A 5/3-Approximation for Finding Spanning Trees with Many Leaves in Cubic Graphs

José R. Correa[1,⋆], Cristina G. Fernandes[2,⋆⋆], Martín Matamala[3,⋆], and Yoshiko Wakabayashi[2,⋆⋆]

[1] School of Business, Universidad Adolfo Ibáñez, Chile
correa@uai.cl
[2] Department of Computer Science, Universidade de São Paulo, Brazil
{cris,yw}@ime.usp.br
[3] Departamento de Ingeniería Matemática, Universidad de Chile, Chile
mmatamal@dim.uchile.cl

Abstract. For a connected graph G, let $L(G)$ denote the maximum number of leaves in a spanning tree in G. The problem of computing $L(G)$ is known to be NP-hard even for cubic graphs. We improve on Loryś and Zwoźniak's result presenting a 5/3-approximation for this problem on cubic graphs. This result is a consequence of new lower and upper bounds for $L(G)$ which are interesting on their own. We also show a lower bound for $L(G)$ that holds for graphs with minimum degree at least 3.

1 Introduction

The MaxLeaf consists of the following problem. Given a connected graph G, find a spanning tree in G with as many leaves as possible. This problem is NP-hard [3] even for cubic graphs [6], and is known to be MAX SNP-complete [2]. Lu and Ravi [9,10] gave the first approximation algorithms for MaxLeaf. Solis-Oba [11] described the currently best approximation algorithm: a greedy 2-approximation.

All graphs considered in this paper are connected, unless otherwise specified. We use n to denote the number of vertices of the graph in question. To the best of our knowledge, Storer [12] was the first to consider MaxLeaf on cubic graphs. He showed that every cubic graph has a spanning tree with at least $\lceil n/4 + 2 \rceil$ leaves. Griggs, Kleitman, and Shastri [4] complemented this result by showing that this bound is tight. As a side note, they also provided a simple polynomial time algorithm (alternative to Storer's) that finds a spanning tree with at least $\lceil n/4 + 2 \rceil$ leaves in a cubic graph. As an illustration, Fig. 1(a) presents a graph that achieves this bound. On the other hand, Linial and Sturtevant [7] proved that Storer's lower bound holds even for graphs with minimum degree three.

⋆ Research partially supported by CONICYT (Chile) through Anillo en Redes ACT08.
⋆⋆ Research partially supported by CNPq (Proc. 490333/04, 307011/03-8, 308138/04-0) and ProNEx - FAPESP/CNPq Proc. No. 2003/09925-5 (Brazil).

C. Kaklamanis and M. Skutella (Eds.): WAOA 2007, LNCS 4927, pp. 184–192, 2008.

Kleitman and West [5] extended the study of Linial and Sturtevant and considered MaxLeaf on graphs with minimum degree at least k, for arbitrary values of k and for small values of k as well.

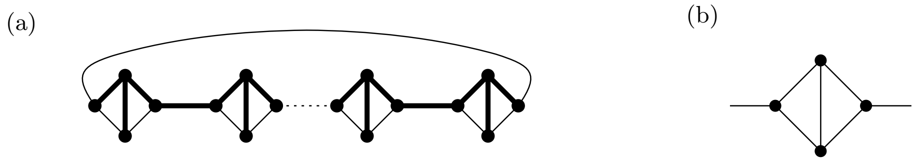

Fig. 1. (a) A cubic graph and a spanning tree with $n/4 + 2$ leaves indicated by the dark edges. (b) A diamond.

For a graph G, we let $L(G)$ denote the maximum number of leaves in a spanning tree of G. As we mentioned, the result of Storer [12] is constructive and can be restated as a proof of a lower bound on $L(G)$ for a cubic graph G. Furthermore, the main result provided by Griggs et al. [4] is a better lower bound on $L(G)$ for the case of 3-connected cubic graphs. It can actually be seen as a constructive proof of the fact that every 3-connected and also every triangle-free cubic graph has a spanning tree with at least $\lceil (n + 4)/3 \rceil$ leaves.

A *diamond* is a complete graph on 4 vertices minus an edge, also denoted by $K_4 - e$. We say that a subgraph of a given graph G is a *cubic diamond* if it is a diamond in which all of its vertices have degree 3 in G (see Fig. 1 (b)). In graphs with minimum degree at least 3, we want to distinguish those diamonds that are cubic and those that are not. The 3-*dimensional cube graph* is denoted by Q_3. Specifically, the previous bound by Griggs et al. [4] holds for all cubic graphs that do not contain diamonds. In fact, Griggs et al. observed that their bound is tight for Q_3 and that, for any other cubic graph, the sometimes stronger lower bound of $\lceil (n + 5)/3 \rceil$ holds. They also noted that this lower bound is tight for both 3-connected and triangle-free cubic graphs. (See examples in Fig. 2.)

For the purpose of this paper, it is interesting to point out that Griggs et al. result implies a 3/2-approximation for MaxLeaf in 3-connected cubic graphs, since any spanning tree in a cubic graph has at most $n/2+1$ leaves. More recently, there has been some interest in obtaining approximation results for cubic graphs. Indeed, Loryś and Zwoźniak [8] presented a 7/4-approximation for MaxLeaf in cubic graphs. Very recently, Bonsma [1] proved that if G is a connected graph of minimum degree at least 3 with d cubic diamonds, then G has a spanning tree with at least $\lceil (2n - d + 12)/7 \rceil$ leaves.

In this paper, we prove a lower bound on $L(G)$ for a cubic graph G that also takes into account the diamonds present in the graph (but not only their number). Our lower bound is always at least as good as the one for cubic graphs derived from Bonsma's lower bound.

As most previous work, our proof is constructive, so it gives a polynomial algorithm that produces a spanning tree of the given graph with as many leaves as the claimed lower bound. Our algorithm uses the one of Griggs et al. [4] for diamond-free cubic graphs. The better lower bound, together with a related

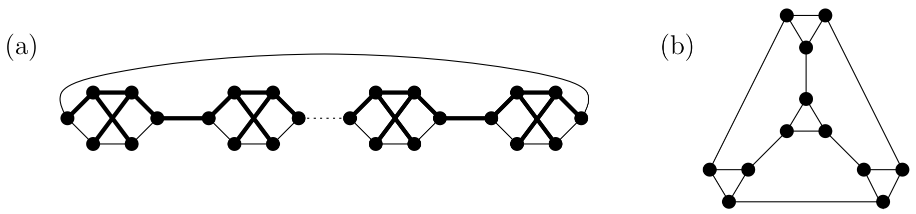

Fig. 2. (a) A triangle-free cubic graph and a spanning tree, indicated by the dark edges, with $n/3 + 2 = \lceil (n + 5)/3 \rceil$ leaves. (b) A 3-connected cubic graph G on $n = 12$ vertices obtained from K_4 by replacing each of its vertices with a triangle. Observe that $L(G) = 6 = \lceil (n + 5)/3 \rceil$.

upper bound, allows us to improve upon the result of Loryś and Zwoźniak [8], obtaining a 5/3-approximation for MaxLeaf in cubic graphs.

This paper is organized as follows. In the next section we derive the new lower bound on $L(G)$, while in Section 3, we prove the new upper bound on $L(G)$. In Section 4, we present the 5/3-approximation with its analysis. Section 5 discusses the extension for graphs with minimum degree at least 3. We conclude with some final remarks in Section 6.

2 A New Lower Bound

The way the diamonds are spread in the graph plays an important role in the new lower bound. It is expressed by a new parameter whose definition follows.

Call *internal* the two vertices in a diamond that have all neighbors within the diamond, and *external* the other two vertices of the diamond (see Fig. 3 (a)). For a cubic graph G, let G^r be the graph obtained from G after the removal of all internal vertices of its diamonds. We denote by c the number of components of G^r. For instance, if G is the graph in Fig. 1(a) with d diamonds, then G^r consists of d disjoint edges and $c = d$ in this case.

The new lower bound is given in the next theorem. It depends on the number n of vertices in the graph and on the parameter c defined above. Recall that Q_3 is the 3-dimensional cube graph.

Theorem 1. *Let $G \neq Q_3$ be a connected cubic graph with d diamonds. Then G has a spanning tree with at least* $\max\{lb_1, lb_2\}$ *leaves, where $lb_1 = \lceil (n - d + 5)/3 \rceil$ and $lb_2 = 3d - 2c + 2$. Moreover,* $\max\{lb_1, lb_2\} \geq \lceil (3n - 2c + 17)/10 \rceil$.

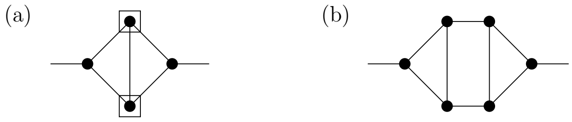

Fig. 3. (a) The squares indicate the internal vertices in a diamond. The other two vertices are the external ones. (b) A double diamond.

Proof. For the first lower bound lb_1 on $L(G)$, let G' be the graph obtained from G after replacing each diamond by the graph in Fig. 3 (b), which we call a *double diamond*. Because of the structure of G', from any spanning tree of G', it is easy to get a spanning tree of G with at most one leaf less per double diamond. The number of vertices in G' is $n' = n + 2d$. Observe that G' is diamond-free. So, from the result of Griggs et al. [4], we conclude that G' has a spanning tree T' with at least $\lceil (n' + 5)/3 \rceil = \lceil (n + 2d + 5)/3 \rceil$ leaves. Thus, from T', we can get a spanning tree T in G with at least $\lceil (n + 2d + 5)/3 \rceil - d = \lceil (n - d + 5)/3 \rceil = lb_1$ leaves. Therefore $L(G) \geq lb_1$.

For the second lower bound lb_2 on $L(G)$, let F be a forest in G consisting of spanning trees in each component of G^r. As G^r has $2d$ vertices of degree one, F has at least $2d$ leaves. Extend F in two phases to obtain a spanning tree in G. In the first phase, add to F edges from $c - 1$ of the diamonds to connect the c components of F and all vertices in these $c - 1$ diamonds. This can be done by losing two leaves and gaining one for each of the $c - 1$ diamonds. In the second phase, add edges from the remaining diamonds to connect its internal vertices to F, losing one leaf and gaining two per diamond. This results in a tree with $2d - (c - 1) + (d - (c - 1)) = 3d - 2c + 2 = lb_2$ leaves. Thus, $L(G) \geq lb_2$.

The maximum of these two lower bounds on $L(G)$ is at least the value they achieve when they are equal. That is, when $(n - d + 5)/3 = 3d - 2c + 2$. From this we deduce that $d = (n + 6c - 1)/10$ and, plugging it back in one of the two lower bounds, we get that $\max\{lb_1, lb_2\} \geq \lceil (3n - 2c + 17)/10 \rceil$. □

There are tight examples for the bound on $L(G)$ given by this theorem. For instance, the graph in Fig. 1(a) is a tight example with $c = n/4$. Indeed, Theorem 1 says that there is a spanning tree in this graph that has at least $\lceil (3n - 2c + 17)/10 \rceil = \lceil n/4 + 17/10 \rceil = n/4 + 2$ leaves. The tree of dark edges in Fig. 1(a) is optimal and has these many leaves. For another tight example, consider the graph indicated in Fig. 4. It consists of d double diamonds connected as a chain and forming a circuit, with one of the edges in each double diamond substituted by a diamond. Call this graph H. The number of vertices in H is $n = 10d$ and in this case $c = 1$. Theorem 1 says that there is a spanning tree in this graph that has at least $\lceil (3n - 2c + 17)/10 \rceil = \lceil (3n + 15)/10 \rceil = 3d + 2$ leaves. The spanning tree in dark edges in Fig. 4 is optimal and has exactly $3d + 2$ leaves.

Based on the example in Fig. 4, one might suspect that any tight example is not 3-connected after we replace each diamond by an edge. Note, however, that

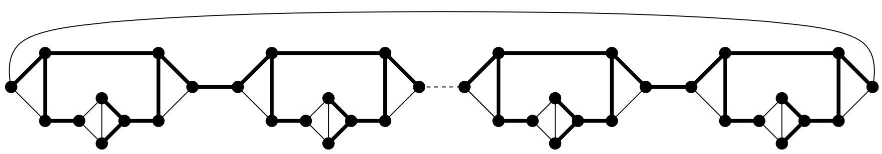

Fig. 4. A tight example for Theorem 1

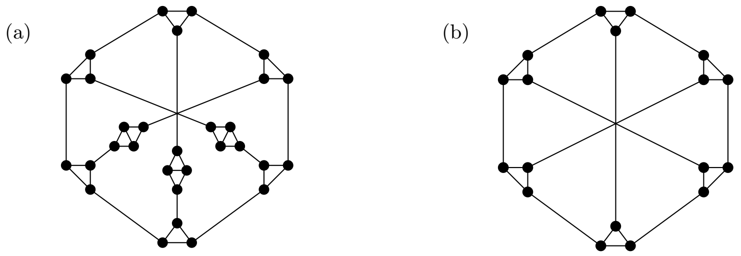

Fig. 5. (a) Another tight example for Theorem 1. (b) The 3-connected graph obtained from the example in (a) after the replacement of each diamond by an edge.

the graph shown in Fig. 5 (a) is a tight example and it remains 3-connected even after we perform these operations, as one can see in Fig. 5 (b).

2.1 Comparison with Bonsma's Lower Bound

Bonsma [1] recently proved that if G is a connected graph with d diamonds and minimum degree at least 3, then $L(G) \geq \lceil (2n - d + 12)/7 \rceil$. It is natural to ask how this result specialized to cubic graphs compares with the lower bound we have given in Theorem 1. To answer this question, let us consider the case $d \neq 0$ (when $d = 0$ the lower bound given by Griggs et al. [4] is as good as the lower bound given by Bonsma, and it is better when $n > 8$).

Let $lb_B = \lceil (2n - d + 12)/7 \rceil$. If $c = d$ then $n = 4d$ and in this case $lb_1 = lb_2 = lb_B$. If $c < d$ then $n > 4d + 1$. Adding $6n - 7d + 35$ on both sides of the last inequality, we obtain $7n - 7d + 35 > 6n - 3d + 36$. Thus, $7(n - d + 5) > 3(2n - d + 12)$, and therefore $lb_1 \geq lb_b$. (If $n \geq 4d + 22$, then $lb_1 > lb_b$.)

We note that the difference between lb_1 and lb_B might be not so negligible. For the tight example shown in Fig. 4, if we take $n = 70p$, where p is a positive integer (that is, G is a necklace with $7p$ double diamonds), we have that $lb_B = 19p + 2$, while $lb_1 = lb_2 = L(G) = 21p + 2$. In this case, lb_B is around 10 % smaller than lb_1.

3 New Upper Bound

In this section, we prove a new upper bound on $L(G)$ that involves c. We recall that c is the number of components of G^r, where G^r is the graph obtained from G after the removal of all internal vertices of its diamonds. This upper bound will be useful in the analysis of the proposed approximation, that will be presented in the next section.

Theorem 2. *If G is a connected cubic graph, then any spanning tree of G has at most $\lfloor n/2 - c + 2 \rfloor$ leaves.*

Proof. Let T be an arbitrary spanning tree in G. As G is cubic, T has $(n - d_2 + 2)/2$ leaves, where d_2 is the number of vertices of degree two in T. Indeed, denoting the number of vertices in T of degree i by d_i, for $i = 1, 2, 3$, we have that $n = d_1 + d_2 + d_3$ and $2(n-1) = d_1 + 2d_2 + 3d_3$. From these two equalities, we deduce that $d_1 = (n - d_2 + 2)/2$.

Now observe that, as G^r has c components, edges of at least $c-1$ diamonds will be used to connect components of G^r in T. Each diamond that is used to connect a component of G^r to another contributes with at least two different vertices of degree two in T. (See Fig. 6.) That is, the number of vertices of degree two in T is at least $2(c-1)$. In symbols, $d_2 \geq 2(c-1)$.

From this and from the previous observation, we deduce that T has at most $\lfloor n/2 - c + 2 \rfloor$ leaves. Hence, $L(G) \leq \lfloor n/2 - c + 2 \rfloor$. □

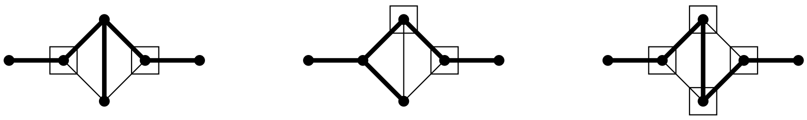

Fig. 6. Possible ways (excluding symmetric cases) to use a diamond to connect components of G^r spanning all vertices. The squared vertices have degree two in the graph of dark edges.

4 The Algorithm

Now we describe an algorithm whose approximation ratio is derived from the lower and upper bounds presented.

Algorithm $A(G)$
Input: a connected cubic graph G
Output: a spanning tree of G with at least $\frac{3}{5}L(G)$ leaves

$d \leftarrow$ number of diamonds in G
$G' \leftarrow$ graph obtained from G by substituting each diamond by a double diamond
$T' \leftarrow$ GKS(G') ▷ T' is a spanning tree of G' given by the algorithm of Griggs et al.
$T_1 \leftarrow$ spanning tree of G obtained from T' (see proof of Theorem 1)
$G^r \leftarrow$ graph obtained from G by removing the internal vertices of each diamond
$F \leftarrow$ forest consisting of a spanning tree in each component of G^r
$c \leftarrow$ number of components of G^r
$D \leftarrow$ set of $c-1$ diamonds that, if added back to G^r, make it connected
for each diamond h in D
    add to F the three edges of h incident to a same internal vertex
for each diamond h not in D
    add to F the two edges of h incident to a same external vertex
let T_2 be the resulting tree
let T be the one between T_1 and T_2 with more leaves
return T

The proof of Theorem 1 gives us immediately an algorithm to construct spanning trees with at least $\max\{lb_1, lb_2\}$ leaves. Just for completeness, we present it in pseudocode. We use GKS to refer to the algorithm of Griggs, Kleitman, and Shastri [4].

Theorem 3. *Algorithm A is a $5/3$-approximation for* MaxLeaf *on cubic graphs.*

Proof. First note that, as GKS is polynomial, A is a polynomial-time algorithm. Indeed, all but lines 3 and 8 can be implemented to run in linear time. For line 8, one can use some disjoint sets data structure and achieve almost linear time. So the most time consuming step is the execution of GKS in line 3.

As for the approximation ratio, let $|A(G)|$ denote the number of leaves in the tree produced by A with G as input. Indeed, A is a $5/3$-approximation, because

$$\begin{aligned}\frac{L(G)}{|A(G)|} &\leq \left(\frac{n-2c+4}{2}\right)\left(\frac{10}{3n-2c+17}\right) \\ &= 5\,\frac{n-2c+4}{3n-2c+17} \\ &\leq 5\,\frac{n-2c+4}{3n-2c-4c+12} \\ &= 5\,\frac{n-2c+4}{3(n-2c+4)} \\ &= \frac{5}{3}.\end{aligned}$$

The first inequality holds by Theorems 1 and 2. □

5 Constructions and Extension for Minimum Degree 3

Our lower bound shown in Theorem 1 calls attention to the fact that diamonds might not be what makes $L(G)$ smaller, closer to $n/4$. Indeed, we found interesting the following construction that proves this fact. Let H be a diamond-free cubic graph, and let T be an arbitrary spanning tree in G. Let G be the graph obtained from H by substituting every edge not in T by a diamond. Despite the fact that G has many diamonds, there exist spanning trees in G with $n/2+1$ leaves, where n is the number of vertices of G, which is as much as it could. (The number of diamonds in G is $n/6+1/3$.)

Another general construction that we found interesting is the one already exemplified in Fig. 2 (b). Given a cubic graph H, substitute each vertex of H by a triangle. Let G be the resulting graph. Note that G is (cubic) diamond-free. Then $L(G) = n/3+2$. The fact that $L(G) \geq n/3+2$ follows immediately from the lower bound of Griggs et al. [4] for cubic diamond-free graphs. On the other hand, let T be an arbitrary spanning tree of G and denote the number of vertices in T of degree i by d_i, for $i = 1, 2, 3$. Then, as already observed, $n = d_1 + d_2 + d_3$ and $2(n-1) = d_1 + 2d_2 + 3d_3$. From these two equalities, we deduce that $d_1 = d_3 + 2$. But T has at most one degree 3 vertex per triangle. So

$d_3 \leq n/3$ and $L(G) \leq n/3 + 2$. (In fact, a similar construction was described by Griggs et al. [4, p. 671].)

As already mentioned, Bonsma [1] proved that if G is a connected graph of minimum degree at least 3 with d cubic diamonds, then G has a spanning tree with at least $\lceil (2n - d + 12)/7 \rceil$ leaves. We used this bound to obtain a result similar to Theorem 1 for graphs of minimum degree at least 3.

Theorem 4. *Every connected graph G of minimum degree at least three with d cubic diamonds has a spanning tree with at least* $\max\{lb_B, lb_2\}$ *leaves, where* $lb_B = \lceil (2n - d + 12)/7 \rceil$ *and* $lb_2 = 3d - 2c + 2$. *Moreover,* $\max\{lb_B, lb_2\} \geq \lceil (3n - c + 19)/11 \rceil$.

In some cases, the bound lb_2 is better than the bound lb_B of Bonsma [1]. In fact, for the example shown in Fig. 4, if we take $n = 770p$ (that is, a necklace with $77p$ double diamonds) then $lb_B = 209p + 2$ and $lb_2 = 231p$.

Unfortunately, the upper bound for graphs with minimum degree 3 is $n - 1$ (and is tight), and therefore we cannot derive an approximation algorithm better than Solis-Oba's [11] for this case using this lower bound.

6 Final Remarks

Galbiati, Maffioli, and Morzenti [2] proved that MaxLeaf is MAX SNP-complete, but there is no such proof for cubic graphs. We suspect that this case is also MAX SNP-complete. It would be nice to settle this question.

Also, we conjecture that there is a 3/2-approximation algorithm for MaxLeaf on cubic graphs. In fact, in many cases the algorithm described in this paper achieves this ratio.

Acknowledgements

The authors would like to thank the referees for their comments and suggestions.

References

1. Bonsma, P.: Spanning trees with many leaves: new extremal results and an improved FPT algorithm. Technical report, Faculty of EEMCS, University of Twente, The Netherlands (2006)
2. Galbiati, G., Maffioli, F., Morzenti, A.: A short note on the approximability of the maximum leaves spanning tree problem. Information Processing Letters 52(1), 45–49 (1994)
3. Garey, M., Johnson, D.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco (1979)
4. Griggs, J., Kleitman, D., Shastri, A.: Spanning trees with many leaves in cubic graphs. Journal of Graph Theory 13(6), 669–695 (1989)
5. Kleitman, D., West, D.: Spanning trees with many leaves. SIAM Journal on Discrete Mathematics 4(1), 99–106 (1991)

6. Lemke, P.: The maximum-leaf spanning tree problem in cubic graphs is NP-complete. In: IMA Preprint Series 428, Mineapolis (1988)
7. SLinial, N., Sturtevant, D.: Private communication (1987) (see [5])
8. Lory, K., Zwoźniak, G.: Approximation algorithm for maximum leaf spanning tree problem for cubic graphs. In: Möhring, R.H., Raman, R. (eds.) ESA 2002. LNCS, vol. 2461, pp. 686–698. Springer, Heidelberg (2002)
9. Lu, H.I., Ravi, R.: The power of local optimization: Approximation algorithms for maximum-leaf spanning tree. In: Proceedings of 13th Annual Allerton Conference on Communication, Control, and Computing, pp. 533–542 (1992)
10. Lu, H.I., Ravi, R.: Approximating maximum leaf spanning trees in almost linear time. Journal of Algorithms 29(1), 132–141 (1998)
11. Solis-Oba, R.: 2-approximation algorithm for finding a spanning tree with maximum number of leaves. In: Bilardi, G., Pietracaprina, A., Italiano, G.F., Pucci, G. (eds.) ESA 1998. LNCS, vol. 1461, pp. 441–452. Springer, Heidelberg (1998)
12. Storer, J.: Constructing full spanning trees for cubic graphs. Information Processing Letters 13(1), 8–11 (1981)