Efficient Algorithms for the MAX k-VERTEX COVER Problem*

Federico Della Croce[1] and Vangelis Th. Paschos[2,3]

[1] D.A.I., Politecnico di Torino, Italy
federico.dellacroce@polito.it
[2] PSL Research University, Université Paris-Dauphine, LAMSADE, CNRS, UMR 7243, France
paschos@lamsade.dauphine.fr
[3] Institut Universitaire de France

Abstract. We first devise moderately exponential exact algorithms for MAX k-VERTEX COVER, with time-complexity exponential in n but with polynomial space-complexity by developing a branch and reduce method based upon the measure-and-conquer technique. We then prove that, there exists an exact algorithm for MAX k-VERTEX COVER with complexity bounded above by the maximum among c^k and γ^τ, for some $\gamma < 2$, where τ is the cardinality of a minimum vertex cover of G (note that MAX k-VERTEX COVER $\notin$ **FPT** with respect to parameter k unless **FPT** = **W[1]**), using polynomial space. We finally study approximation of MAX k-VERTEX COVER by moderately exponential algorithms. The general goal of the issue of moderately exponential approximation is to catch-up on polynomial inapproximability, by providing algorithms achieving, with worst-case running times importantly smaller than those needed for exact computation, approximation ratios unachievable in polynomial time.

1 Introduction

In the MAX k-VERTEX COVER problem a graph $G(V, E)$ with $|V| = n$ vertices $1, \ldots, n$ and $|E|$ edges (i, j) is given together with an integer value $k < n$. The goal is to find a subset $K \subset V$ with cardinality k, that is $|K| = k$, such that the total number of edges covered by K is maximized. In its decision version, MAX k-VERTEX COVER can be defined as follows: "given G, k and ℓ, does G contain k vertices that cover at least ℓ edges?". MAX k-VERTEX COVER is **NP**-hard (it contains the minimum vertex cover problem as particular case), but it is polynomially approximable within approximation ratio $3/4$, while it cannot be solved by a polynomial time approximation schema unless **P** = **NP**. The interested reader can be referred to [19,30] for more information about approximation issues for this problem.

* Research supported by the French Agency for Research under the program TODO, ANR-09-EMER-010 and by a Lagrange fellowship of the Fondazione CRT, Torino, Italy.

J.C.M. Baeten, T. Ball, and F.S. de Boer (Eds.): TCS 2012, LNCS 7604, pp. 295–309, 2012.
© IFIP International Federation for Information Processing 2012

In the literature, we often find this problem under the name PARTIAL VERTEX COVER problem. It is mainly studied from a parameterized complexity point of view (see [17] for information on fixed-parameter (in)tractability). A problem is fixed-parameter tractable with respect to a parameter t, if it can be solved (to optimality) with time-complexity $O(f(t)p(n))$ where f is a function that depends on the parameter t, and p is a polynomial on the size n of the instance. In what follows, when dealing with fixed parameter tractability of MAX k-VERTEX COVER, we shall use notation MAX k-VERTEX COVER(t) to denote that we speak about fixed parameter tractability with respect to parameter t. Parameterized complexity issues for MAX k-VERTEX COVER are first studied in [3] where it is proved that PARTIAL VERTEX COVER is fixed-parameter tractable with respect to parameter ℓ, next in [28] where it is proved that it is **W[1]**-hard with respect to parameter k (another proof of the same result can be found in [9]) and finally in [31] where the fixed-parameter tractability results of [3] are further improved. Let us also quote the paper by [24], where it is proved that in apex-minor-free graphs graphs, PARTIAL VERTEX COVER can be solved with complexity that is subexponential in k.

The seminal Courcelle's Theorem [13] (see also [21,20] as well as [37] for a comprehensive study around this theorem) assures that decision problems defined on graphs that are expressible in terms of monadic second-order logic formulæ are fixed parameter tractable when the treewidth[1] of the the input-graph G, denoted by w, is used as parameter. Courcelle's Theorem can be also extended to a broad class of optimization problems [1]. As MAX k-VERTEX COVER belongs to this class, it is fixed parameter tractable with respect to w. In most of cases, "rough" application of this theorem, involves very large functions $f(w)$ (see definition of fixed-parameter tractability given above).

In [34], it is proved that *given a nice tree decomposition, there exists a fixed-parameter algorithm (based upon dynamic programming), with respect to parameter w that solves* MAX k-VERTEX COVER *in time $O(2^w k(w^2 + k) \cdot |I|)$, where $|I|$ is the number of nodes of the nice tree decomposition and in exponential space.* In other words, MAX k-VERTEX COVER$(w) \in$ **FPT**, but the fixed-parameter algorithm of [34] uses exponential space. Let us note that in any graph G, denoting by τ the size of a minimum vertex cover of G, it holds that $w \leqslant \tau$. So, MAX k-VERTEX COVER$(\tau) \in$ **FPT** too, but through the use of exponential space (recall that, as adopted above, MAX k-VERTEX COVER(τ) denotes the MAX k-VERTEX COVER problem parameterized by the size τ of a minimum vertex cover.

Very frequently, a serious problem about fixed-parameter tractability with respect to w is that it takes too much time to compute the "nice tree decomposition" that also derives the value of w. More precisely, this takes time $O^*(1.7549^n)$

[1] A tree decomposition of a graph $G(V, E)$ is a pair (X, T) where T is a tree on vertex set $V(T)$ the vertices of which we call nodes and $X = (\{X_i : i \in V(T)\})$ is a collection of subsets of V such that: (i) $\cup_{i \in V(T)} X_i = V$, (ii) for each edge $(v, w) \in E$, there exist an $i \in V(T)$ such that $\{v, w\} \in X_i$, and (iii) for each $v \in V$, the set of nodes $\{i : v \in Xi\}$ forms a subtree of T. The width of a tree decomposition $(\{X_i : i \in V(T)\}, T)$ equals $\max_{i \in V(T)}\{|Xi| - 1\}$. The treewidth of a graph G is the minimum width over all tree decompositions of G.

(notation $O^*(\cdot)$ ignores polynomial factors) by making use of exponential space and time $O^*(2.6151^n)$ by making use of polynomial space [25]. Note that the problem of deciding if the treewidth of a graph is at most w is fixed-parameter tractable and takes time $O(2^{O(w^3)}n)$ [33].

Dealing with solution of MAX k-VERTEX COVER by exact algorithms with running times (exponential) functions of n, let us note that a trivial optimal algorithm for MAX k-VERTEX COVER takes time $O^*(\binom{n}{k}) = O^*(n^k)$, and polynomial space, producing all the subsets of V of size k. This turns to a worst-case $O^*(2^n)$ time (since $\binom{n}{k} \leqslant 2^n$ with equality for $k = \frac{n}{2}$). An improvement of this bound is presented in [9], where an exact algorithm with complexity $O^*(n^{\omega\lceil k/3\rceil+O(1)})$ was proposed based upon a generalization of the $O^*(n^{\omega t})$ algorithm of [35] for finding a $3t$-clique in a graph, where $\omega = 2.376$. This induces a complexity $O^*(n^{0.792k})$, but exponential space is needed. As far as we know, no exact algorithm with running time $O^*(\gamma^n)$, for some $\gamma < 2$, is known for MAX k-VERTEX COVER.

In this paper, we first devise an exact branch and reduce algorithm based upon the measure-and-conquer paradigm by [22] (Section 2) requiring running time $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$, where Δ denotes the maximum degree of G, and polynomial space. The algorithm is then tailored to graphs with maximum degree 3 inducing a running time $O^*(1.3339^n)$ (Section 4). In Section 3, we devise a fixed parameter algorithm, with respect to parameter τ where, as mentioned above, τ is the cardinality of a minimum vertex cover of G that works in time $O^*(2^\tau)$ and needs only polynomial space. By elaborating a bit more this result we then show that the time-complexity of this algorithm is indeed either $O^*(\gamma^\tau)$ for some $\gamma < 2$ or $O^*(c^k)$, for some $c > 2$. In other words, this algorithm either works in time better than 2^τ or it is fixed parameter with respect to the size k of the desired cover. Finally, we show that the technique used for proving that MAX k-VERTEX COVER(τ) $\in$ **FPT**, can be used to prove inclusion in the same class of many other well-known combinatorial problems. A corollary of the inclusion of MAX k-VERTEX COVER(τ) in **FPT**, is that MAX k-VERTEX COVER in bipartite graphs can be solved in time $O^*(2^{n/2}) \simeq O^*(1.414^n)$. Finally, in Section 5, we address the question of approximating MAX k-VERTEX COVER within ratios "prohibited" for polynomial time algorithms, by algorithms running with moderately exponential complexity. The general goal of this issue is to cope with polynomial inapproximability, by developing algorithms achieving, with worst-case running times significantly lower than those needed for exact computation, approximation ratios unachievable in polynomial time. This approach has already been considered for several other paradigmatic problems such as MINIMUM SET COVER [7,15], MIN COLORING [2,6], MAX INDEPENDENT SET and MIN VERTEX COVER [5], MIN BANDWIDTH [16,26], ... Similar issues arise in the field of FPT algorithms, where approximation notions have been introduced, for instance, in [10,18]. In this framework, we particularly quote [32] where it is proved that, although not in **FPT**, MAX k-VERTEX COVER(k) is approximable by an FPT (with respect to k) approximation schema, where function $f(k)$ (in the time-complexity of this schema) is quite large, i.e., around something like $O^*(k^{2k^2})$.

2 An $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$-Time Polynomial Space Algorithm in General Graphs

In what follows, we denote by α_j the total number of vertices adjacent to j that have been discarded in the previous levels of the search tree. We denote by d_j the degree of vertex j and by $N(j)$ the set of vertices adjacent to j, that is the neighborhood of j. Notice that, whenever a branch on a vertex j occurs, for each $l \in N(j)$, if j is selected then d_l is decreased by one unit as edge (j, l) is already covered by j. Alternatively, j is discarded: correspondingly d_l is not modified and α_l is increased by one unit. We propose in this section a branch and reduce approach based on the measure-and-conquer paradigm (see for instance [22]). Consider a classical binary branching scheme on some vertex j where j is either selected or discarded. Contrarily to the classical branch-and-reduce paradigm where for each level of the search tree we define as *fixed* those vertices that have already been selected or discarded, while we define as *free* the other vertices, when using measure-and-conquer, we do not count in the measure the fixed vertices, namely the vertices that have been either selected or discarded at an earlier stage of the search tree and we count with a weight w_h the free vertices h. The vertex j to be selected is the one with largest coefficient $c_j = d_j - \alpha_j$. Let $c_{\max}$ denote such a coefficient, hence $c_{\max} \leqslant \Delta$. Then, each free vertex h is assigned a weight $w_h = w_{[i]}$ with $i = c_i = d_h - \alpha_h$ and we impose $w_{[0]} \leqslant w_{[1]} \leqslant w_{[2]} \leqslant w_{[3]} \leqslant \ldots \leqslant w_{[c_{\max}]} = 1$ that is the weights of the vertices are strictly increasing in their c_j coefficients.

We so get recurrences on the time $T(p)$ required to solve instances of size p, where the size of an instance is the sum of the weights of its vertices. Since initially $p = n$, the overall running time is expressed as a function of n. This is valid since when $p = 0$, there are only vertices with weight $w_{[0]}$ in the graph and, in this case, the problem is immediately solved by selecting the $k - \gamma$ vertices with largest α_j (if $\gamma < k$ vertices have been selected so far). Correspondingly free vertices j with no adjacent free vertices receive weight $w_{[0]} = 0$.

We claim that MAX k-VERTEX COVER can be solved with running time $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$ by the following algorithm called `MAXKVC`:

> Select j such that c_j is maximum and branch according to the following exhaustive cases:
> 1. if $c_j \geqslant 3$, then branch on j and either select or discard j;
> 2. else, $c_j \leqslant 2$ and `MAXKVC` is polynomially solvable.

Theorem 1. *Algorithm `MAXKVC` solves* MAX k-VERTEX COVER *with running time $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$ using polynomial space.*

Proof. To prove the above statement, we first show that the branch in step 1 can be solved with complexity $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$ and then we show that step 2 is polynomially solvable. Consider step 1. We always branch on the vertex j with largest $c_j = c_{\max} \leqslant \Delta$ where $c_j \geqslant 3$ and either we select or discard j. If we select j, vertex j is fixed and $c_{\max}$ vertices (the neighbors of j) decrease their degree (and correspondingly their coefficient) by one unit. Similarly, if we discard j, vertex j is fixed and $c_{\max}$ vertices (the neighbors of j) decrease their coefficient as their degree remains unchanged but their α parameter is increased by one unit. Hence, the recurrence becomes:

$$T(p) \leqslant 2T\left(p - w_{[c_{\max}]} - \sum_{h \in N(j)} \left(w_{[c_h]} - w_{[c_h - 1]}\right)\right)$$

By constraining the weights to satisfy the inequality:

$$w_{[j]} - w_{[j-1]} \leqslant w_{[j-1]} - w_{[j-2]} \quad \forall j = 2, \ldots, c_{\max}$$

the previous recurrence becomes in the worst-case:

$$T(p) \leqslant 2T\left(p - w_{[c_{\max}]} - c_{\max}\left(w_{[c_{\max}]} - w_{[c_{\max}-1]}\right)\right)$$

As $c_{\max} \leqslant \Delta$, where the equality occurs when $\alpha_j = 0$, the above recurrence becomes, in the worst-case, $T(p) \leqslant 2T\left(p - w_{[\Delta]} - \Delta\left(w_{[\Delta]} - w_{[\Delta-1]}\right)\right)$.

Summarizing, to handle graphs with maximum degree Δ, we need to guarantee that the recurrences $T(p) \leqslant 2T(p - w_{[i]} - i(w_{[i]} - w_{[i-1]}))$, $\forall i \in 3, \ldots, \Delta$ (as $c_j \geqslant 3$), and the constraints:

$$w_{[i]} - w_{[i-1]} \leqslant w_{[i-1]} - w_{[i-2]} \ \ \forall i = 2, \ldots, \Delta$$
$$0 = w_{[0]} \leqslant w_{[1]} \leqslant w_{[2]} \leqslant w_{[3]} \leqslant \ldots \leqslant w_{[\Delta-1]} \leqslant w_{[\Delta]} = 1$$

are satisfied simultaneously. This corresponds to a non linear optimization problem of the form:

$$\min \ \alpha$$
$$\alpha^{(w_{[i]} + i(w_{[i]} - w_{[i-1]}))} \geqslant 2 \ \ \forall i = 3, \ldots, \Delta \tag{1}$$
$$w_{[i]} - w_{[i-1]} \leqslant w_{[i-1]} - w_{[i-2]} \ \ \forall i = 2, \ldots, \Delta \tag{2}$$
$$0 = w_{[0]} \leqslant w_{[1]} \leqslant w_{[2]} \leqslant w_{[3]} \leqslant \ldots \leqslant w_{[\Delta-1]} \leqslant w_{[\Delta]} = 1 \tag{3}$$

We so get performances 1.4142^n, for $\Delta = 3$, 1.5157^n, for $\Delta = 4$, 1.5866^n, for $\Delta = 5$, 1.6405^n, for $\Delta = 6$, 1.6817^n, for $\Delta = 7$, or 1.7143^n, for $\Delta = 8$.

Interestingly enough, for all these values of Δ, the complexity corresponds to $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$. Indeed, this is not accidental. By setting:

$$w_{[i]} = \frac{(i-1)(\Delta+1)}{(i+1)(\Delta-1)} \ \ \forall i = 2, \ldots, \Delta \tag{4}$$
$$w_{[1]} = \frac{1}{2} w_{[2]} \tag{5}$$
$$w_{[0]} = 0 \tag{6}$$

we can see that constraints (2) and (3) are satisfied. To see that inequalities (2) are satisfied, notice that:

$$w_{[3]} - w_{[2]} = w_{[2]} - w_{[1]} \ = \ \frac{1}{3} w_{[3]}$$
$$w_{[2]} - w_{[1]} = w_{[1]} - w_{[0]} \ = \ w_{[1]}$$

For the general recursion with $i \geqslant 4$, we have to show that $w_{[i]} - w_{[i-1]} \leqslant w_{[i-1]} - w_{[i-2]}$, i.e., that $w_{[i]} - 2w_{[i-1]} + w_{[i-2]} \leqslant 0$. This corresponds to:

$$\begin{aligned}
&\left(\frac{i-1}{i+1} - 2\frac{i-2}{i} + \frac{i-3}{i-1}\right)\left(\frac{\Delta+1}{\Delta-1}\right) \leqslant 0 \\
\Longrightarrow\ & \frac{i-1}{i+1} - 2\frac{i-2}{i} + \frac{i-3}{i-1} \leqslant 0 \\
\Longleftrightarrow\ & i(i-1)^2 - 2(i-2)\left(i^2-1\right) + i(i-3)i + 1 \leqslant 0 \\
\Longleftrightarrow\ & i^3 - 2i^2 + i - 2i^3 + 4i^2 + 2i - 4 + i^3 - 2i^2 - 3i = -4 \leqslant 0, \quad \forall i
\end{aligned}$$

Also, to see that inequalities (3) are satisfied, notice that equations (4) imply:

$$\begin{aligned}
w_{[\Delta]} &= 1 \\
w_{[i]} &> 0 \ \ \forall i = 2, \ldots, \Delta \\
w_{[i]} &> w_{[i-1]} \ \ \forall i = 3, \ldots, \Delta
\end{aligned}$$

while equations (5) and (6) imply $w_{[2]} > w_{[1]} > w_{[0]} = 0$.

Finally, notice that such values of $w_{[j]}s$ satisfy constraints (1) that now correspond to $\Delta - 2$ copies of the inequality $\alpha^{\frac{\Delta+1}{\Delta-1}} \geqslant 2$ where the minimum value of α is obviously given by $2^{\frac{\Delta-1}{\Delta+1}n}$. Consequently, the overall complexity of step 1 is $O^*(2^{\frac{\Delta-1}{\Delta+1}n})$.

We consider now step 2. For $c_j = c_{\max} \leqslant 2$, MAX k-VERTEX COVER can be seen as a maximum weighted k-vertex cover problem in an undirected graph G where each vertex j has a weight α_j and a degree $d_j = c_j$ and the maximum vertex degree is 2. But this problem has been shown to be solvable in $O(n)$ time by dynamic programming in [36]. ♦

3 MAX k-VERTEX COVER and Fixed-Parameter Tractability

Denote by $(a - \overline{b} - c)$, a branch of the search tree where vertices a and c are selected and vertex b is discarded. Consider the vertex j with maximum degree Δ and neighbors $l_1, \ldots, l_\Delta$. As j has maximum degree, we may assume that if there exists an optimal solution of the problem where all neighbors of j are discarded, then there exists at least one optimal solution where j is selected. Hence, a branching scheme (called *basic branching scheme*) on j of type:

$$\left[l_1, \left(\overline{l_1} - l_2\right), \ldots, \left(\overline{l_1} - \overline{l_2} - \ldots - \overline{l_{\Delta-1}} - l_\Delta\right), \left(\overline{l_1} - \overline{l_2} - \ldots - \overline{l_\Delta} - j\right)\right]$$

can be applied. Hence, the following easy but interesting result holds.

Proposition 1. *The* MAX k-VERTEX COVER *problem can be solved to optimality in* $O^*(\Delta^k)$.

Proof. Consider vertex j with maximum degree Δ and neighbors $l_1, \ldots, l_\Delta$ where the basic branching scheme of type $[l_1, (\overline{l_1} - l_2), (\overline{l_1} - \overline{l_2} - l_3), \ldots, (\overline{l_1} - \overline{l_2} - \ldots - \overline{l_{\Delta-1}} - l_\Delta), (\overline{l_1} - \overline{l_2} - \ldots - \overline{l_\Delta} - j)]$ can be applied. Then, the last two branches can be substituted by the branch $(\overline{l_1} - \overline{l_2} - \ldots - \overline{l_{\Delta-1}} - j)$ as, if all neighbors of j but one are not selected, any solution including the last neighbor l_Δ but not including j is not better than the solution that selects j.

Now, one can see that the basic branching scheme generates Δ nodes. On the other hand, we know that in each branch of the basic branching scheme at least one vertex is selected. As, at most k nodes can be selected, the overall complexity cannot be superior to $O^*(\Delta^k)$. ♦

Corollary 1. MAX k-VERTEX COVER(k) *in bounded degree graphs is in* **FPT**.

Note that Corollary 1 can also be proved without reference to Proposition 1. Indeed, in any graph of maximum degree Δ, denoting by ℓ the value of an optimal solution for MAX k-VERTEX COVER, $\ell \leqslant k\Delta$. Then, taking ito account that MAX k-VERTEX COVER(ℓ) $\in$ **FPT**, immediately derives Corollary 1.

Now, let $V' \subset V$ be a minimum vertex cover of G and let τ be the size of V' that is $\tau = |V'|$. Correspondingly, let $I = V \setminus V'$ be a maximum independent set of G and set $\alpha = |I|$. Notice that V' can be computed, for instance, in $O^*(1.2738^\tau)$ time by means of the fixed-parameter algorithm of [12], and using polynomial space. Let us note that we can assume $k \leqslant \tau$. Otherwise, the optimal value ℓ for MAX k-VERTEX COVER would be equal to $|E|$ and one could compute a minimum vertex cover V' in G and then one could arbitrarily add $k - \tau$ vertices without changing the value of the optimal solution.

Theorem 2. *The following two assertions hold for* MAX k-VERTEX COVER*:*

1. *there exists an $O^*(2^\tau)$-time algorithm that uses polynomial space;*
2. *there exists an algorithm running in time $O^*(\max\{\gamma^\tau, c^k\})$, for two constants $\gamma < 2$ and $c > 4$, and needing polynomial space.*

Proof. For proving item 1, fix some minimum vertex cover V' of G and consider some solution K for MAX k-VERTEX COVER, i.e., some set of k vertices of G. Any such set is distributed over V' and its associated independent set $I = V \setminus V'$. Fix now an optimal solution K^* of MAX k-VERTEX COVER and denote by S' the subset of V' that belongs to K^* (S' can be eventually the empty set) and by I' the part of K^* belonging to I. In other words, the following hold:

$$\begin{aligned} K^* &= S' \cup I' \\ S' &\subseteq V' \\ I' &\subseteq I \quad = \quad V \setminus V' \end{aligned}$$

Given S' (assume $|S'| = k'$), it can be completed into K^* in polynomial time. Indeed, for each vertex i belonging to I we need simply to compute (in linear time) the total number e_i of edges (i, j) for all $j \in V' \setminus S'$. Then, I' is obtained by selecting the $k - k'$ vertices of I with largest e_i value. So, the following algorithm can be used for MAX k-VERTEX COVER:

1. compute a minimum vertex cover V' (using the algorithm of [11]);
2. for every subset $S' \subseteq V'$ of cardinality at most k, take the $k - |S'|$ vertices of $V \setminus V'$ with the largest degrees to $V' \setminus S'$; denote by I' this latter set;
3. return the best among the sets $S' \cup I'$ so-computed (i.e., the set that covers the maximum of edges).

Step 1 takes time $O^*(1.2738^\tau)$, while step 2 has total running time $O^*(\sum_{i=1}^{k}\binom{\tau}{i})$ that is at most $O^*(2^\tau)$.

Note that, from item 1 of Theorem 2, it can be immediately derived that MAX k-VERTEX COVER can be solved to optimality in $O^*(2^{\frac{\Delta-1}{\Delta}n})$ time. Indeed if a graph G has maximum degree Δ, then for the maximum independent set we have $\alpha \geqslant \frac{n}{\Delta}$. Also, we can assume that G is not a clique on $\Delta+1$ vertices (note that MAX k-VERTEX COVER is polynomial in cliques). In this case, G can be colored with Δ colors [8]. In such a coloring the cardinality of the largest color is greater than $\frac{n}{\Delta}$ and, a fortiori, so is the cardinality of a maximum independent set (since each color is an independent set). Consequently, $\tau \leqslant \frac{\Delta-1}{\Delta}n$.

In what follows, we improve the analysis of item 1 and prove item 2 that claims, informally, the instances of MAX k-VERTEX COVER that are not fixed-parameter tractable (with respect to k) are those solved with running time better than $O^*(2^\tau)$.

For this observe that the running time of the algorithm in the proof of item 1 is $O^*(\sum_{i=1}^{k}\binom{\tau}{i})$. As mentioned above, k can be assumed to be smaller than, or equal to, τ. Consider some positive constant $\lambda < 1/2$. We distinguish the following two cases: $\tau > k \geqslant \lambda\tau$ and $k < \lambda\tau$.

If $\tau > k \geqslant \lambda\tau$, then $\tau \leqslant k/\lambda$. As $\lambda < 1/2$, $k/\lambda > 2k$ and, since $i \leqslant k$, we get using Stirling's formula:

$$\sum_{i=1}^{k}\binom{\tau}{i} \leqslant \sum_{i=1}^{k}\binom{k/\lambda}{i} \leqslant k\binom{k/\lambda}{k} \sim k\,\frac{\frac{k}{\lambda}^{\frac{k}{\lambda}}}{k^k\left(\frac{k}{\lambda}-k\right)^{\left(\frac{k}{\lambda}-k\right)}} = k\left(\frac{\frac{1}{\lambda}^{\frac{1}{\lambda}}}{\left(\frac{1}{\lambda}-1\right)^{\left(\frac{1}{\lambda}-1\right)}}\right)^{k} = O^*\left(c^k\right) \tag{7}$$

for some constant c that depends on λ and it is fixed if λ is so.

If $k < \lambda\tau$, then, by the hypothesis on λ, $2k < \tau$ and, since $i \leqslant k$, expression $\sum_{i=1}^{k}\binom{\tau}{i}$ is bounded above by $k\binom{\tau}{k}$. In all, using also Stirling's formula the following holds:

$$\sum_{i=1}^{k}\binom{\tau}{i} \leqslant k\binom{\tau}{k} \leqslant k\binom{\tau}{\lambda\tau} \sim k\frac{\tau^\tau}{(\lambda\tau)^{(\lambda\tau)}[(1-\lambda)\tau]^{(1-\lambda)\tau}} = k\left(\frac{1}{\lambda^\lambda(1-\lambda)^{(1-\lambda)}}\right)^{\tau} \overset{\lambda<1/2}{<} O^*\left(2^\tau\right) \tag{8}$$

In other words, if $k < \lambda\tau$, then MAX k-VERTEX COVER can be solved in time at most $O^*(\gamma^\tau)$, for some γ that depends on λ and is always smaller than 2 for $\lambda < 1/2$.

Table 1. The values of c and γ for some values of λ

λ	0.01	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.40	0.45	0.49
$\frac{\frac{1}{\lambda^{\frac{1}{\lambda}}}}{\left(\frac{1}{\lambda}-1\right)^{\left(\frac{1}{\lambda}-1\right)}}$	270.47	53.00	25.81	16.74	12.21	9.48	7.66	6.36	5.38	4.61	4.11
$\frac{1}{\lambda^{\lambda}(1-\lambda)^{1-\lambda}}$	1.06	1.22	1.38	1.53	1.65	1.75	1.84	1.91	1.96	1.99	1.9996

Expressions (7) and (8) derive the claim and conclude the proof. In Table 1 the values of c and γ are given for some values of λ. ♦

Let us note that the technique of item 1 of Theorem 2, that consists of determining a decomposition of the input graph into a minimum vertex cover and a maximum independent set and then of taking a subset S' of a minimum vertex cover V' of the input-graph and of completing it into an optimal solution can be applied to several other well-known combinatorial **NP**-hard problems. We sketch here some examples:

– in MIN 3-DOMINATING SET (dominating set in graphs of maximum degree 3), the set S' is completed in the following way:
  - take all the vertices in $I \setminus \Gamma_I(S')$ (in order to dominate vertices in $V' \setminus S'$);
  - if there remain vertices of $V' \setminus S'$ not dominated yet solve a MIN SET COVER problem considering $\Gamma_I(S')$ as the set-system of the latter problem and assuming that a vertex $v \in \Gamma_I(S')$, seen as set, contains its neighbors in $V' \setminus S'$ as elements; since $\Gamma_I(S')$ is the neighborhood of S', the degrees of its vertices to $V' \setminus S'$ are bounded by 2, that induces a polynomial MIN SET COVER problem ([27]);

– in MIN INDEPENDENT DOMINATING SET, S' is completed by the set $I \setminus \Gamma_I(S')$, where $\Gamma_I(S')$ is the set of neighbors of S' that belong to I;

– in EXISTING DOMINATING CLIQUE, MIN DOMINATING CLIQUE (if any), MAX DOMINATING CLIQUE (if any) and MAX CLIQUE, S' can eventually be completed by a single vertex of $\Gamma_I(S')$.

Theorem 3. MIN INDEPENDENT DOMINATING SET, EXISTING DOMINATING CLIQUE, MIN DOMINATING CLIQUE, MAX DOMINATING CLIQUE, MAX CLIQUE *and* MIN 3-DOMINATING SET *can be solved in time $O^*(2^\tau)$ using polynomial space.*

4 Tailoring Measure-and-Conquer to Graphs with Maximum Degree 3

Let us note that, as it is proved in [23], for any $\epsilon > 0$, there exists an integer n_ϵ such that the pathwidth of every (sub)cubic graph of order $n > n_\epsilon$ is at most $(1/6 + \epsilon)n$. Based upon the fact that there exists for MAX k-VERTEX COVER(w) an $O^*(2^w)$-time exponential space algorithm [34], and taking into account that in (sub)cubic graphs $w \leqslant (1/6+\epsilon)n$, the following corollary is immediately derived.

Corollary 2. MAX k-VERTEX COVER *in graphs with maximum degree 3 can be solved in time* $O^*(2^{n/6}) = O^*(1.123^n)$ *using exponential space.*

In this section we tailor the measure-and-conquer approach developed in Section 2 to graphs with $\Delta = 3$, in order to get an improved running-time algorithm for this case needing polynomial space. The following remark holds.

Remark 1. The graph can be cubic just once. When branching on a vertex j of maximum degree 3, we can always assume that it is adjacent to at least one vertex h that has already been selected or discarded. That is, either $d_h \leqslant 2$, or $\alpha_h \geqslant 1$, that is $c_h \leqslant 2$. Indeed, the situation where the graph is 3-regular occurs at most once (even in case of disconnection). Thus, we make only one "bad" branching (where every free vertex of maximum degree 3 is adjacent only to free vertices of degree 3). Such a branching may increase the global running time only by a constant factor.

Lemma 1. *Any vertex* i *with* $d_i \leqslant 1$ *and* $\alpha_i = 0$ *can be discarded w.l.o.g.*

Proof. If $d_i = \alpha_i = 0$, then i can be obviously discarded. If $d_i = 1$ and $\alpha_i = 0$, then i is adjacent to another free vertex h. But then, if h is selected, i becomes of degree 0 and can be discarded. Alternatively, h is discarded, but then any solution with i but not h is dominated by that including h instead of i. ♦

Lemma 2. *Any vertex* i *with* $\alpha_i \geqslant 2$ *and* $d_i = 3$ *can be selected w.l.o.g.*

Proof. If $\alpha_i = 3$, then i can be obviously selected. If $d_i = 3$ and $\alpha_i = 2$, then i is adjacent to another free vertex h. But then, if h is discarded, we have $\alpha_i = 3$ and i can be selected. Alternatively, h is selected, but then any solution with h but not i is dominated by that including i instead of h. ♦

To solve MAX k-VERTEX COVER on graphs with $\Delta = 3$, consider the following algorithm, called `MAXKVC-3`.

> Select j such that c_j is maximum and branch according to the following exhaustive cases:
>
> 1. if $c_j = 3$, assume, w.l.o.g., that j is adjacent to i, l, m free vertices with $c_i \leqslant 2$ (see in [14]) and $c_i \leqslant c_l \leqslant c_m$, and branch on j according to the following exhaustive subcases:
>   (a) $c_i = c_l = c_m = 1$
>   (b) $c_i = c_l = 1$, $c_m = 2$
>   (c) $c_i = c_l = 1$, $c_m = 3$
>   (d) $c_i = 1$, $c_l = c_m = 2$ with l, m adjacent
>   (e) $c_i = 1$, $c_l = c_m = 2$ with l, m non adjacent
>   (f) $c_i = 1$, $c_l = 2$, $c_m = 3$
>   (g) $c_i = c_l = 2$, $c_m = 3$ with i, l adjacent
>   (h) $c_i = c_l = 2$, $c_m = 3$ with i, l non adjacent
>   (i) $c_i = 2$, $c_l = c_m = 3$
> 2. else $c_j \leqslant 2$ and `MAXKVC-3` is polynomially solvable.

The following Theorem 4 holds in graphs with maximum degree 3 (due to space constraints, the proof is omitted; it can be found in [14]).

Theorem 4. *Algorithm `MAXKVC-3` solves* MAX k-VERTEX COVER *on graphs with maximum degree 3 with running time $O^*(1.3339^n)$ and using polynomial space.*

5 Approximating MAX k-VERTEX COVER by Moderately Exponential Algorithms

We now show how one can get approximation ratios non-achievable in polynomial time using moderately exponential algorithms with worst-case running times better than those required for an exact computation (see [4,5] for more about this issue). Denote by $\text{opt}(G)$ the cardinality of an optimal solution for MAX k-VERTEX COVER in G and by $m(G)$, the cardinality of an approximate solution. Our goal is to study the approximation ratio $m(G)/\text{opt}(G)$.

In what follows, we denote, as previously, by K^* the optimal solution for MAX k-VERTEX COVER. Given a set K of vertices, we denote by $C(K)$, the set of edges covered by K (in other words, the value of a solution K for MAX k-VERTEX COVER is $|C(K)|$; also, according to our previous notation, $\text{opt}(G) = |C(K^*)|$). We first prove the following easy lemma that will be used later.

Lemma 3. *For any $\lambda \in [0,1]$, the subset H^* of λk vertices of K^* covering the largest amount of edges covered by K^*, covers at least $\lambda\,\text{opt}(G)$ edges.*

Proof. Indeed, if the λk "best" vertices of K^* cover less than $\lambda\,\text{opt}(G)$ edges, then any disjoint union of k/λ subsets of K^*, each of cardinality λk covers less than $\text{opt}(G)$ edges, a contradiction. ♦

Now, run the following algorithm, called `APPROX` in what follows:

1. fix some $\lambda \in [0,1]$ and optimally solve MAX λk-VERTEX COVER in G (as previously, let H^* be the optimal solution built and $C(H^*)$ be the edge-set covered by H^*);
2. remove H^* and $C(H^*)$ from G and approximately solve MAX $(1-\lambda)k$-VERTEX COVER in the surviving graph (by some approximation algorithm); let K' be the obtained solution;
3. output $K = H^* \cup K'$.

It is easy to see that if $T(p,k)$ is the running time of an optimal algorithm for MAX k-VERTEX COVER, where p is some parameter of the input-graph G (for instance, n, or τ), then the complexity of `APPROX` is $T(p,\lambda k)$. Furthermore, `APPROX` requires polynomial space.

Theorem 5. *If $T(p,k)$ is the running time of an optimal algorithm for* MAX k-VERTEX COVER*, then, for any $\epsilon > 0$,* MAX k-VERTEX COVER *can be approximated within ratio $1-\epsilon$ with worst-case running time $T(p,(1+2\sqrt{1-3\epsilon})k/3)$ and polynomial space.*

Proof. Denote by K^* an optimal solution of MAX k-VERTEX COVER in G, by G_2 the induced subgraph $G[V \setminus H^*]$ of G, by $\text{opt}_{(1-\lambda)}(G_2)$, the value of an optimal for MAX $(1-\lambda)k$-VERTEX COVER in G_2. Suppose that E' edges are common between $C(H^*)$ and $C(K^*)$. This means that $C(K^*) \setminus E'$ edges of $C(K^*)$ are in G_2 and are exclusively covered by the vertex-set $L^* = K^* \setminus H^*$ that belongs to G_2. Set $\ell^* = |L^*|$ and note that $\ell^* \leqslant k$ and $\ell^* \geqslant (1-\lambda)k$.

According to Lemma 3, the $(1-\lambda)k$ "best" vertices of L^* cover more than $(1-\lambda)|C(K^*) \setminus E'| = (1-\lambda)(\text{opt}(G) - |E'|)$ edges in G_2 and these vertices constitute a feasible solution for MAX $(1-\lambda)k$-VERTEX COVER in G_2. Hence:

$$\text{opt}_{(1-\lambda)}(G_2) \geqslant (1-\lambda)(\text{opt}(G) - |E'|) \tag{9}$$

Taking into account (9), the fact that K' in step 2 of `APPROX` has been computed by, say, a ρ-approximation algorithm and the fact that $|E'| \leqslant |C(H^*)|$, we get:

$$\begin{aligned} m(G) = C(H^*) + C(K') &\geqslant C(H^*) + \rho(1-\lambda)\text{opt}_{(1-\lambda)}(G_2) \\ &\geqslant C(H^*) + \rho(1-\lambda)(\text{opt}(G) - |E'|)\, C(H^*) \\ &\quad + \rho(1-\lambda)(\text{opt}(G) - C(H^*)) \\ &\geqslant (1-\rho(1-\lambda))C(H^*) + \rho(1-\lambda)\,\text{opt}(G) \end{aligned} \tag{10}$$

Using once more Lemma 3, $|C(H^*)| \geqslant \lambda\,\text{opt}(G)$, and combining it with (10), we get:

$$\frac{m(G)}{\text{opt}(G)} \geqslant \rho(1-\lambda) + \lambda(1-\rho(1-\lambda)) \tag{11}$$

Setting $\rho = \frac{3}{4}$ in (11), in order to achieve an approximation ratio $m(G)/\text{opt}(G) = 1-\epsilon$, for some $\epsilon > 0$, we have to choose an λ satisfying $\lambda = (1 + 2\sqrt{1-3\epsilon})/3$, that completes the proof of the theorem. ♦

Corollary 3. MAX k-VERTEX COVER *can be approximated within ratio* $1-\epsilon$ *and with running time:*

$$\min\left\{O^*\left(n^{\left(1+2\sqrt{1-3\epsilon}\right)(\omega k)/9}\right), O^*\left(\frac{\tau}{\left(1+2\sqrt{1-3\epsilon}\right)k/3}\right)\right\}$$

and polynomial space.

For Corollary 3, just observe that the running-times claimed for the first two entries are those needed to optimally solve MAX λk-VERTEX COVER (the former due to [9] and the latter due to item 1 of Theorem 2). Note that the second term in the min expression in the corollary is an FPT approximation schema (with respect to parameter τ). Observe also that for the cases where the time needed for solving MAX k-VERTEX COVER is given by the c^k expression of item 1 of Theorem 2, this represents an improvement with respect to the FPT approximation schema of [32]. Note finally that the result of Theorem 5 is indeed a kind of reduction between moderately exponential (or parameterized) approximation and exact (or parameterized) computation for MAX k-VERTEX COVER in the sense that exact solution on some subinstance of the problem derives an approximation for the whole instance.

Finally, let us close this section and the paper by some remarks on what kind of results can be expected in the area of (sub)exponential approximation. All the algorithms given in this section have exponential running time when we seek for a *constant* approximation ratio (unachievable in polynomial time). On the other hand, for several problems that are hard to approximate in polynomial time (like MAX INDEPENDENT SET, MIN COLORING, ...), subexponential time can be easily reached for ratios depending on the input-size (thus tending to ∞, for minimization problems, or to 0, for maximization problems). An interesting question is to determine, for these problems, if it is possible to devise a constant approximation algorithm working in subexponential time. An easy argument shows that this is not always the case. For instance, the existence of subexponential approximation algorithms (within ratio better than 4/3) is quite improbable for MIN COLORING since it would imply that 3-COLORING can be solved in subexponential time, contradicting so the "exponential time hypothesis" [29]. We conjecture that this is true for any constant ratio for MIN COLORING. Anyway, the possibility of devising subexponential approximation algorithms for **NP**-hard problems, achieving ratios forbidden in polynomial time or of showing impossibility of such algorithms is an interesting open question that deserves further investigation.

References

1. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. J. Algorithms 12(2), 308–340 (1991)
2. Björklund, A., Husfeldt, T., Koivisto, M.: Set partitioning via inclusion-exclusion. SIAM J. Comput. 39(2), 546–563 (2009)
3. Bläser, M.: Computing small partial coverings. Inform. Process. Lett. 85(6), 327–331 (2003)
4. Bourgeois, N., Escoffier, B., Paschos, V.T.: Efficient approximation by "low-complexity" exponential algorithms. Cahier du LAMSADE 271, LAMSADE, Université Paris-Dauphine (December 2007), `http://www.lamsade.dauphine.fr/cahiers/PDF/cahierLamsade271.pdf`
5. Bourgeois, N., Escoffier, B., Paschos, V.T.: Efficient Approximation of Combinatorial Problems by Moderately Exponential Algorithms. In: Dehne, F., Gavrilova, M., Sack, J.-R., Tóth, C.D. (eds.) WADS 2009. LNCS, vol. 5664, pp. 507–518. Springer, Heidelberg (2009)
6. Bourgeois, N., Escoffier, B., Paschos, V.T.: Efficient approximation of MIN COLORING by moderately exponential algorithms. Inform. Process. Lett. 109(16), 950–954 (2009)
7. Bourgeois, N., Escoffier, B., Paschos, V.T.: Efficient approximation of MIN SET COVER by moderately exponential algorithms. Theoret. Comput. Sci. 410(21-23), 2184–2195 (2009)
8. Brooks, R.L.: On coloring the nodes of a network. Math. Proc. Cambridge Philos. Soc. 37, 194–197 (1941)
9. Cai, L.: Parameter complexity of cardinality constrained optimization problems. The Computer Journal 51, 102–121 (2008)

10. Cai, L., Huang, X.: Fixed-Parameter Approximation: Conceptual Framework and Approximability Results. In: Bodlaender, H.L., Langston, M.A. (eds.) IWPEC 2006. LNCS, vol. 4169, pp. 96–108. Springer, Heidelberg (2006)
11. Chen, J., Kanj, I., Jia, W.: Vertex cover: further observations and further improvements. J. Algorithms 41, 280–301 (2001)
12. Chen, J., Kanj, I., Xia, G.: Improved upper bounds for vertex cover. Theoret. Comput. Sci. 411(40-42), 3736–3756 (2010)
13. Courcelle, B.: The monadic second-order logic of graphs. i. recognizable sets of finite graphs. Information and Computation 85, 12–75 (1990)
14. Croce, F.D., Paschos, V.T.: On the max k-vertex cover problem. Cahier du LAMSADE 307, LAMSADE, Université Paris-Dauphine (2011)
15. Cygan, M., Kowalik, L., Wykurz, M.: Exponential-time approximation of weighted set cover. Inform. Process. Lett. 109(16), 957–961 (2009)
16. Cygan, M., Pilipczuk, M.: Exact and approximate bandwidth. Theoret. Comput. Sci. 411(40–42), 3701–3713 (2010)
17. Downey, R.G., Fellows, M.R.: Parameterized complexity. Monographs in Computer Science. Springer, New York (1999)
18. Downey, R.G., Fellows, M.R., McCartin, C.: Parameterized Approximation Problems. In: Bodlaender, H.L., Langston, M.A. (eds.) IWPEC 2006. LNCS, vol. 4169, pp. 121–129. Springer, Heidelberg (2006)
19. Feige, U., Langberg, M.: Approximation algorithms for maximization problems arising in graph partitioning. J. Algorithms 41(2), 174–211 (2001)
20. Fellows, M.: Towards Fully Multivariate Algorithmics: Some New Results and Directions in Parameter Ecology. In: Fiala, J., Kratochvíl, J., Miller, M. (eds.) IWOCA 2009. LNCS, vol. 5874, pp. 2–10. Springer, Heidelberg (2009)
21. Fellows, M.R., Lokshtanov, D., Misra, N., Rosamond, F.A., Saurabh, S.: Graph Layout Problems Parameterized by Vertex Cover. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 294–305. Springer, Heidelberg (2008)
22. Fomin, F., Grandoni, F., Kratsch, D.: A measure & conquer approach for the analysis of exact algorithms. J. Assoc. Comput. Mach. 56(5), 1–32 (2009)
23. Fomin, F.V., Høie, K.: Pathwidth of cubic graphs and exact algorithms. Inform. Process. Lett. 97(5), 191–196 (2006)
24. Fomin, F.V., Lokshtanov, D., Raman, V., Saurabh, S.: Subexponential algorithms for partial cover problems. Inform. Process. Lett. 111(16), 814–818 (2011)
25. Fomin, F.V., Villanger, Y.: Treewidth Computation and Extremal Combinatorics. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 210–221. Springer, Heidelberg (2008)
26. Fürer, M., Gaspers, S., Kasiviswanathan, S.P.: An Exponential Time 2-Approximation Algorithm for Bandwidth. In: Chen, J., Fomin, F.V. (eds.) IWPEC 2009. LNCS, vol. 5917, pp. 173–184. Springer, Heidelberg (2009)
27. Garey, M.R., Johnson, D.S.: Computers and intractability. W. H. Freeman, San Francisco (1979)
28. Guo, J., Niedermeier, R., Wernicke, S.: Parameterized Complexity of Generalized Vertex Cover Problems. In: Dehne, F., López-Ortiz, A., Sack, J.-R. (eds.) WADS 2005. LNCS, vol. 3608, pp. 36–48. Springer, Heidelberg (2005)
29. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? J. Comput. System Sci. 63(4), 512–530 (2001)
30. Jäger, G., Srivastav, A.: Improved approximation algorithms for maximum graph partitioning problems. J. Comb. Optim. 10(2), 133–167 (2005)

31. Kneis, J., Langer, A., Rossmanith, P.: Improved Upper Bounds for Partial Vertex Cover. In: Broersma, H., Erlebach, T., Friedetzky, T., Paulusma, D. (eds.) WG 2008. LNCS, vol. 5344, pp. 240–251. Springer, Heidelberg (2008)
32. Marx, D.: Parameterized complexity and approximation algorithms. The Computer Journal 51(1), 60–78 (2008)
33. Marx, D.: Fixed parameter algorithms. Open lectures for PhD students in computer science (January 2010)
34. Moser, H.: Exact algorithms for generalizations of vertex cover. Master's thesis, Fakultät für Mathematik und Informatik, Friedrich-Schiller-Universität Jena (2005)
35. Nešetřil, J., Poljak, S.: On the complexity of the subgraph problem. Comment. Math. Univ. Carolinae, 415–419 (1985)
36. Niedermeier, R., Rossmanith, P.: On efficient fixed-parameter algorithms for weighted vertex cover. J. Algorithms 47(2), 63–77 (2003)
37. Praveen, M.: Logic, Courcelle's theorem and application. IMPECS School on Parameterized and Exact Computation (December 2010)