Springer, Heidelberg (2006)
12. Dorn, F.: How to use planarity efficiently: new tree-decomposition based algorithms. In: Proc. of the 33rd International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2007). LNCS, vol. 4769, pp. 280–291 (2007)
13. Downey, R.G., Fellow, M.R.: Parameterized complexity. In: Monographs in Computer Science. Springer, Heidelberg (1999)
14. Downey, R.G., Fellows, M.R.: Fixed parameter tractability and completeness. Cong. Num. 87, 161–187 (1992)
15. Fiege, U.: A threshold of $\ln n$ for approximating set cover. Journal of ACM 45, 634–652 (1998)
16. Fomin, F.V., Thilikos, D.M.: Dominating sets in planar graphs: branch-width and exponential speed-up. SIAM Journal on Computing 36(2), 281–309 (2006)
17. Garey, M.R., Johnson, D.S.: Computers and Intractability, a Guide to the Theory of NP-Completeness. Freeman, New York (1979)

18. Gu, Q.P., Tamaki, H.: Optimal branch decomposition of planar graphs in $O(n^3)$ time. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 373–384. Springer, Heidelberg (2005)
19. Haynes, T.W., Hedetniemi, S.M., Hedetniemi, S.T., Henning, M.A.: Domination in graphs applied to electronic power networks. SIAM J. on Discrete Mathematics 15(4), 519–529 (2002)
20. Haynes, T.W., Hedetniemi, S.T., Slater, P.J.: Domination in graphs. In: Monographs and Textbooks in Pure and Applied Mathematics, vol. 209. Marcel Dekker, New York (1998)
21. Haynes, T.W., Hedetniemi, S.T., Slater, P.J.: Fundamentals of domination in graphs. In: Monographs and Textbooks in Pure and Applied Mathematics, vol. 208. Marcel Dekker, New York (1998)
22. Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences 9, 256–278 (1974)
23. Kanj, I.A., Perkovic, L.: Improved parameterized algorithms for planar dominating set. In: Diks, K., Rytter, W. (eds.) MFCS 2002. LNCS, vol. 2420, pp. 399–410. Springer, Heidelberg (2002)
24. Li, X.-Y.: Algorithmic, geometric and graphs issues in wireless networks. Journal of Wireless Communications and Mobile Computing (WCMC) 6(2), 119–140 (2003)
25. Mehlhorn, K., Näher, S.: LEDA: A Platform for Combinatorial and Geometric Computing. Cambridge University Press, New York (1999)
26. Reinelt, G.: TSPLIB-A traveling salesman library. ORSA J. on Computing 3, 376–384 (1991)
27. Robertson, N., Seymour, P.D.: Graph minors I. Excluding a forest. Journal of Combinatorial Theory Series B 35, 39–61 (1983)
28. Robertson, N., Seymour, P.D.: Graph minors II. Algorithmic aspects of tree-width. Journal of Algorithms 7, 309–322 (1986)
29. Robertson, N., Seymour, P.D.: Graph minors X. Obstructions to tree decomposition. J. of Combinatorial Theory Series B 52, 153–190 (1991)
30. Sanchis, L.A.: Experimental analysis of heuristic algorithms for the dominating set problem. Algorithmica 33, 3–18 (2002)
31. Schaeffer, G.: Random sampling of large planar maps and convex polyhedra. In: Proc. of the 31st Annual ACM Symposium on the Theory of Computing (STOC 1999), pp. 760–769 (1999)
32. Seymour, P.D., Thomas, R.: Call routing and the ratcatcher. Combinatorica 14(2), 217–241 (1994)
33. Wan, P.J., Alzoubi, K.M., Frieder, O.: A simple heuristic for minimum connected dominating set in graphs. International Journal of Found. Comput. Sci. 14(2), 323–333 (2003)
34. West, D.B.: Introduction to Graph Theory. Prentice Hall Inc., Upper Saddle River (1996)