三角形を含まない3-正則グラフの最大独立点集合を求める近似アルゴリズムについて

†マグナス ハルドソン ††吉原 貴仁

†北陸先端科学技術大学院大学 ††東京工業大学

三角形を含まない3-正則グラフの最大独立点集合を求める問題(Miscut)とその近似アルゴリズムについて考察を行なう．入力グラフを上記のように制限してもこの問題はNP困難であることが知られている[14]．本研究ではこの問題を解く二つの単純な近似アルゴリズムを提案し，それぞれのアルゴリズムの近似比率の評価を行なう．近似アルゴリズムAの近似比率とは，Aが求める独立点点集合の大きさに対する入力グラフGの最大独立点集合の大きさの比と定義する．本研究で考察するアルゴリズムは，グラフの最小次数の頂点を1つ選び独立点集合に加える．そしてこの頂点とこれに隣接するすべての頂点を取り除く．これをグラフがなくなるまで繰り返すものである．これをMINとよぶ．本研究ではMINのMiscutに対する近似比率が上下限ともに漸近的に3/2に一致することを示す．さらにMINの独立点集合の求め方に改良を加えたアルゴリズムMIN*を提案し，この近似比率の解析をMINと同様に行なった．これよりMIN*の近似比率は高々10/7であることを示す．そしてこの近似比率の値を29/21よりも小さくできないことを具体的にグラフを構成することにより示す．そしてMIN*を三角形を含まないr-正則のグラフにも適用できるように拡張を行なう．MINを使って独立点集合を求める時に，最小次数の頂点が複数存在する場合に下手な選び方をしてしまうと最適解を求められないことがある．このような時，MINが迷わずに最適解を求められるようなアドバイスをMINに与えることができたら，果たしてMINはMiscutの任意の入力に対してその最適解を常に求めることが可能か否かという問題を考える．本研究ではこの問題に対して反例となるグラフを構成し，否定的な結果を得る．

On Approximation Algorithms for Maximum Independent Set Problem on Cubic Graphs without Triangles

† Magnús M. Halldórsson †† Kiyohito Yoshihara

† Japan Adv. Institute of Science and Technology
†† Tokyo Institute of Technology

We study greedy algorithms for maximum independent sets of 3-regular graphs without triangles. It is known that the problem of finding a maximum independent set over cubic graphs without triangles is NP-hard [14]. We show that the performance ratio of a simple greedy algorithm, named MIN, is asymptotically $3/2 = 1.50$. We propose a modified greedy algorithm whose performance ratio is asymptotically at most $10/7 \approx 1.428$ and at least $29/21 \approx 1.381$ over the same domain and make this algorithm handle wilder classes of graphs. Finally, we show an inherent weakness of a family of greedy algorithms which are based on the same strategy as MIN.

1 Introduction

An *independent set* of a graph G is a subset of vertices in which no two vertices are adjacent to each other. A *maximum independent set* is an independent set with the largest cardinality. A problem of finding a maximum independent set is called Maximum independent set. Maximum independent set is one of the most famous $\mathcal{NP}$-hard problems [6], and polynomial time exact algorithms for this problem are unlikely to exist unless $\mathcal{P} = \mathcal{NP}$. Thus, it is interesting to explore algorithms that produce not always an optimal solution but a near-optimal solution as a next approach. The performance of an approximation algorithm is generally measured by the "performance ratio". The performance ratio of an approximation algorithm is defined by a ratio of the size of an optimal solution (the size of the maximum independent set) to the size of a solution (the size of an independent set) found by the algorithm (see [6, 10]). Unfortunately, Arora *et al.* [1] proved the nonapproximability of Maximum independent set. They showed this problem has no polynomial time approximation algorithm whose performance ratio is less than n^δ for some $\delta > 0$, where n is the number of vertices. Although Maximum independent set is not approximable well, and problems occurring in our real world can be reduced to this problem, some practical constraints will usually be added to such reduced instances and we do not necessarily have to deal with full general problem. It is enough for us to consider problems restricted its domain to a subclass of graphs. For instance, Baker [2] has recently shown that the problem restricted its domain to planar graphs admits a polynomial time approximation scheme. (For more information about a polynomial time approximation scheme, see [6, 10]). The problem of bounded-degree graphs has been also studied. In this paper, we will consider the problem restricted its domain to cubic (the degrees of all vertices are three) graphs without triangles (a complete subgraph of three vertices).

Maximum independent set for graphs with the maximum degree two has many polynomial time algorithms that find an optimal solution. However, when the maximum degree is three (not necessarily cubic), the problem remains $\mathcal{NP}$-hard [5]. But they are a little easier to handle than the general one. We easily find that the problem of bounded-degree has approximation algorithms with a constant ratio. The *minimum degree heuristic* is one of them. It is easy to show that, for any graphs with degree bounded by a constant Δ, the algorithm yields an independent set containing at least $n/(\Delta+1)$ vertices. By this, the ratio becomes at most $(\Delta+1)$. The best known approximation algorithm of bounded-degree graphs is designed by Berman and Fujito [3]. Their algorithm attains its ratio at most $(\Delta+3)/5$. For bounded-degree graphs with a large maximum degree, Halldórsson and Radhakrishnan [8] showed the best ratio $O(\Delta/\log\log\Delta)$. In view of the minimum degree heuristic of bounded-degree graphs, Halldórsson and Radhakrishnan [7] analyzed this greedy algorithm and showed that its ratio is $(\Delta+2)/3$ and this is the best possible. We will consider whether or not the performance ratio of the minimum degree heuristic over cubic graphs *without triangles* can be improved better than 5/3.

If a graph does not contain a large clique (complete subgraph), the graph is expected to have a large independent set. Based on this observation, there have been many studies on the size of the maximum independent set of bounded-degree graphs without triangles (complete subgraphs of three vertices) and on approximation algorithms over the domain. Although we allow graphs without triangles, the problem is still $\mathcal{NP}$-hard [14].

Let us focus on cubic graphs without triangles. If a graph is r-regular (the degrees of all vertices are r), then we easily check that the size of a maximum independent set is at most $n/2$ (We will actually verify this at Lemma 2.1 in Section 2.1). Moreover, if a graph is cubic, Stanton [13] showed that the size of the maximum independent set is at least $14n/5$ and this is the best possible from Fajtlowicz graph [4]. Shearer [11, 12] got a lower bound of the size of an independent set obtained by a random algorithm in terms of a degree sequence, which yields an independent set containing at least $17n/50$ vertices as an expected value.

In this paper, we study the minimum degree heuristic. And we consider whether or not we can improve the ratio shown by Halldórsson and Radhakrishnan [7] if we allow a given graph to be cubic *without triangles*. The minimum degree heuristic is obviously inferior to the algorithm of Berman and Fujito [3]; however, its strength lies in its simplicity and practicality. Finally, we summarize our results:

1. The ratio of the minimum degree heuristic is 3/2 asymptotically when applied to cubic graphs *without triangles* and this is the best possible by construction.

2. We propose a modified minimum degree heuristic and get a better ratio of 10/7 over the same domain. And we also show that this ratio cannot be improved less than 29/21 by construction.

3. We extend our algorithm to handle r-regular graphs without triangles.

4. We show an inherent weakness of a family of greedy algorithms based on the same strategy as the minimum degree heuristic.

Preliminaries

Maximum independent set restricted to connected cubic graphs without triangles is called Miscut.

We denote some symbols and notations used frequently in this paper. Through this paper, standard symbols and notations referred in general textbooks (see, e.g.[9]) are used so long as not remarked in particular.

n	the number of vertices
$d(v)$	the degree of a vertex v
Δ	the maximum degree of G
δ	the minimum degree of G
I	a maximum independent set
α	the size of a maximum independent set
$N(v)$	the set of vertices adjacent to v

For an algorithm A for Miscut (Maximum independent set), the *ratio* $R_A(G)$ of A is defined as follows.

$$R_A(G) \stackrel{\text{def}}{=} \frac{\alpha\ (G)}{A\ (G)} .$$

Here $A\ (G)$ stands for the size of an independent set of G found by A and $\alpha\ (G)$ the size of the maximum independent set of G. The *performance ratio* of A is defined by

$$R_A \stackrel{\text{def}}{=} \max_G R_A(G) .$$

The *absolute ratio* for A is given by

$$AR_A \stackrel{\text{def}}{=} \min_{G \text{ of } n \text{ vertices.}} \frac{A(G)}{n} .$$

2 Approximation Algorithm

We introduce our first approximation algorithm.

Once we pick a vertex of a given graph as a member of an independent set, we can no longer choose any vertices in $N(v)$. Intuitively, in order to get as large a maximum independent set as possible, we propose to take a vertex with the minimum degree. Based on this idea, a greedy algorithm for Miscut called MIN is described as in Table 1. For the recent results of bounded-degree graphs include triangles, see [7]. MIN can obviously find an independent set of G in $O(|V|+|E|)$ time.Many of results and proofs are shown in [7]. For the sake of completeness, we review these results and proofs in the next section.

```
MIN(G)   /* G is cubic without triangles.
    begin
        I ← φ ;
        while G ≠ φ do
            choose v ∈ V
            such that d(v) = min_{w∈V} d(w);
            I ← I ∪ {v} ;
            G ← G − ({v} ∪ N(v)) ;
        end − while ;
        output(I) ;
    end.
```

Table 1: A pseudo code of MIN.

2.1 The performance ratio of MIN

We shall consider a performance ratio of MIN applied to cubic graphs without triangles in this section. We denote a degree of vertex picked at the ith step as d_i. Note that the degree is not in the original graph but when it is deleted. We regard a **while** loop in Table 1 as one step. We also define integer variables t_d to be the total number of steps that MIN chooses a vertex of degree d from first step to the final tth step. That is, $t_d = |\{i : d_i = d\}|$. MIN iterates choosing a vertex v and removes v and vertices in $N(v)$ out of a graph simultaneously. Eventually, the total number of vertices deleted by MIN becomes n. Therefore, we get,

$$t_0 + 2t_1 + 3t_2 + 4t_3 = n. \qquad (1)$$

Next we evaluate the total number of edges deleted by MIN. Since MIN always picks a vertex with the minimum degree as a member of an independent set, the sum of degrees of vertices removed at the ith step is at least $d_i(d_i+1)$. Two sets e_{i2}, e_{i1} of edges deleted at the ith step are defined respectively as follows.

e_{i2}	$\stackrel{\text{def}}{=}$	the set of edges both of which end points are deleted at the ith step.
e_{i1}	$\stackrel{\text{def}}{=}$	the set of edges just one of which end point is deleted at the ith step.

Since the total sum of degrees of vertices picked at the ith step equals to the twice of the cardinality of e_{i2} plus that of e_{i1}, the following inequality holds.

$$2|e_{i2}| + |e_{i1}| \geq d_i(d_i + 1). \tag{2}$$

Furthermore, since graphs contain no triangles, there is no edges among the vertices in $N(v)$. Then,

$$|e_{i2}| \leq \binom{d_i + 1}{2} - \binom{d_i}{2}. \tag{3}$$

By (2) and (3), the total number of edges deleted at the ith step is at least $\binom{d_i+1}{2} + \binom{d_i}{2}$. Thus, the number of edges removed by MIN from the first step to the final tth step becomes at least

$$\sum_{i=1}^{t} \left\{ \binom{d_i + 1}{2} + \binom{d_i}{2} \right\} = \sum_{i=1}^{t} d_i^2 .$$

Every cubic graph has $3n/2$ edges. Thus,

$$|E| = 3n/2 \geq \sum_{i=1}^{t} d_i^2 .$$

In terms of t_d,

$$t_1 + 4t_2 + 9t_3 \leq 3n/2. \tag{4}$$

We try to solve an integer programming problem with the above constraints (1) and (4). By observing the behavior of MIN and the structure of our inputs in detail, we notice that there are some other constraints which allows us to sharpen the ratio. Here we give such constraints as the following lemma.

Lemma 2.1. *The following two inequalities and an equation hold.*

$$t_0 + t_1 + t_2 + t_3 \leq \alpha \leq n/2. \tag{5}$$

$$nt_0 + t_1 \geq 1. \tag{6}$$

$$t_3 = 1 \tag{7}$$

Proof.

The proof of (5)

For any independent set I (maximum or not), the number of edges incident to the vertex in I equals $3|I|$, while the total number of edges in cubic graphs is $3n/2$. Therefore, $|I| \leq n/2$.

The proof of (6)

We show the inequality by contradiction. That is, we can assume a graph such that $t_0 = t_1 = 0$ holds. We focus on the final tth stage of MIN. Since $t_1 = 0$ and $t_0 = 0$, there is no possibility for a vertex with $d\ (v) = 1$ to be picked. Neither the graph can be an isolated point. Therefore, the graph must be a triangle in the last stage. Since we restrict our input graphs to cubic without triangles, this leads to contradiction.

The proof of (7)

Since our inputs are restricted to connected cubic graphs without triangles, MIN picks such a vertex $d(v) = 3$ only at the first step. Thus, for an arbitrary cubic graph, t_3 must be one. □

With the argument above, we find a lower bound of AR_{MIN} by solving the following integer programming problem.

$$\begin{array}{lllc}
\text{minimize} & t_0 + t_1 + t_2 + t_3 & & \\
\text{subject to} & t_0 + 2t_1 + 3t_2 + 4t_3 & = & n \\
 & t_1 + 4t_2 + 9t_3 & \leq & 3n/2 \\
 & t_0 + t_1 + t_2 + t_3 & \leq & n/2 \\
 & t_0 + t_1 & \geq & 1 \\
 & t_3 & = & 1 \\
 & t_i \geq 0\ (0 \leq i \leq 3)\ \in \mathrm{Z} & &
\end{array}$$

It is easy to see that the problem we have to deal with is finally reduced to the two-variable integer programming problem. Solution of this problem relaxed to linear programming problem is $(t_1, t_2) = (1, \frac{n-6}{3})$. We can derive a solution of original integer programming problem from these. Thus, we get $AR_{\mathrm{MIN}} \geq 1/3$.

Theorem 2.2.

$$AR_{\mathrm{MIN}} \geq 1/3.$$

2.2 Bipartite Graphs

Here, we show that MIN can always get an optimal solution if a given graph does not have any odd cycles. We begin with giving a famous graph theoretical fact.

Proposition 2.3. ([9]) *For an arbitrary graph G (not necessarily cubic without triangles), G has no odd cycles if and only if G is a bipartite graph.*

Lemma 2.4. *For r-regular graph G, G is bipartite if and only if $\alpha = n/2$.*

Proof. See [14]. □

We get the following theorem by Proposition 2.3 and Lemma 2.4.

Theorem 2.5. *If G is bipartite and r-regular, then* MIN *can always find an maximum independent set of G.*

Proof. See [14]. □

Theorem 2.6.

$$R_{\text{MIN}} \leq 3/2 - 3/n \tag{8}$$

Proof. For all cubic graphs without triangles, MIN can get an independent set containing at least $n/3$ vertices by Theorem 2.2. Thus, the ratio is at most $R_{\text{MIN}} \leq \frac{\alpha}{n/3}$. If a graph does not contain any odd cycle, that is, bipartite, then MIN can always find its maximum independent set by Theorem 2.5. Since we analyze the worst cases, it is sufficient to consider graphs containing a few odd cycles. If a cubic graph contains an odd cycle, then the size of a maximum independent set is at most $\frac{n}{2} - 1$. Therefore, the performance ratio is at most $(\frac{1}{2}n - 1)/\frac{1}{3}n$. □

2.3 Nonapproximability of MIN

In this section, we show that graphs for which MIN may miss an optimal solution. The following graph G_{hard} (we call this graph "hard graph" for simplicity) is one of instances which MIN does not necessarily find an optimal solution in the worst case. This graph is consisted of three subgraphs called *front unit*, *repetition unit*, and *back unit*. The entire of this hard graph is illustrated in Figure 1. We easily see that the hard graph is cubic and contains no triangles.

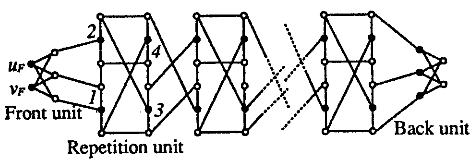

Figure 1: An example of a hard graph G_{hard}. The black vertices will be put in an independent set by MIN.

We next show that the worst behavior of MIN. At the first step, assume that MIN may remove the vertex u_F with degree three in the front unit. After that, this front unit resulted in an isolated vertex v_F. MIN should pick this vertex at the second step. From the third step to the $(t-3)$th step, for each repetition unit, MIN may pick the four vertices in order of the numbers shown in the leftmost repetition unit in Figure 1. We verify easily that after removing all four vertices, MIN will have to treat the same graph except that just one repetition unit lost. After removing all repetition units, MIN will pick three vertices from the back unit in an arbitrary order.

Finally, we evaluate the performance ratio when MIN works as above. The number of vertices n in the hard graph G_{hard} is equal to $12m + 10$ if we assume the number of repetition units is m. The size of the independent set of this hard graph G_{hard} obtained by MIN is $4m+5$. Since an repetition unit is bipartite, the size of the maximum independent set is $6m + 4$. Combining these, we get,

$$R(G_{\text{hard}})_{\text{MIN}} = \frac{6m+4}{4m+5} = \frac{3}{2} - \frac{21}{2(n+5)} \ . \tag{9}$$

Theorem 2.7.

$$R_{\text{MIN}} \approx 3/2.$$

Proof. Followed immediately by (8) and (9). □

3 Modified Algorithm

3.1 Strategy and Performance Ratio

The worst behavior of MIN when applied to the hard graphs G_{hard} in the previous section suggests us some way to modify a strategy of MIN. A situation that MIN reveals its weakness might happen whenever there exist many vertices of the same degree. In this section, we propose a modified version of MIN, named MIN*. Our idea is as follows:

> *Suppose the minimum degree of a graph is two and there are several vertices of degree two. If there exists a vertex such that degree of at least one of its neighbor is three,* MIN* *should always pick that vertex.*

If no such vertex exists and the minimum degree is two, then we show that the graph is composed of several disjoint cycles. In such a case, MIN* can always get an optimal solution of the graph.

Lemma 3.1.

$$AR^*_{\text{MIN}} \geq \frac{7}{20}n - \frac{1}{10}.$$

Proof. Here we consider the performance ratio of MIN*. For our analysis, we recall the degree sequence in Section 2. Moreover, we divide this sequence into two disjoint subsequences X and Y as in Figure 2. While MIN* yields a subsequence X, sc Min* necessarily selects a vertex at least one of whose neighbor is of degree more than two if MIN picks a vertex of degree two. We call this type of picking *β reduction*. While MIN* produces a subsequence Y, MIN* picks a vertex whose both neighbor are of degree two if MIN* picks a vertex of degree two. We call this type of picking γ *reduction* similarly. Observe that if one β reduction is executed, then the number of edges deleted in this step is at least five. Analogously, if one γ reduction is executed, exact four edges are removed. Furthermore, we define $t_{2\beta}$ and $t_{2\gamma}$ as the total number of executions of β reductions and γ reductions from the first step to the final step, respectively. By our strategy, we can assume that a resulting graph is composed of a few disjoint cycles after removing the vertex of degree d_m, which is the last chosen vertex when MIN* yields a subsequence X. In the following several steps after removing the vertex of degree d_m, MIN* proceed optimally for the resulting disjoint cycles.

$$\overbrace{\boxed{d_1, d_2, \cdots, d_m}}^{X}\ \overbrace{\boxed{d_{m+1}, d_{m+2}, \cdots, d_t}}^{Y}$$

Figure 2: A degree sequence produced by MIN_4^*.

With the new variables, $t_{2\beta}$ and $t_{2\gamma}$, we amend the equations (4) and (2) in Section 2 as below, respectively.

$$t_0 + 2t_1 + 3(t_{2\beta} + t_{2\gamma}) + 4t_3 = n. \qquad (10)$$

$$t_1 + 4t_{2\gamma} + 5t_{2\beta} + 9t_3 \leq 3n/2. \qquad (11)$$

Note that γ reductions occur only in subsequence Y by our strategy. Moreover, since our input graphs contain no triangles, the shortest length cycle is C_4. Hence, if one γ reduction is executed, then there happens at least one selection of a vertex of degree zero or one. By this, we get

$$t_{2\gamma} \leq t_0 + t_1 \qquad (12)$$

Now, we obtain the following inequality by $5 \times (10) - (11) + (12)$.

$$6t_0 + 10t_1 + 10t_{2\gamma} + 10t_{2\beta} + 11t_3 \geq \frac{7}{2}n.$$

We saw that $t_3 = 1$ in Lemma 2.1. Thus,

$$AR^*_{\text{MIN}} \geq \frac{7}{20}n - \frac{1}{10}.$$

□

Theorem 3.2.

$$\frac{29}{21} \leq R_{\text{MIN}^*} \leq \frac{10}{7}.$$

Proof. First, we prove the upper bound. Since the size of maximum independent sets in cubic graphs which MIN* might not always get an optimal solution are at most $n/2 - 1$ by 2.5. Combining with the result of Lemma 3.1, the performance ratio of MIN* is at most $\frac{n/2-1}{7n/20-1/10}$, which is equal to 10/7 asymptotically. On the lower bound, we can construct a graph whose ratio is 29/21 asymptotically when applied to MIN* in the worst case in the same way as in the Section 2.3. The graph is consisted of three subgraphs and has repetition unit similar to the hard graph in Section 2.3. Due to its large and complicated construction and lack of space, we omit to draw it exactly here. □

3.2 Extensions

In this section, we make our modified algorithm MIN* be applicable to a wider class of r-regular graphs without triangles, where $r \geq 4$ is a constant. But we will state only an idea due to lack of space. We derive the performance ratio of the algorithm with the same way as in Section 3.1 (see [14]).

At first, we propose an algorithm MIN_r^* for r-regular graphs without triangles. If the minimum degree of a given graph is δ and there exist a few vertices with degree δ at some step, MIN_r^* should choose a vertex v such that at least one vertex in $N(v)$ is degree more than δ. If there exists no such a vertex, we claim that the graph at the step is consisted of several δ-regular components without triangles. In such a case, we should employ MIN_δ^* for these components henceforth. This is an idea. We give the detail of MIN_r^* in Table 2.

$\text{MIN}_r^*(G)$
  /* G is r-regular and triangle-free
  begin
    $I \leftarrow \phi$;
    while $G \neq \phi$ **do**
      choose $v \in V$
        such that $d(v) = \min_{w \in V} d(w)$;
      if there exists u
        such that $u \in N(v)$
          and $d(u) > d(v)$
      then $I \leftarrow I \cup \{v\}$;
          $G \leftarrow G - (\{v\} \cup N(v))$;
      else $\text{MIN}_{d(v)}^*(G)$;
    end – while ;
    output(I) ;
  end.

Table 2: A pseudo code of MIN_r^*.

4 Algorithm with Advice

In the previous sections, we considered the performance ratios of two greedy algorithms. These two algorithms have a common basic strategy of removing a vertex with the minimum degree in a graph at each stage. We easily see by the worst case behaviors of them applied to their each hard graph that weaknesses of the two greedy algorithms, MIN and MIN*, appear when they have several possible ways of picking. If we could give these algorithms some advice such that they could proceed optimally whenever they face branch road, how would the algorithms work better? Or could such an algorithm with the advice necessarily catch an optimal solution?

In this section, we consider ability of an algorithm with advice such that an algorithm could always choose a vertex in an optimal solution whenever there are a few vertices of the same minimum degree. For short, we call this ultimate algorithm $\textsc{Min}^{\text{Advice}}$ in this section. And we will show that even if we employ $\textsc{Min}^{\text{Advice}}$, there remains graphs $\textsc{Min}^{\text{Advice}}$ can not find an optimal solution. This reveals that a family of algorithms based on the same strategy as MIN are substantially weak.

In the following, we shall construct graphs $G^{\text{A}}_{\text{hard}}$ such that $\textsc{Min}^{\text{Advice}}$ might miss an optimal solution. We fix a maximum independent set I of $G^{\text{A}}_{\text{hard}}$. First, we make a subgraph H^{A}_5 as in Figure 3.

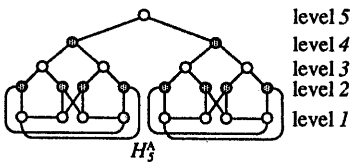

Figure 3: A subgraph H^{A}_5 of $G^{\text{A}}_{\text{hard}}$.

The size of the maximum independent set of H^{A}_5 is 10, whereas just 9 when we assume H^{A}_5 are directly applied to $\textsc{Min}^{\text{Advice}}$ as an instance. The shaded vertices in Figure 3 are in I. Let the level of the root of H^{A}_5 be 5th and the levels of the other vertices in H^{A}_5 be as in Figure 3 respectively. Next, we construct a pseudo binary tree with four H^{A}_5s as its leaves, which is illustrated in Figure 4. Note that the level increases just by 2. We make a pseudo binary tree with its top level $2n+1$ ($n \geq 2$) by repeating the same operation $n-2$ times. Denote this pseudo binary tree H^{A}_{2n+1}.

Finally, we join two H^{A}_{2n+1}s as in Figure 4 in order to make the entire graph cubic and triangle-free. This graph is $G^{\text{A}}_{\text{hard}}$.

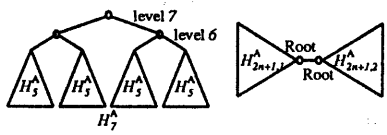

Figure 4: A pseudo binary tree H^{A}_7 and a brief construction of $G^{\text{A}}_{\text{hard}}$.

Now, we pursue a behavior of $\textsc{Min}^{\text{Advice}}$ over $G^{\text{A}}_{\text{hard}}$. An optimal solution I of $G^{\text{A}}_{\text{hard}}$ is containing only and all vertices on the even levels of each $H^{\text{A}}_{2n+1,1}$ and $H^{\text{A}}_{2n+1,2}$. Assume that $\textsc{Min}^{\text{Advice}}$ picks any vertex in $H^{\text{A}}_{2n+1,1}$ at the first step. It is easy to verify that $\textsc{Min}^{\text{Advice}}$ can proceeds optimally and find an optimal solution of $H^{\text{A}}_{2n+1,1}$. But after that $\textsc{Min}^{\text{Advice}}$ will face an critical situation that the degree of all vertices in I are three and the minimum degree is two. That is, $\textsc{Min}^{\text{Advice}}$ must unwillingly choose the degree two vertex of the root of $H^{\text{A}}_{2n+1,2}$ *not* in I. What is worse, at the next step, $\textsc{Min}^{\text{Advice}}$ will again face the same situation as before, since each $H^{\text{A}}_{2n+1,2}$ is made in a pseudo *binary tree*. Critical situations will continue until all the vertices on the fifth level of $H^{\text{A}}_{2n+1,2}$ are removed.

Theorem 4.1. *For $G^{\text{A}}_{\text{hard}}$ of two subgraphs H^{A}_{2n+1},*

$$R_{\textsc{Min}^{\text{Advice}}}(G^{\text{A}}_{\text{hard}}) \approx 16/15.$$

Proof. We estimate $R_{\textsc{Min}^{\text{Advice}}}(G^{\text{A}}_{\text{hard}})$. Let the size of the maximum independent set of a subgraph H^{A}_{2n+1} be h_{2n+1} and the size of an independent set found by $\textsc{Min}^{\text{Advice}}$ be h'_{2n+1} respectively. We easily check that the following two recurrence relations hold by its construction for $n \geq 3$.

$$\begin{cases} h_5 = 10, \\ h_{2n+1} = 4h_{2n-1} + 2. \end{cases} \quad \begin{cases} h'_5 = 9, \\ h'_{2n+1} = 4h'_{2n-1} + 1. \end{cases}$$

From these, we get,

$$h_{2n+1} = 4^{n-2} \cdot \frac{32}{3} - \frac{2}{3}.$$

$$h'_{2n+1} = 4^{n-2} \cdot \frac{28}{3} - \frac{1}{3}.$$

The size of the maximum independent set of $G^{\text{A}}_{\text{hard}}$ is $2 \times h_{2n+1}$ and the size of an independent set found by $\text{MIN}^{\text{Advice}}$ is $h_{2n+1} + h'_{2n+1}$. Thus, the ratio becomes,

$$R_{\text{MIN}^{\text{Advice}}}(G^{\text{A}}_{\text{hard}}) = \frac{4^{n-2} \cdot \frac{64}{3} - \frac{4}{3}}{4^{n-2} \cdot \frac{60}{3} - \frac{1}{3}} .$$

For a large n, $R_{\text{MIN}^{\text{Advice}}}(G^{\text{A}}_{\text{hard}})$ is asymptotically equal to $16/15$ ($\approx 1.0\dot{6}$). □

5 Conclusion

In this paper, we analyzed some greedy algorithms for Maximum independent set of cubic graphs without triangles and evaluated their performance ratios. Restricting our domain, we found that a naïve greedy algorithm, MIN, attains its performance ratio $3/2 = 1.50$ and this is the best possible by construction whereas the ratio for graphs with maximum degree three and including triangles is $5/3 \approx 1.\dot{6}$ [7]. Next, we modified MIN by observing its worst case behavior and derived a better ratio, $10/7 \approx 1.428$. And we showed this ratio could not be improved less than $29/21 \approx 1.381$ by construction. Furthermore, We extended our modified algorithm to wider classes of graphs, such as 4-regular graphs without triangles. Finally, we considered the most powerful greedy algorithm and showed its drawback by construction. This implies that a family of greedy algorithms designed with the same strategy as MIN is inherently weak.

Acknowledgements

We would like to thank Osamu Watanabe for informative discussions and suggestions.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proc. of 33rd IEEE Symp. on Found. of Comp. Sci.*, pages 14–23, October 1992.

[2] B. S. Baker. Approximation algorithms for NP-complete problems on planer graphs. *J. ACM*, 41:153–180, 1994.

[3] P. Berman and T. Fujito. Approximating Independent Set in Degree 3 Graphs. *Unpublished manuscript*, 1994.

[4] S. Fajtlowicz. The independence ratio for cubic graphs. In *Proc. 8th Southeastern Conf. on Combinatorics, Graph Theory, and Computing*, pages 273–277, 1977.

[5] M. R. Garey and D. S. Johnson. Some Simplified NP-Complete Graph Problems. *Theor. Comput. Sci.*, 1:237–267, September 1974.

[6] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[7] M. M. Halldórsson and J. Radhakrishnan. Greedy is Good:Approximating Independent Set in Sparse and Bounded-degree Graphs. In *Proc. of 26th Symp. on Theory of Computing*, pages 439–448, May 1994.

[8] M. M. Halldórsson and J. Radhakrishnan. Improved approximation of independent sets in bounded-degree graphs. In *Proc. Fourth Scand. Workshop on Algorithm Theory*, pages 195–206. Springer-Verlag LNCS No.824, 1994.

[9] K. Mehlhorn. *Graph Algorithms and NP-Completeness, Data Structures and Algorithms* . Springer-Verlag, 1984.

[10] C. H. Papadimitriou. *Computational Complexity.* Addison Wesley, 1994.

[11] J. B. Shearer. A note on the independence number of triangle-free graphs. *Discrete Math.*, 46:83–87, August 1983.

[12] J. B. Shearer. A note on the independence number of triangle-free graphs, II. *J. Combin. Theory Ser. B*, 53:300–307, January 1991.

[13] W. Stanton. Some Ramsey-type numbers and the independent ratio. *Trans. Amer. Math. Soc.*, 256:353–370, 1979.

[14] K. Yoshihara. The Approximation Algorithms for Maximum Independent Set Problem of Cubic Graphs without Triangles. Master's thesis, Tokyo Institute of Technology, 1995.