Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 326 (2004) 57–67
Theoretical Computer Science
www.elsevier.com/locate/tcs

On the complexity of fixed parameter clique and dominating set

Friedrich Eisenbrand[a], Fabrizio Grandoni[b,*]

[a] *Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, Saarbrücken, Germany*
[b] *Dipartimento di Informatica, Sistemi e Produzione, Università di Roma "Tor Vergata", Via del Politecnico 1, 00133 Roma, Italy*

Received 11 June 2003; received in revised form 15 April 2004; accepted 18 May 2004
Communicated by O. Watanabe

Abstract

We provide simple, faster algorithms for the detection of cliques and dominating sets of fixed order. Our algorithms are based on reductions to rectangular matrix multiplication. We also describe an improved algorithm for diamonds detection.

Keywords: Parameterized algorithms; Clique; Dominating set; Diamonds detection

1. Introduction

Clique and *dominating set* are classical problems of complexity theory. It is well known that both problems are NP-complete [10] and hard to approximate [11,14,17].

Parameterized complexity [7] is a discipline which tries to understand from which aspects of the input (the *parameter*) the hardness of problems originates. A parameterized problem is *fixed parameter tractable* if its time complexity is polynomial in the size i of the input, once that the size p of the parameter is fixed, and if the asymptotic running time in this case is independent from p. In other words, if the time complexity can be bounded by a function of the kind $f(p)i^c$, where $f(\cdot)$ is an arbitrary function and c is a constant (independent of p).

The *fixed parameter clique* and *dominating set* problems consist in determining whether an undirected graph G of n nodes contains a clique and a dominating set

* Corresponding author. Tel.: +39-496819325506; fax: +39-496819325599.
E-mail addresses: eisen@mpi-sb.mpg.de (F. Eisenbrand), grandoni@disp.uniroma2.it (F. Grandoni).

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2004.05.009

Table 1
Running time comparison for clique problem

ℓ	Previous best [15,16]	This paper
4	$O(n^{3.376})$, $O(e^{1.688})$	$O(n^{3.334})$, $O(e^{1.682})$
5	$O(n^{4.376})$, $O(e^{2.188})$	$O(n^{4.220})$, $O(e^{2.147})$
6	$O(n^{4.751})$, $O(e^{2.559})$	$O(e^{2.376})$
7	$O(n^{5.751})$, $O(e^{2.876})$	$O(n^{5.714})$, $O(e^{2.857})$

of ℓ nodes, respectively, where ℓ is the parameter. Currently, these problems are not known to be fixed parameter tractable. Indeed, this is also believed not to be the case. In a seminal work, Downey and Fellows [6,7] delivered completeness results for these problems. If one could show that fixed parameter clique and dominating set are fixed parameter tractable, then, by reduction, this would be the case for many other parameterized problems. More precisely, let *FPT* be the family of the fixed parameter tractable problems. Downey and Fellows showed that the fixed parameter clique and dominating set problems are complete for the complexity classes $W[1]$ and $W[2]$, respectively, where

$$FPT \subseteq W[1] \subseteq W[2].$$

It is conjectured that $FPT \neq W[1]$, which would imply that both problems considered are not fixed parameter tractable. Though this conjecture is far from being proved (it would imply $P \neq NP$), new evidences which support it have been constantly reported [5,9].

These results motivated us to study the complexity of these two prototypical problems and to look for faster algorithms to solve them.

1.1. Main results

We provide an improved algorithm for the fixed parameter clique problem. Let $O(n^{\omega(r,s,t)})$ denote the running time of the multiplication of an $n^r \times n^s$ matrix by an $n^s \times n^t$ matrix. Our algorithm runs in time $O(n^{\beta(\ell)}) = O(n^{\omega(\lfloor \ell/3 \rfloor, \lceil (\ell-1)/3 \rceil, \lceil \ell/3 \rceil)})$ on graphs of n nodes. If $\ell \geqslant 6$, our algorithm also runs in time $O(e^{\beta(\ell)/2})$ on graphs of e edges. This means an improvement over the fastest known methods [15,16] for dense and sparse graphs in the case that $\ell \equiv 1 \pmod 3$ and $4 \leqslant \ell \leqslant 16$ as well as that $\ell \equiv 2 \pmod 3$ and $\ell \geqslant 5$. In addition, for sparse graphs we obtain faster running times for $\ell \equiv 0 \pmod 3$ when $\ell \geqslant 6$. A comparison of the running times of the previous best algorithms and of our algorithm is depicted in Table 1 for $4 \leqslant \ell \leqslant 7$.

Nešetřil and Poljak [16] showed how to reduce the detection of an arbitrary (induced) subgraph of fixed order ℓ, to the detection of a clique of the same order in an auxiliary graph of ℓn nodes. For certain subgraphs, one can however do better. A *diamond* is obtained from a clique with four nodes by removing one edge. Kloks et al. [15] showed how to detect an induced diamond in time $O(n^\omega + e^{3/2})$. We improve on their result, by presenting a $O(e^{3/2})$ algorithm for this task.

Table 2
Running time comparison for dominating set problem

ℓ	Previous best	This paper
2	$O(n^3)$	$O(n^{2.376})$
3	$O(n^4)$	$O(n^{3.334})$
4	$O(n^5)$	$O(n^{4.220})$
5	$O(n^6)$	$O(n^{5.220})$
6	$O(n^7)$	$O(n^{6.063})$
7	$O(n^8)$	$O(n^{7.063})$

Finally, we provide an improved algorithm for the fixed parameter dominating set problem, of running time $O(n^{\omega(\lfloor \ell/2 \rfloor, 1, \lceil \ell/2 \rceil)})$. This answers a question posed by Regan [18], who asked whether there exists an algorithm which is faster than the $O(n^{\ell+1})$ trivial one. A comparison of our algorithm and the trivial one is depicted in Table 2 for $2 \leqslant \ell \leqslant 7$.

1.2. Related work

Itai and Rodeh [13] showed how to detect a *triangle* (clique of 3 nodes) in $O(n^\omega)$ time, where $\omega < 2.376$ is the exponent of fast square matrix multiplication [4]. Nešetřil and Poljak [16] generalized the algorithm of Itai and Rodeh to the detection of cliques of arbitrary (fixed) order ℓ. Their algorithm has time complexity $O(n^{\alpha(\ell)})$, where $\alpha(\ell) = \lfloor \ell/3 \rfloor \omega + \ell \pmod 3$.

Alon et al. [1] showed how to detect a triangle in $O(e^{2\omega/(\omega+1)})$ time. Kloks et al. [15] generalized the result of Alon et al. to cliques of arbitrary order. The running time of their algorithm is $O(e^{\alpha(\ell)\alpha(\ell-1)/(\alpha(\ell)+\alpha(\ell-1)-1)})$. If $\ell \pmod 3 \neq 0$, their running time is $O(e^{\alpha(\ell)/2})$, which is never inferior to the running time obtained by Nešetřil and Poljak for dense graphs. This does not hold when $\ell \pmod 3 = 0$. In that case the $O(n^{\alpha(\ell)})$ algorithm is faster if G is dense enough. Kloks et al. posed the question whether there exists a $O(e^{\alpha(\ell)/2})$ algorithm for the detection of cliques of arbitrary order $\ell \geqslant 3$. Our results provide a partially positive answer.

1.3. Notation

In this paper, we only deal with undirected graphs $G = (V, E)$, where V denotes the set of *vertices* (or *nodes*) and E denotes the set of *edges*. The *order* of G is the number of its nodes. The *neighborhood* $N(v)$ of a node v is the subset of nodes of G which are adjacent to v. The *degree* $d(v)$ of v is the number of its neighbors. A *clique* is a graph such that each pair of distinct nodes is adjacent. The cliques of 3 nodes are also called *triangles*. By K_ℓ we denote a clique of order ℓ.

A graph $G' = (V', E')$ is a *subgraph* of G if $V' \subseteq V$ and $E' \subseteq E$. A subgraph G' of G is an *induced subgraph* of G if two nodes in V' are adjacent in G' if and only if they are adjacent in G. If G' is an induced subgraph of G, we say that V' *induces* G' on G and we denote G' by $G[V']$. Two graphs are *isomorphic* if they admit an

isomorphism, that is a bijection between their vertex sets which preserves adjacency. If a graph F is isomorphic to an (induced) subgraph G' of G, G *contains* the (induced) subgraph F.

A subset V' of the nodes of G *dominates* a node $v \in V$ if v belongs to V' or v is adjacent to at least one node in V'. The set V' is a *dominating set* of G if all the nodes of G are dominated by V'.

To measure our running times, we always assume that a graph is represented via adjacency lists and that $e = \Omega(n)$.

2. Cliques

In this section, we present our algorithm for the fixed parameter clique problem. We distinguish between the detection of cliques in dense graphs (Section 2.1) and the detection of cliques in sparse graphs (Section 2.2).

2.1. Cliques in dense graphs

In this section, we present our algorithm for the fixed parameter clique problem in dense graphs.

We first recall the algorithm of Nešetřil and Poljak [16] for the same problem. If $\ell = 3h$ for some $h \in \mathbb{N}$, one creates an auxiliary graph $\widetilde{G}$ which has a node for each K_h in G and an edge between a pair of nodes if and only if the corresponding nodes in G form a K_{2h}. The graph G contains a K_{3h} if and only if $\widetilde{G}$ contains a triangle. As $\widetilde{G}$ has $O(n^h)$ nodes, a triangle in it can be detected in $O(n^{\omega h})$ time using fast square matrix multiplication [13]. If ℓ is not divisible by 3 one applies the following fact. A node v is contained in a K_ℓ if and only if the graph $G(v) = G[N(v)]$ induced on G by the neighborhood of v contains a $K_{\ell-1}$. This implies that one can detect a K_ℓ of G by applying an algorithm to detect a $K_{\ell-1}$ in each graph $G(v)$, $v \in V$. Thus one can detect a K_ℓ in $O(n^{\alpha(\ell)})$ time, where $\alpha(\ell) = \lfloor \ell/3 \rfloor \omega + \ell \pmod 3$.

The idea behind our algorithm is to allow for different orders of the sub-cliques, so that the case $\ell \pmod 3 \neq 0$ is not treated separately. Let ℓ_1, ℓ_2 and ℓ_3 be equal to $\lfloor \ell/3 \rfloor$, $\lceil (\ell-1)/3 \rceil$ and $\lceil \ell/3 \rceil$, respectively (notice that $\ell = \ell_1 + \ell_2 + \ell_3$). The graph G contains a K_ℓ if and only if a triangle is contained in the following 3-partite auxiliary graph $\widetilde{G}$. The nodes of $\widetilde{G}$ are partitioned into sets V_i for each $i \in \{1,2,3\}$, where the nodes in V_i are the cliques of order ℓ_i of G. A node $u \in V_i$ is adjacent to a node $v \in V_j$, $i \neq j$, if and only if the nodes of u and v induce a $K_{\ell_i+\ell_j}$ in G.

A triangle of $\widetilde{G}$ can be detected in the following way. For each pair of nodes $\{u,v\}$, $u \in V_1$ and $v \in V_3$, we compute the number $P(u,v)$ of 2-length paths between u and v through a node of V_2. The graph $\widetilde{G}$ contains a triangle if and only if there is a pair of adjacent nodes $\{u,v\}$, $u \in V_1$ and $v \in V_3$, such that $P(u,v) > 0$. The cost of the algorithm is bounded by the cost to compute the number of 2-length paths, that is the time required to multiply the $O(n^{\ell_1}) \times O(n^{\ell_2})$ adjacency matrix of the nodes in V_1 with the nodes in V_2 by the $O(n^{\ell_2}) \times O(n^{\ell_3})$ adjacency matrix of the nodes in V_2 with the nodes in V_3.

Theorem 1. *There is an algorithm which determines whether a graph G contains a clique of fixed order $\ell \geqslant 3$ in time* $O(n^{\beta(\ell)}) = O(n^{\omega(\lfloor \ell/3 \rfloor, \lceil (\ell-1)/3 \rceil, \lceil \ell/3 \rceil)})$.

If the rectangular matrix multiplication is carried out via the straightforward partition into square blocks and fast square matrix multiplication, one obtains the same time complexity of Nešetřil and Poljak:

$$\beta(\ell) \leqslant (\ell_3 - \ell_1) + (\ell_2 - \ell_1) + \omega(\ell_1, \ell_1, \ell_1) = \ell \pmod 3 + \ell_1 \omega = \alpha(\ell).$$

An asymptotically better bound can be obtained, when $\ell \pmod 3 \neq 0$, by using more sophisticated fast rectangular matrix multiplication algorithms [3,12].

Consider first the case $\ell = 3h+1$, where $h \in \mathbb{N}$. If $r \geqslant 1.171$, the bound on $\omega(1,1,r)$ given in [12] (which is not expressed via a closed formula) is superior to the trivial bound $\omega(1,1,r) \leqslant r - 1 + \omega$. This implies that $\beta(3h+1)$ is strictly less than $\alpha(3h+1)$ for any positive $h \leqslant 5$:

$$\beta(3h+1) = \omega(h,h,h+1) = h\omega\left(1,1,\frac{h+1}{h}\right) < h\left(\frac{h+1}{h} - 1 + \omega\right) = \alpha(3h+1).$$

Consider now the case $\ell = 3h+2$, where $h \in \mathbb{N}$. The best current bound for $\omega(r,1,1)$, $0 \leqslant r \leqslant 1$, is [3,12]

$$\omega(r,1,1) \leqslant \begin{cases} 2 + o(1) & \text{if } 0 \leqslant r \leqslant \delta = 0.294, \\ \omega + \dfrac{(1-r)(2-\omega)}{1-\delta} & \text{if } \delta < r \leqslant 1. \end{cases} \tag{1}$$

This implies that $\beta(3h+2)$ is strictly less than $\alpha(3h+2)$ for any positive h:

$$\begin{aligned} \beta(3h+2) = (h+1)\omega\left(\frac{h}{h+1},1,1\right) &\leqslant (h+1)\left(\omega + \frac{(1-\frac{h}{h+1})(2-\omega)}{1-\delta}\right) \\ &= \alpha(3h+2) - \frac{(\omega-2)\delta}{1-\delta}. \end{aligned}$$

Summing up, our algorithm is asymptotically faster than the algorithm of Nešetřil and Poljak if $l \pmod 3 = 1$ and $l \leqslant 16$, or $l \pmod 3 = 2$.

Nešetřil and Poljak [16] showed how to reduce the detection of an arbitrary (induced) subgraph of fixed order ℓ, to the detection of a clique of the same order in an auxiliary graph of ℓn nodes. From this reduction and Theorem 1:

Corollary 1. *There is an algorithm which determines whether a graph G contains a given (induced) subgraph of fixed order $\ell \geqslant 3$ in time* $O(n^{\beta(\ell)})$.

2.2. Cliques in sparse graphs

In this section we are concerned with the detection of cliques in a sparse graph G. In particular we want to develop efficient algorithms which depend on the number e of edges only.

Kloks et al. [15] described an algorithm for the detection of cliques in sparse graphs. The idea is to partition the vertex set into the set L of the nodes of degree smaller than Δ (*low-degree nodes*), and the set H of the remaining nodes (*high-degree nodes*), where Δ has to be fixed carefully. First, one looks for a K_ℓ which contains at least one low-degree node. Specifically, for every node $v \in L$, one looks for a $K_{\ell-1}$ in $G(v) = G[N(v)]$ by using an algorithm for the detection of cliques in dense graphs. Then one looks for a K_ℓ which contains high-degree nodes only.

By applying the algorithm of Section 2.1, a K_ℓ containing at least one low-degree node can be detected in $O(\sum_{v\in L} d(v)^{\beta(\ell-1)}) = O(e\Delta^{\beta(\ell-1)-1})$ steps. As $\sum_{v\in V} d(v) = 2e$, the number of high-degree nodes is bounded by $|H| \leqslant 2e/\Delta$. Then a clique formed by high-degree nodes only can be detected in $O((e/\Delta)^{\beta(\ell)})$ time. By setting $\Delta = e^{(\beta(\ell)-1)/(\beta(\ell)+\beta(\ell-1)-1)}$, the complexity of the procedure is $O(e^{\beta(\ell)\beta(\ell-1)/(\beta(\ell)+\beta(\ell-1)-1)})$. This bound already outperforms the previous best bound $O(e^{\alpha(\ell)\alpha(\ell-1)/(\alpha(\ell)+\alpha(\ell-1)-1)})$, which is obtained by using the algorithm of Nešetřil and Poljak.

However, the $O(n^{\beta(\ell)})$ running time obtained by the dense case algorithm is superior for some values of ℓ if the graph is sufficiently dense. This is because, for some values of ℓ, $\beta(\ell) < \beta(\ell-1)+1$ and thus $\beta(\ell)\beta(\ell-1)/(\beta(\ell)+\beta(\ell-1)-1) > \beta(\ell)/2$. The natural question arises, whether there exists an algorithm with a $O(e^{\beta(\ell)/2})$ running time for any $\ell \geqslant 3$. We now give a positive answer to this for the case $\ell \geqslant 6$. Erdős [8] proved the following lemma (see also [2]).

Lemma 1 (Erdős, 1962). *Let $e = \binom{s}{2} + t$ be the number of edges of a graph G, where $s, t \in \mathbb{N}$ and $s < t$. Then G contains at most $\binom{s}{\ell} + \binom{t}{\ell-1}$ cliques of order $\ell \geqslant 3$ and this upper bound is tight.*

Thus there are at most $O(e^{\ell/2})$ K_ℓ in G. It turns out that all such cliques can be enumerated within the same time bound.

Proposition 1. *There exists an algorithm which enumerates all the K_ℓ in G in $O(e^{\ell/2})$ steps, for any $\ell \geqslant 2$.*

Proof. Let L denote the set of nodes with degree smaller than a given Δ, and let H denote the set of the remaining nodes. We distinguish two kinds of cliques: the cliques which contain at least one node of L and the cliques formed by nodes in H only.

The cliques of the first kind can be enumerated by enumerating all the $K_{\ell-1}$ contained in the neighborhood of each node in L. This can be done in $O(\sum_{v\in L} d(v)^{\ell-1}) = O(e\Delta^{\ell-2})$ steps. The number of high-degree nodes is bounded by $|H| \leqslant 2e/\Delta$. This implies that the cliques of the second kind can be enumerated in $O((e/\Delta)^{\ell})$ steps. Setting $\Delta = \sqrt{e}$, one obtains the claimed time bound. □

Notice that we can label each edge of G with the number of K_ℓ to which it belongs to within the same time bound. Assume that the set of nodes is totally ordered. As one enumerates the K_ℓ, one can generate a list U of ordered ℓ-tuples $T = (t_1, \ldots, t_\ell)$, which represent the nodes of each K_ℓ. For each T one has to augment the label of

edge $\{t_i, t_j\}$, for each $1 \leqslant i < j \leqslant \ell$, by 1. To do this in linear time, we consider all the possible choices of i and j, and we generate lists $U_{i,j}$ which consist of the pairs (t_i, t_j) for each $T \in U$. Next, we lexicographically sort each $U_{i,j}$ with radix sort in linear time. Then we scan each list and add 1 to the edge label corresponding to each scanned pair (t_i, t_j). Notice that this can be done in linear time in the number of edges and in the size of the lists. We thus have the following corollary.

Corollary 2. *For each $\ell \geqslant 2$, one can label each edge of G with the number of K_ℓ to which it belongs in time* $O(e^{\ell/2})$.

An algorithm to detect a K_ℓ, $\ell \geqslant 6$, in $O(e^{\beta(\ell)/2})$ time derives from the previous results and from the algorithm described in Section 2.1. As in the dense case, we build the 3-partite auxiliary graph $\widetilde{G}$ and we look for a triangle in it. Remember that the partition V_i, $i \in \{1,2,3\}$, is formed by the K_{ℓ_i} of G, with ℓ_1, ℓ_2 and ℓ_3 being $\lfloor \ell/3 \rfloor$, $\lceil (\ell-1)/3 \rceil$ and $\lceil \ell/3 \rceil$, respectively. From Proposition 1, the set V_i, $i \in \{1,2,3\}$, has cardinality $O(e^{\ell_i/2})$, and it can be created within the same time bound. The cost to detect a triangle in $\widetilde{G}$ is bounded by the time required to multiply the $O(e^{\ell_1/2}) \times O(e^{\ell_2/2})$ adjacency matrix of the nodes in V_1 with the nodes in V_2 by the $O(e^{\ell_2/2}) \times O(e^{\ell_3/2})$ adjacency matrix of the nodes in V_2 with the nodes in V_3. This multiplication costs $O(e^{\omega(\ell_1/2, \ell_2/2, \ell_3/2)}) = O(e^{\beta(\ell)/2})$.

Theorem 2. *There is an algorithm which determines whether a graph G contains a clique of fixed order $\ell \geqslant 6$ in time* $O(e^{\beta(\ell)/2})$.

Note that the complexity of our algorithm for the sparse case is never larger than the complexity of the dense case algorithm of Section 2.1. Moreover, since $\beta(\ell)/2 \leqslant \beta(\ell)\beta(\ell-1)/(\beta(\ell) + \beta(\ell-1) - 1)$, this algorithm is faster than the algorithm of Kloks et al. when $\ell \geqslant 6$. By now, this is not the case for $3 \leqslant \ell \leqslant 5$. Here, the fastest detection algorithm for sparse graphs is the one of Kloks et al. Whether there exists a $O(e^{\beta(\ell)/2})$ algorithm for $3 \leqslant \ell \leqslant 5$ is an interesting open problem.

3. Dominating sets

The fastest known algorithm for the fixed parameter dominating set problem is the $O(n^{\ell+1})$ trivial algorithm which enumerates all the subsets of ℓ nodes of G and tests whether one of these subsets forms a dominating set. We recall that a dominating set is a subset V' of nodes such that every node not in V' is adjacent to at least one node in V' (Fig. 1). Here we give a faster algorithm based on fast matrix multiplication.

Let V_h be the set of subsets of V of cardinality h, $h \in \mathbb{N}$. Let moreover D_h be a 0-1 matrix whose rows are indexed by the elements of V_h and whose columns are indexed by the elements of V. Given $w \in V_h$ and $v \in V$, $D_h[w,v] = 0$ if and only if w dominates v. Let ℓ_1 and ℓ_2 be equal to $\lfloor \ell/2 \rfloor$ and $\lceil \ell/2 \rceil$, respectively (notice that $\ell = \ell_1 + \ell_2$). It is not hard to show that the matrix $D' = D_{\ell_1} \cdot D_{\ell_2}^{\mathrm{T}}$ contains a zero entry if and only if G admits a dominating set of size ℓ. More precisely, given $w \in V_{\ell_1}$ and $z \in V_{\ell_2}$, $D'[w,z]$ is the number of elements of V which are not dominated by $w \cup z$. The matrix D' can be computed in $O(n^{\omega(\ell_1, 1, \ell_2)})$ time.

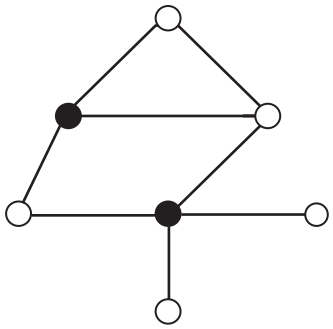

Fig. 1. The black nodes form a dominating set, that is the nodes of G are either adjacent to the black nodes or belong to the black nodes.

Theorem 3. *There is an algorithm which determines whether a graph G contains a dominating set of fixed cardinality $\ell \geqslant 2$ in time* $O(n^{\omega(\lfloor \ell/2 \rfloor, 1, \lceil \ell/2 \rceil)})$.

The algorithm above improves on the trivial algorithm for every value of $\ell \geqslant 2$ even if D' is computed via the straightforward decomposition and fast square matrix multiplication:

$$\omega(\lfloor \ell/2 \rfloor, 1, \lceil \ell/2 \rceil) \leqslant \lfloor \ell/2 \rfloor - 1 + \lceil \ell/2 \rceil - 1 + \omega = \ell + \omega - 2 < \ell + 1.$$

A better time bound is obtained by using more sophisticated rectangular matrix multiplication algorithms [12]. In Table 2, the complexity of our algorithm is compared with the trivial algorithm complexity in the case $2 \leqslant \ell \leqslant 7$. Interestingly, the complexity of our algorithm is $O(n^{\ell + o(1)})$ for any fixed $\ell \geqslant 8$. In fact, from Eq. (1), we have

$$\begin{aligned} \omega(\lfloor \ell/2 \rfloor, 1, \lceil \ell/2 \rceil) &\leqslant \lceil \ell/2 \rceil - \lfloor \ell/2 \rfloor + \omega(\lfloor \ell/2 \rfloor, 1, \lfloor \ell/2 \rfloor) \\ &\leqslant \lceil \ell/2 \rceil - \lfloor \ell/2 \rfloor + \lfloor \ell/2 \rfloor (2 + o(1)) \\ &= \ell + o(1). \end{aligned}$$

4. Diamonds

In this section, we consider the detection of induced diamonds. A *diamond* is a graph with four nodes, which results from a K_4 via the deletion of one edge (Fig. 2). By Corollary 1, an induced diamond can be detected in $O(n^{\beta(4)}) = O(n^{3.334})$ time. Interestingly, there is a faster algorithm for this problem. Kloks et al. [15] showed that a diamond can be detected in $O(n^{\omega} + e^{3/2})$ steps. Here, we present an algorithm for diamond detection which runs in $O(e^{3/2})$ steps. Our algorithm does not make use of fast matrix multiplication.

We again use the technique to decompose the set of nodes into high- and low-degree nodes H and L. The value of Δ will be determined in the sequel. A diamond contains

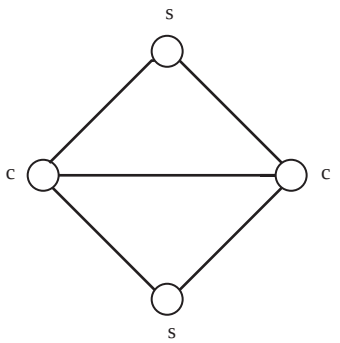

Fig. 2. A diamond. Side and central nodes are labeled with s and c, respectively.

two nodes of degree three and two of degree two. Let us call the nodes of the first kind *central nodes*, and the other two nodes *side nodes*. First, we look for a diamond which contains a low-degree central node. For every low-degree node v, we create the adjacency matrix of $G(v) = G[N(v)]$, we compute the connected components in $G(v)$ and we check if they are all cliques. If not, v is contained in a diamond and we can detect it in $O(d(v)^2)$ time. The complexity of this step is bounded by the time required to create the adjacency matrices. We can do that in $O(e + \sum_{v \in L} d(v)^2) = O(e\Delta)$ steps in the following way. We create an n-elements vector R that we initialize to 0. Then the algorithm proceeds through n rounds. In the vth round we fill in all the rows of the adjacency matrices which correspond to the node v. First, we set to 1 all the entries of R corresponding to the neighbors of v. Now, the vector R is equal to the vth row of the adjacency matrix of G (which is not available). Then for each neighbor u of v, we detect the row corresponding to v in the adjacency matrix of $G(u)$, and we fill in that row in linear time by using the vector R. At the end of each round we reset the non-zero entries of R. This procedure has a linear cost in the number of edges and in the size of the adjacency matrices created.

If no diamond is detected in the first step, we look for a diamond which contains a low degree side node. It follows from Corollary 2 that we can label each edge with the number of triangles of G in which it is contained in $O(e^{1.5})$ steps. Consider a low degree node v. The graph $G(v)$ is a disjoint union of cliques. The node v belongs to a diamond if and only if there exists an edge in a clique K_h of $G(v)$ which belongs to at least h triangles. This second step costs $O(e^{1.5} + \sum_{v \in L} d(v)^2) = O(e^{1.5} + e\Delta)$ time.

The diamonds not yet considered are formed by high-degree nodes only. The size of H is bounded by $|H| \leqslant 2e/\Delta$. Using the same approach of the first step, we can detect a diamond of this kind in $O(e + \sum_{v \in H} d_H(v)^2) = O(e + e^2/\Delta)$ steps, where $d_H(v)$ is the number of high degree neighbors of v.

The above described procedure runs in time $O(e^{1.5} + e\Delta + e^2/\Delta)$. This complexity is minimized by setting $\Delta = \sqrt{e}$.

Theorem 4. *There is an algorithm which determines whether a graph G contains an induced diamond in time* $O(e^{1.5})$.

For the remainder of this section we consider the problem to count the number d of induced diamonds of Kloks et al. [15] described an algorithm to count the number k of K_4 in G. Considering the results of Section 2.1, the running time of their algorithm is $O(e^{\beta(4)\beta(3)/(\beta(4)+\beta(3)-1)})$. They moreover noticed that:

$$t = \sum_{\{u,w\}\in E} \binom{A^2[u,w]}{2} = 6k + d,$$

where A is the 0-1 adjacency matrix of G ($A^2[u,w]$ is equal to the number of 2-length paths between u and w). Thus, the value of d can be determined in

$$O(n^\omega + e^{\beta(4)\beta(3)/(\beta(4)+\beta(3)-1)})$$

steps, where $O(n^\omega)$ is the time required to compute A^2.

A better bound can be obtained by using Corollary 2. We can label in $O(e^{1.5})$ time each edge $\{u,w\}\in E$ with the number $T(u,w)$ of triangles in which that edge is contained. As $T(u,v)$ is equal to $A^2[u,w]$ for any $\{u,w\} \in E$, we can compute t within the same time bound. Then the value of d can be computed in $O(e^{\beta(4)\beta(3)/(\beta(4)+\beta(3)-1)}) = O(e^{1.682})$ steps.

Proposition 2. *There is an algorithm which counts the number of induced diamonds contained in a graph G in $O(e^{\beta(4)\beta(3)/(\beta(4)+\beta(3)-1)})$ steps.*

5. Concluding remarks

The clique and dominating set problems are two of the best-studied problems in complexity theory. Their parameterized versions are conjectured not to be fixed parameter tractable. For the fixed parameter clique problem there are two non-trivial algorithms: one for the sparse and one for the dense case. We improve upon both of these algorithms. We provide a faster algorithm for the fixed parameter dominating set problem, for which the best algorithm known is the trivial one. We moreover present a faster algorithm for the detection of diamonds which does not make use of fast matrix multiplication techniques.

References

[1] N. Alon, R. Yuster, U. Zwick, Finding and counting given length cycles, Algorithmica 17 (3) (1997) 209–223.

[2] B. Bollobás, Extremal Graph Theory, in: London Mathematical Society Monographs, Vol. 11, Academic Press [Harcourt Brace Jovanovich Publishers], London, 1978.

[3] D. Coppersmith, Rectangular matrix multiplication revisited, J. Complexity 13 (1997) 42–49.

[4] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, J. Symbolic Comput. 9 (3) (1990) 251–280.

[5] R. Downey, Parameterized complexity for the skeptic, in: IEEE Annu. Conf. on Computational Complexity, 2003, pp. 147–170.

[6] R.G. Downey, M.R. Fellows, Fixed-parameter tractability and completeness. II. On completeness for $W[1]$, Theoret. Comput. Sci. 141 (1–2) (1995) 109–131.
[7] R.G. Downey, M.R. Fellows, Parameterized Complexity, Monographs in Computer Science, Springer, Berlin, 1999.
[8] P. Erdős, On the number of complete subgraphs contained in certain graphs, Magyar Tudomanyos Akad. Mat. Kutató Intezetenek Közl. 7 (1962) 459–464.
[9] M.R. Fellows, New directions and new challenges in algorithm design and complexity, parameterized, in: Workshop on Algorithms and Data Structures, 2003, pp. 505–519.
[10] M.R. Garey, D.S. Johnson, Computers and Intractability. A Guide to the Theory of NP-Completeness, Freemann, New York, 1979.
[11] J. Håstad, Clique is hard to approximate within $n^{1-\varepsilon}$, Acta Math. 182 (1) (1998) 105–142.
[12] X. Huang, V. Pan, Fast rectangular matrix multiplication and applications, J. Complexity 14 (2) (1998) 257–299.
[13] A. Itai, M. Rodeh, Finding a minimum circuit in a graph, SIAM J. Comput. 7 (4) (1978) 413–423.
[14] S. Khanna, R. Motwani, M. Sudan, U. Vazirani, On syntactic versus computational views of approximability, SIAM J. Comput. 28 (1) (1999) 164–191.
[15] T. Kloks, D. Kratsch, H. Müller, Finding and counting small induced subgraphs efficiently, Inform. Process. Lett. 74 (3–4) (2000) 115–121.
[16] J. Nešetřil, S. Poljak, On the complexity of the subgraph problem, Commentationes Mathematicae Universitatis Carolinae 26 (2) (1985) 415–419.
[17] R. Raz, S. Safra, A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP, in: ACM Symp. on the Theory of Computing, 1997, pp. 475–484.
[18] K.W. Regan, Finitary substructure languages, in: Structure in Complexity Theory Conference, 1989, pp. 87–96.