An improved exact algorithm for maximum independent set in sparse graphs

Nicolas Bourgeois, Bruno Escoffier, Vangelis Th. Paschos

UNIVERSITE PARIS DAUPHINE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision

CNRS UMR 7024

CAHIER DU LAMSADE

272

Janvier 2008

An improved exact algorithm for maximum independent set in sparse graphs

Nicolas Bourgeois, Bruno Escoffier, Vangelis Th. Paschos

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

An improved exact algorithm for maximum independent set in sparse graphs

N. Bourgeois, B. Escoffier, V. Th. Paschos
LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine, France
{bourgeois,escoffier,paschos}@lamsade.dauphine.fr

January 1, 2008

Abstract

We present an $O^*(1.0977^n)$ tree-search based exact algorithm for MAX INDEPENDENT SET in graphs with maximum degree 3. It can be easily seen that this algorithm also works in graphs with average degree 3.

1 Introduction

Very active research has been recently conducted around the development of optimal algorithms for **NP**-hard problems with non-trivial worst-case complexity. In this paper we handle MAX INDEPENDENT SET3, that is the MAX INDEPENDENT SET problem in graphs with maximum degree 3.

Given a graph $G(V, E)$, MAX INDEPENDENT SET consist of finding a maximum-size subset $V' \subseteq V$ such that for any $(v_i, v_j) \in V' \times V'$, $(v_i, v_j) \notin E$. MAX INDEPENDENT SET is a paradigmatic problem in theoretical computer science and numerous studies carry either on its approximation or on its solution by exact algorithms with non-trivial worst-case complexity. The best such complexity is, to our knowledge, the $O^*(1.18^n)$ algorithm by [7].

One of the most studied versions of MAX INDEPENDENT SET is its restriction in graphs with maximum degree 3, denoted by MAX INDEPENDENT SET3 in what follows. Dealing with exact computation of MAX INDEPENDENT SET3, several algorithms have been devised successively improving worst case complexity of its solution. Let us quote here the $O^*(1.1259^n)$ algorithm by [1], the $O^*(1.1254)$ algorithm by [2], the $O^*(1.1225^n)$ algorithm by [4], the $O^*(1.1120)$ algorithm [5] and the $O^*(1.1034^n)$ algorithm by [6]. In this paper, based upon a refined branching with respect to [5], we devise an exact algorithm for MAX INDEPENDENT SET-3 with worst-case running time of $O^*(1.0977^n)$. As it hopefully will be understood from the analysis, our result remains valid also for graphs where the maximum degree is higher but the average degree is bounded by 3.

Let $T(\cdot)$ be a super-polynomial and $p(\cdot)$ be a polynomial, both on integers. In what follows, using notations in [8], for an integer n, we express running-time bounds of the form $p(n) \cdot T(n)$ as $O^*(T(n))$, the asterisk meaning that we ignore polynomial factors. We denote by $T(n)$ the worst-case time required to exactly solve the considered combinatorial optimization problem with n variables. We recall (see, for instance, [3]) that, if it is possible to bound above $T(n)$ by a recurrence expression of the type $T(n) \leq \sum T(n-r_i)+O(p(n))$, we have $\sum T(n-r_i)+O(p(n)) = O^*(\alpha(r_1, r_2, \ldots)^n)$ where $\alpha(r_1, r_2, \ldots)$ is the largest zero of the function $f(x) = 1 - \sum x^{-r_i}$.

Consider a graph $G(V, E)$. Denote by $\alpha(G)$ the size of an optimal solution for MAX INDEPENDENT SET on G. For convenience, if $H \subseteq V$, we will denote by $\alpha(H)$ the cardinality of an optimal solution on the subgraph of G induced by H. For any vertex v, we denote by $\Gamma(v)$ the

set of its neighbors and by $deg(v) = |\Gamma(v)|$ its degree. We denote by $\alpha(G|v)$ (resp., $\alpha(G|\bar{v})$) the size of the optimal solution if we include v (resp. if we do not include v).

2 Preprocessing

The MAX INDEPENDENT SET-3-instance tackled is parameterized by $d = m - n$, where $n = |V|$ and $m = |E|$; $T(d)$ will denote the maximum running time of our algorithm on a graph whose parameter is smaller than or equal to d.

Before running the algorithm, we perform a preprocessing of the graph, in order to first remove vertices of degree 1 and 2 as well as dominated vertices. Some of the properties this preprocessing is based upon are easy and already known, mainly from [5]. We keep them for legibility.

Lemma 1. *Assume that there exists $v \in V$ such that $deg(v) = 1$. Then, there exists a maximal independent set S^* such that $v \in S^*$*

Proof. Let w be the only neighbor of v. If v is not selected, then w must be selected (else the solution would not even be maximal for inclusion). Then:

$$\alpha(G|v) = \alpha(V - \{v, w\}) + 1 \geq \alpha(V - \{v, w\} - \Gamma(w)) + 1 = \alpha(G|\bar{v})$$

Lemma 2. *Let $v, w \in V$, such that $v \in \Gamma(w)$ and $\Gamma(w) - \{v\} \subseteq \Gamma(v) - \{w\}$. Then, there exists a maximal independent set S^* such that $v \notin S^*$.*

Proof. Since $V - \{v\} - \Gamma(v) \subseteq V - \{w\} - \Gamma(w)$, $\alpha(G|v) \leq \alpha(G|w)$. We say that v is dominated by w and we can remove it from our graph. If v both dominates and is dominated by w, we choose at random the only one we keep. ∎

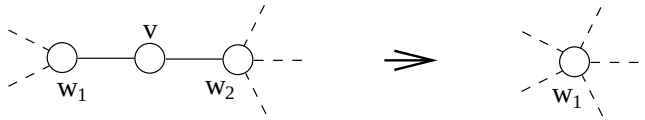

Figure 1: Vertex folding

Lemma 3. *Let $v, w_1, w_2 \in V$, such that $deg(v) = 2$, $\Gamma(v) = \{w_1, w_2\}$ and w_1 is not a neighbor of w_2. Then* MAX INDEPENDENT SET *on $G(V, E)$ is equivalent to the following problem:*

1. *form the subgraph $G'(V', E')$ induced by $V - \{v, w_2\}$. For any $x \neq v$ such that $\{x, w_2\} \in E$, add $\{x, w_1\}$ to E' (see Figure 1);*
2. *compute a solution, say S, for* MAX INDEPENDENT SET *on $G'(V', E')$;*
3. *if $w_1 \in S$, $S^* = S \cup \{w_2\}$. Else $S^* = S \cup \{v\}$.*

Proof. Notice at first that, according to lemma 1:

$$\begin{aligned} \alpha(G|\bar{v}) &= \alpha(G|w_1, w_2) = \alpha(G'|w_1) + 1 \\ \alpha(G|v) &= \alpha(G|\bar{w_1}, \bar{w_2}) = \alpha(G'|\bar{w_1}) + 1 \end{aligned}$$

From what it holds $\alpha(G) = \alpha(G') + 1$. ∎

This reduction is called *vertex folding* in [2].

Summing up the previous properties, we are now able to operate a reduction as soon as the graph has a vertex of degree 1 or 2.

Proposition 1. *Assume that there exists some vertex v such that $1 \leq deg(v) \leq 2$. Then, there exists a graph $G'(V', E')$ with $|V'| < |V|$ and $d' = |E'| - |V'| \leq d$ such that it is equivalent to compute* MAX INDEPENDENT SET *on G or on G'.*

Proof. The following holds:

- if $deg(v) = 1$, according to lemma 1 we may add it to the solution and remove its neighbor w from the graph. $d' = d - deg(w) + 2 \leq d$;
- if $deg(v) = 2$ and its neighbors w_1, w_2 are adjacent to each other, then v dominates them. According to lemma 2 we may remove w_1 from the graph; then lemma 1 allows us to add v to the solution and remove w_2. $d' = d - deg(w_1) - deg(w_2) + 4 \leq d$;
- finally, if $deg(v) = 2$ and its neighbors w_1, w_2 are not adjacent to each other, the equivalent graph we built in lemma 3 verifies $d' \leq d$. ∎

In other terms, we can always consider any vertex from our graph has at least degree 3.

We conclude this section by a remark that will be helpful later in the branching analysis.

Remark 1. Note that in the last case of Proposition 1 ($deg(v) = 2$ and its neighbors w_1, w_2 are not adjacent to each other) then (at least) one of the two following cases occurs when reducing the graph: either (i) $d' \leq d-1$ (if w_1 and w_2 are both adjacent to a third vertex $x \neq v$), or (ii) a vertex of degree at least 4 is created in G'. ∎

3 Branching

In this section, we consider that the whole preprocessing described in Section 2 has been computed as long as possible. That means no vertex has degree 2 or less, and no vertex is dominated.

Lemma 4. *Fix some vertex v. The number $\mathcal{N}$ of edges that are adjacent to at least one of its neighbors is bounded below by:*

$$deg(v) + \frac{1}{2} \sum_{w \in \Gamma(v)} deg(w)$$

Proof. Let I (resp. Ω) the inner (resp. outer) edges of $\Gamma(v)$, that means edges linking two vertices from $\Gamma(v)$ (resp. one vertex from $\Gamma(v)$ and one vertex from $V - \Gamma(v)$). Then:

$$\mathcal{N} = |\Omega| + |I|$$
$$\sum_{w \in \Gamma(v)} deg(w) = |\Omega| + 2|I|$$

From what we get:

$$\mathcal{N} = \sum_{w \in \Gamma(v)} deg(w)/2 + |\Omega|/2 \tag{1}$$

Notice that any $w \in \Gamma(v)$ has at least one neighbor in $V - \{v\} - \Gamma(v)$; else w would dominate v. Moreover, w is adjacent to v. Thus, $|\Omega| \geq 2deg(v)$. ∎

Proposition 2. *Assume that there exists some vertex v whose degree is at least 5. Then, $T(d) \leq T(d-4) + T(d-7)$.*

Proof. We branch on v. If we choose not to add v to the solution, then we can remove v and any vertex incident to him, that means $d' = d - deg(v) + 1 \leq d - 4$. On the other hand, if v belongs to optimal, we remove from our graph v and all its neighbors, that means (according to lemma 4):

$$\begin{aligned} n' &= n - 1 - deg(v) \\ m' &\leq m - deg(v) - \frac{1}{2} \sum_{w \in \Gamma(v)} deg(w) \\ d' &\leq d + 1 - \frac{1}{2} \sum_{w \in \Gamma(v)} deg(w) \end{aligned}$$

And $\sum_{w \in \Gamma(v)} deg(w) \geq 15$. Since d' has to be an integer, this leads to the expected result. ∎

Proposition 3. *Assume that there exists some vertex v whose degree is 4. Then, $T(d) \leq T(d-3) + T(d-5)$. Moreover, assume that one of the following cases holds:*

1. *one neighbor of v has degree 4;*
2. *any neighbor has degree 3 but the subgraph induced by $\Gamma(v)$ contains at most one edge.*

Then, $T(d) \leq T(d-3) + T(d-6)$.

Proof. We branch on v. If we choose not to add v to the solution, then we can remove v and any vertex incident to him, that means $d' = d - deg(v) + 1 \leq d - 3$. On the other hand, if v belongs to optimal, we remove from our graph v and all its neighbors:

$$d' \leq d + 1 - \frac{1}{2} \sum_{w \in \Gamma(v)} deg(w)$$

Dealing with the general case, $\sum_{w \in \Gamma(v)} deg(w) \geq 12$. If *1.* holds, $\sum_{w \in \Gamma(v)} deg(w) \geq 13$. On case *2.*, we just have to notice that $|I| \leq 1$ means $|\Omega| \geq 10$; replacing this inequality in (1) we get $d' \leq d - 6$. ∎

Proposition 4. *Assume now that the degree of any vertex in our graph is exactly 3. Assume also that G contains some 3-clique $\{a, b, c\}$. Then, $T(d) \leq 2T(d-4)$.*

Proof. Let v the third neighbor of a and u, w the two other neighbors of v. Note that u (and w) differs from b and c, otherwise a would dominate b or c. We branch on v. If v does not belong to the optimal, it is removed, and so are its incident edges. In the remaining graph, a is dominated by b and c (see Figure 2). According to lemma 2 we may add it to the optimal and remove the whole clique. Eventually,

$$d' = (m - 8) - (n - 4) = d - 4$$

On the other hand, if v has to be added to the solution, we may remove it and its neighbors from the graph. That means, according to lemma 4,

$$d' \leq (m - 3 - 9/2) - (n - 4) = d - 7/2$$

Since d' has to be an integer, that leads to the expected inequality. ∎

Proposition 5. *Assume now that G is a 3-clique-free graph where any vertex has degree exactly 3. Assume also that G contains the subgraph described in Figure 3 (two pentagons sharing two adjacent edges). Then, $T(d) \leq T(d-3) + T(d-5)$.*

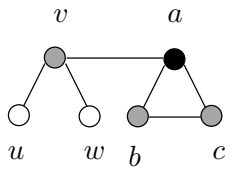

Figure 2: Clique $\{a, b, c\}$

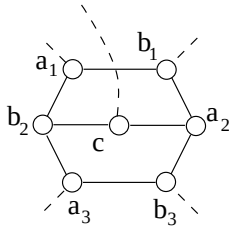

Figure 3: Pentagons sharing two adjacent edges

Proof. Notice at first that in a 3-clique-free graph where any vertex has degree exactly 3, each time we add a vertex to the solution and remove its neighbors, we decrease d by 5. Indeed, the set of neighbors of a vertex contains no inner edge, else it would form a clique. From what we get $\mathcal{N} = \sum_{w \in \Gamma(v)} deg(w) = 9$ and finally

$$d' = (m - 9) - (n - 4) = d - 5$$

We branch on c, and consider now the case it does not belong to our solution. Once c and its incident edges have been removed, $deg(a_2) = deg(b_2) = 2$. We reduce them in any order. In the remaining graph,

$$d' = (m - 8) - (n - 5) = d - 3 \ \blacksquare$$

Proposition 6. *Assume now that G is a 3-clique-free graph where any vertex has degree exactly 3. Assume also that G does not contain the subgraph described in Figure 3. Then, $T(d) \leq T(d-5) + 2T(d-8) + T(d-10)$.*

Proof. In this proposition, instead of branching on a single vertex as previously, we successively branch on several vertices. First of all, consider any $v \in V$. As we saw previously, since there is no 3-clique, if we add v to the solution, we decrease d by 5, else we decrease d by 2. Thus, we can write $T(d) \leq T(d-5) + Q(d-2)$, for some function $Q \leq T$. If we had no further information about the remaining graph, we would have no choice but to write $T(d) \leq T(d-5) + T(d-2)$. But our graph is not any graph; in particular some higher than 3 degree vertices may have been created during our first branching. From now on, the proof will focus on refining analysis of Q thanks to our graph properties.

Let w_1, w_2, w_3 the neighbors of v. Once v and its incident edges have been removed, they all have degree 2. Moreover, no couple of them can be adjacent, else they would form a clique with v. According to lemma 3, we now reduce our graph. We distinguish some different cases.

Consider at first an easy case, where there exists a couple of vertices u_1, u_2 both adjacent to, say, w_1 and w_2 (see Figure 4). We can easily see that taking u_1 and/or u_2 is never interesting, it is never worse to take w_1 and w_2. So, we can add $\{w_1, w_2\}$ to optimal and delete $\{u_1, u_2\}$. This operation decreases d by 2: 4 vertices and 6 edges are removed. Indeed, u_1 and u_2 cannot

be adjacent otherwise u_1 would dominate u_2 (and vice-versa) before branching on v. In other words, $Q(d) \leq T(d-2)$. Eventually,

$$T(d) \leq T(d-5) + T(d-4)$$

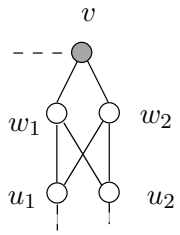

Figure 4: u_1, u_2 both adjacent to w_1 and w_2

Assume now that there exists one single vertex u that is adjacent to, say, w_1 and w_2. Let us denote s and t the third neighbor of respectively w_1 and w_2 (see Figure 5). Notice that in this case u cannot be adjacent to w_3, else it would dominate v. When operating reductions of w_1 and w_2, s, u and t are merged together in a single vertex. Two cases may occur (see Remark 1): either 2 of them have another common neighbor and d decreases by at least one, or our graph contains a vertex of degree 5. In the first case, we get $Q(d) \leq T(d-1)$ and finally $T(d) \leq T(d-5) + T(d-3)$. In the latter case, we now branch on the degree 5 vertex; according to proposition 2, $Q(d) \leq T(d-4) + T(d-7)$, that means:

$$T(d) \leq T(d-5) + T(d-6) + T(d-9)$$

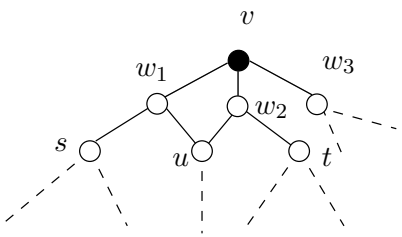

Figure 5: u adjacent to w_1 and w_2

Let us now focus on the main case, where $\Gamma(w_i)$'s are disjoint. Let us denote u_i^1 and u_i^2 the two other neighbors of w_i. When reducing vertex w_i, going back to Remark 1, either d decreases by at least one, or a vertex - say u_i - of degree 4 is created. If for at least one w_i d decreases, then $Q(d) \leq T(d-1)$ and eventually $T(d) \leq T(d-5) + T(d-3)$. Otherwise, when reductions of degree 2 vertices w_i's have been proceeded, the remaining graph contains exactly three vertices whose degree is 4, namely u_1, u_2, u_3. Since our graph is not fully reduced (branching may have disturbed it), it is possible that some neighbor of some u_i dominates it. In this case we can remove it and its incident edges without branching; $Q(d) \leq T(d-3)$, that means

$$T(d) \leq 2T(d-5)$$

Thus, we may now assume that no u_i is dominated. We have to consider wether they are adjacent or not.

If say u_1 and u_2 are adjacent, then we apply Proposition 3 (in the specific case the vertex we branch has at least one neighbor of degree 4): if we add u_1 to the solution, we decrease d by at least 6; on the other hand,if u_1 does not belong to optimal, we decrease it by 3. Hence, we could write $Q(d) \leq T(d-6) + T(d-3)$.

If no u_i is adjacent to any other u_j, this time if we try to use proposition 3 we cannot assert that two degree 4 vertices are adjacent, that would mean decreasing d only by 5. Fortunately, it is possible to prove that $\Gamma(u_1)$ does not have more than one inner edge. Indeed, assume that it has two inner edges; since, before branching on v, we assumed that the graph does not contain any 3-clique, then the unique possibility is described in Figure 6. It means that, before branching on v, our graph contained two pentagons sharing two edges (edges (w_1, u_1^1) and (w_1, u_1^2)). This is in contradiction with hypothesis of proposition 6. Hence, in this case also $Q(d) \leq T(d-6)+T(d-3)$.

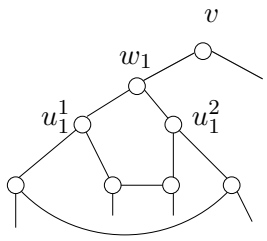

Figure 6: Two pentagons sharing two consecutive edges

At this step, we get $T(d) \leq T(d-5) + Q(d-2) \leq 2T(d-5) + T(d-8)$. This sums up to $T(d) = O^*(1.2076^d)$.

We make a final remark that further improves the running time. Indeed, we will see that after performing the branching on u_1, then there exists another vertex of degree 4 in the remaining graph (or an even better case occurs). By branching on it, we decrease d either by 3 or by 5 (Proposition 3). This leads to $Q(d) \leq T(d-6)+T(d-3-3)+T(d-3-5) = 2T(d-6)+T(d-8)$, and finally $T(d) \leq T(d-5) + 2T(d-8) + T(d-10)$, as claimed in the proposition. To see this, consider two cases. If say u_1 and u_2 are not adjacent, then when branching along the branch "take u_1", u_2 will still have degree at least 4 (or would have been deleted by domination which is even better).

The difficult case occurs when u_1, u_2 and u_3 form a clique. Let s and t the two neighbors of u_1 with degree 3 (see Figure 7).

- If s and t are adjacent, then in $\Gamma(u_1)$ there are only 2 edges (the other one being (u_2, u_3)), otherwise s or t would dominate u_1. Hence, when taking u_1, we delete 12 edges and 5 vertices, which means $Q(d) \leq T(d-3) + T(d-7)$, *i.e.* $T(d) \leq 2T(d-5) + T(d-9)$.
- If s and t are not adjacent, $Q(d) \leq T(d-3) + T(d-6)$ but when reducing s and t (after taking u_1), we are in one of the two cases of Remark 1, *i.e.* either d decreases and $Q(d) \leq T(d-3) + T(d-7)$, or a degree 4 vertex is created.

The analysis of the solutions of the equations induced by the previous analysis shows that the worst case running time corresponds to the case $T(d) \leq T(d-5) + 2T(d-8) + T(d-10)$, which sums up to $T(d) = O^*(1.2048^d)$ ∎

4 Dealing with trees

As noticed in [5], we have to be careful with our previous analysis. Indeed, the adopted measure $d = m - n$ creates a somehow unexpected problematic situation, occurring when several connecting components are created when branching, one or several of them being tree(s). Indeed,

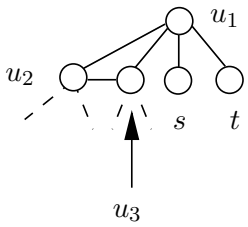

Figure 7: u_1, u_2 and u_3 is a clique

applying preprocessing rules immediately reduces a tree to a single vertex, which can be added to the solution. However, when removing this vertex (*i.e.* when dealing with the tree), the measure $m - n$ *increases*. In other words, when creating a tree, if the measure globally decreases by say x, it decreases only by $x - 1$ in the remaining graph, tree excepted.

To deal with this situation, we study the situations when one or several tree(s) are created when branching. We show that in these particular cases the number of edges deleted is sufficient to compensate the loss induced by the tree(s).

To obtain this, we use a property shown by Fürer [5] that handles the case where there is a separator of size one or two in the graph. More precisely, if there exists one vertex u or a couple of vertex (u_1, u_2) the deletion of which disconnects the graph, creating one connecting component of say constant size, then this connected component can be eliminated before branching. This reduction, not increasing d, can be added for instance to the preprocessing step.

Let us consider that we branch on a vertex v. As pointed out by Fürer [5], if we don't take v, then no tree can be created (since every vertex of the graph has degree at least 3).

Let us concentrate now on the case when taking v. Note that since each vertex in the graph has degree 3, when a tree with r vertices is created, at least $r + 2$ edges link one vertex of the tree to one vertex of $\Gamma(v)$. If v has degree 3 (the graph is three regular), if a tree is created, this tree is a single vertex w linked to the 3 vertices in $\Gamma(v)$. Indeed, each vertex in $\Gamma(v)$ has at least one edge linking it to the rest of the graph (else there would be a separator of size 2), hence at most one edge linking it to the tree. In this case, the graph reduces (before branching on v): we delete v and w and merge the vertices in $\Gamma(v)$ in a single vertex. We delete 6 edges and 4 vertices, hence d decreases.

Assume now that $deg(v) \geq 4$.

Let us note by:

- $|I|$ (as previously) the number of edges linking two vertices in $\Gamma(v)$;
- l the number of trees created when branching on v (taking v);
- t the number of edges linking a vertex in $\Gamma(v)$ to a vertex in one of the trees created;
- e the number of edges linking one vertex of $\Gamma(v)$ to the rest of the graph;
- $D = \sum deg(w)$ for $w \in \Gamma(v)$.

When branching on v, either we don't take v and $d' = d - deg(v) + 1 \leq d - 3$, or we take v. In this latter case: we delete $1 + deg(v)$ vertices v and $\Gamma(v)$, $deg(v) + |I| + t + e$ edges adjacent to vertices in $\Gamma(v)$, and we loose 1 for each tree created. Hence, after handling the tree(s), d has globally decreased by: $\delta = |I| + t + e - 1 - l$.

Note that $t \geq 3$ (or we have a separator of size 2 or less) and $e \geq 3l$ (at least 3 edges for each tree). Hence: $\delta \geq |I| + 3 + 3l - 1 - l \geq 2 + 2l$. In particular, if $l > 1$ then $\delta \geq 6$.

If $l = 1$, then $\delta = |I| + t + e - 2$. $D = 2|I| + t + e + 4$. Then $\delta = D/2 + t/2 + e/2 - 4$. Since $e \geq 3$ and $t \geq 3$, if $D \geq 13$ then $\delta \geq 13/2 - 1$. Since it is an integer, $\delta \geq 6$.

The only remaining case is $D = 12$ (hence $deg(v) = 4$ and each vertex in $\Gamma(v)$ has degree 3), $t = 3$ and $e = 3$ (see Figure 8). But in this case, the tree separated has only one vertex w. Then, the only inner edge in $\Gamma(v)$ is incident to the vertex s in $\Gamma(v)$ non adjacent to w (otherwise there would be a dominated vertex), as depicted in Figure 8. Hence, it is never interesting to take s in the solution (take either v or t instead), and we don't need to branch.

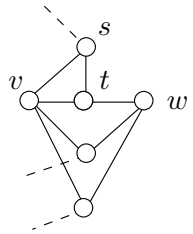

Figure 8: Taking s is never interesting

We decrease d either by 3 (not taking v) or by 6 (taking v). This is in particular the recurrence relation $T(d) \leq T(d-3)+T(d-6)$ needed when operating branching in the analysis of Section 3.

5 The concluding theorem

The analysis provided in Sections 2,3 and 4 allows to state the following result.

Theorem 1. *On any graph, it is possible to solve* MAX INDEPENDENT SET *with running time* $O^*(1.2048^{m-n})$. *In particular, a solution to* 3-MIS *may be computed with running time* $O^*(1.0977^n)$.

References

[1] R. Beigel. Finding maximum independent sets in sparse and general graphs. In *Proc. Symposium on Discrete Algorithms, SODA'99*, pages 856–857, 1999.

[2] J. Chen, I. A. Kanj, and G. Xia. Labeled search trees and amortized analysis: improved upper bounds for NP-hard problems. In T. Ibaraki, N. Katoh, and H. Ono, editors, *Proc.International Symposium on Algorithms and Computation, ISAAC'03*, volume 2906 of *Lecture Notes in Computer Science*, pages 148–157. Springer-Verlag, 2003.

[3] D. Eppstein. Improved algorithms for 3-coloring, 3-edge-coloring, and constraint satisfaction. In *Proc. Symposium on Discrete Algorithms, SODA'01*, pages 329–337, 2001.

[4] F. V. Fomin and K. Høie. Pathwidth of cubic graphs and exact algorithms. *Inform. Process. Lett.*, 97(5):191–196, 2006.

[5] M. Fürer. A faster algorithm for finding maximum independent sets in sparse graphs. In J. R. Corea, A. Hevia, and M. Kiwi, editors, *Proc. Latin American Symposium on Theoretical Informatics, LATIN'06*, volume 3887 of *Lecture Notes in Computer Science*, pages 491–501. Springer-Verlag, 2006.

[6] I. Razgon. A faster solving of the maximum independent set problem for graphs with maximal degree 3. In *Proc. Algorithms and Complexity in Durham, ACiD'06*, pages 131–142, 2006.

[7] J. M. Robson. Finding a maximum independent set in time $O(2^{n/4})$. Tecnical Report 1251-01, LaBRI, Université de Bordeaux I, 2001.

[8] G. J. Wœginger. Exact algorithms for NP-hard problems: a survey. In M. Juenger, G. Reinelt, and G. Rinaldi, editors, *Combinatorial Optimization - Eureka! You shrink!*, volume 2570 of *Lecture Notes in Computer Science*, pages 185–207. Springer-Verlag, 2003.