This is a preprint of an article published in Journal of Combinatorial Optimization.

The final authenticated version is available online at:
https://doi.org/10.1007/s10878-018-0277-7.

The game chromatic index of some trees with maximum degree four and adjacent degree-four vertices

Wai Lam FONG, Wai Hong CHAN

Department of Mathematics and Information Technology, The Education University of Hong Kong, Tai Po, Hong Kong SAR, China.

Ge NONG

Department of Computer Science, Sun Yat-sen University, Guangzhou, China, and SYSU-CMU Shunde International Joint Research Institute, China.

Abstract

We prove that the game chromatic index of trees of maximum degree 4 without three adjacent 4-vertices (degree-four vertices) is at most 5. This relaxes the assumption that the trees do not contain adjacent 4-vertices in a result of Chan and Nong [The game chromatic index of some trees of maximum degree 4. Discrete Appl. Math., 170 (2014), 1-6].

Keywords: game chromatic index, tree, graph coloring game, game chromatic number

1. Introduction

In this paper, we consider the edge-coloring game studied in [2, 5, 6], in which two players, Alice and Bob, alternatively choose a color from a set of colors to color an uncolored edge of an initially uncolored finite and simple graph G such that no adjacent edges receive the same color. Alice wins the game if all edges of G are finally colored successfully; otherwise, Alice loses. Bob takes the first move and he is permitted to skip turns throughout the game, while skipping is not allowed for Alice. The parameter *game chromatic index* $\chi'_g(G)$ of a graph G, which was introduced by Cai and Zhu [4], is defined as the smallest natural number n so that Alice has a winning strategy for the game played on G with n colors. A similar type of games was introduced by Bodlaender [3], in which nodes are colored instead of edges, and the corresponding parameter is called *game chromatic number*.

For any tree T with the maximum degree Δ, Cai and Zhu [4] showed that $\chi'_g(T) \leq \Delta + 2$. Erdős *et al.* [6] then proved that the best possible upper bound is $\Delta + 1$ for trees with $\Delta \geq 2$. This best possible bound is achieved when $\Delta = 3$ [1] or $\Delta \geq 5$ [2, 6]. As this bound trivially holds for $\Delta = 2$, only the case $\Delta = 4$ was left open. For this remaining case, Chan and Nong [5] proved that the bound $\Delta + 1$ is also effective and sharp when T does not contain adjacent 4-vertices or, when T is a caterpillar, which may contain a chain of 4-vertices. In this paper, we prove that the bound 5 is still valid when T contains adjacent 4-vertices and even it is not a caterpillar, provided that no three 4-vertices are adjacent.

2. Trees with $\Delta = 4$ and without three adjacent 4-vertices

THEOREM 1. *Let T be a finite tree with $\Delta = 4$ and containing no three adjacent 4-vertices. Then $\chi'_g(T) \leq 5$.*

Before presenting the proof of the above theorem, we first define several terms as follows:

- A *leaf* is the edge incident with a pendant vertex.
- The *root of a leaf* is the non-pendant vertex incident with a leaf.
- A *trivial path* is a path of length zero.
- A *star-node* is a vertex connected to three or more roots of colored leaves by edge-disjoint (maybe trivial) paths.
- A *k-SN* is a star node connected to exactly k roots of colored leaves by edge-disjoint (maybe trivial) paths.
- A *star-edge* is an edge incident with a star-node.
- A *star-path* is the path connecting two star-nodes.
- A *k-leaf-colored tree* (*k-LCT*) is a tree containing exactly k colored leaves.

- A *v-branch* of a tree containing a vertex v is its maximal subtree which consists of exactly one edge incident with v. A v-branch with no colored edges is called an *uncolored v-branch*.

We have the following remarks for a tree with $\Delta = 4$, based on the above definitions:

1. Each star-node is either a 3- or 4-vertex.

2. A tree has no star nodes if and only if it has at most two colored leaves. A 3-LCT has exactly one 3-SN; a 4-LCT has either exactly one 4-SN or exactly two 3-SN.

Here are some notations and remarks about our figures. Let $\{1, 2, 3, 4, 5\}$ be the color set, and letter a be any color in the color set. A *rectangle*, a *triangle* and a *circle* represent a 4-vertex, a 3-vertex and a vertex, respectively. A *dashed edge* and its two end vertices jointly represent an uncolored path with any length, and this path may be trivial. For example, in Figure II(d), vertex s is incident with the edge with color a if the path is trivial. Uncolored subgraphs may be incident with vertices in the figures.

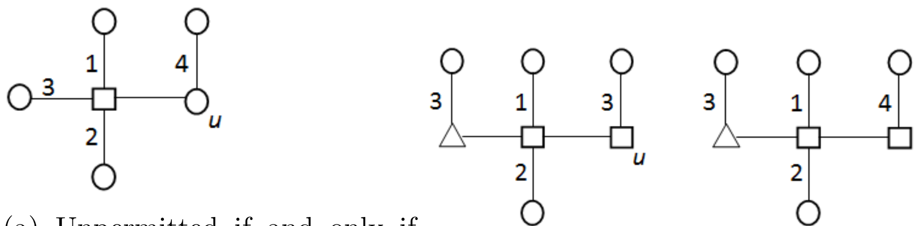
During the game on T, colored edges can be interpreted as *boundaries* to split T into *subtrees*, and each boundary belongs to exactly two subtrees. When an edge is being colored, the subtree containing it will be split into two. For convenience, we define that when a leaf is being colored, the subtree containing it will be split into a K_2 and the subtree itself. Then, we may consider each subtree independently as it is clear that subsequent coloring of any subtree will not affect that of one another.

We call the following types 1 to 8 subtrees *permitted*. Subtrees which are neither completely colored nor permitted are called *unpermitted*.

1. Trees containing no star-nodes.

2. Trees containing exactly one star-node, except those in Figures I(a), I(b) and I(c).

3. A tree with all uncolored edges on it forming a path of length at most three. Moreover, if the length of this path is one (an edge), there is at least one color available for this edge; if the length of it is two or three, there are at least two colors available for each uncolored edge.



(a) Unpermitted if and only if $d(u) \geq 3$.

(b)

(c)

Figure I: Some unpermitted subtrees with exactly one star-node.

4. A tree with all uncolored edges of it forming the union of a path $P = v_0v_1...v_m$ for $3 \leq m \leq 4$ and a tree T_s where all vertices of T_s are not incident with any colored edges and v_m is the unique common vertex of P and T_s. Moreover, at least four colors are available for $v_{m-1}v_m$ and at least two for each of the remaining uncolored edges of P. An example of this type with $m = 4$ is shown in Figure II(a).

5. The tree in Figure II(b).

6. The tree in Figure II(c).

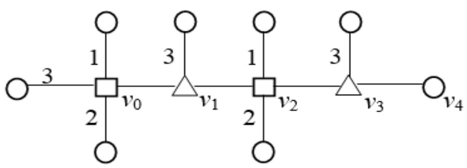
7. The tree in Figure II(d).

8. The tree in Figure II(e).

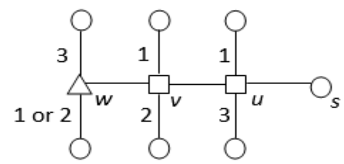
In order to show five colors are enough for Alice to win the game, we need to give a strategy of Alice for choosing and coloring an edge in each of her turns such that Lemma 2 holds. Before that, we first prove that Theorem 1 can be derived from Lemma 2. In the meantime, we would give another lemma that there would be at most one unpermitted subtree after each move of Bob.

Lemma 2. *Suppose after an Alice's move, each subtree is either completely colored or permitted; then, no matter what Bob's move is in his turn, Alice may keep all subtrees completely colored or permitted after her next move.*

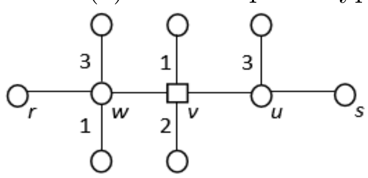
Proof of Theorem 1. It is sufficient to prove that Alice always has an available move in her turns. Since Bob can skip any of his turns and the tree is assumed finite, Alice loses the game if and only if Alice doesn't have a move in some of her turns.



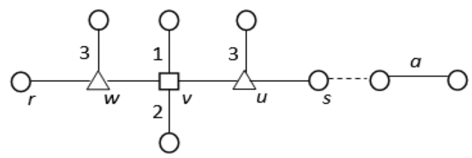
(a) An example of type 4.



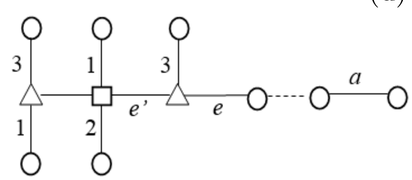
(b) Type 5



(c) Type 6. Vertex r exists if w is a 4-vertex.



(d) Type 7



(e) Type 8

Figure II: Permitted types 4-8 subtrees.

Next, we prove that Alice always has a move such that, right after her moves, all subtrees are permitted or completely colored, by induction on the number of Alice's turns. In her first turn, she only needs to face a 0- or 1-LCT, so it is clear that she can make a move, and, right after her first move, each subtree has at most 2 colored edges. All subtrees are then either completely colored or of type 1. As a result, if Lemma 2 holds, by induction, Alice can keep all subtrees permitted if they are not completely colored after each of her turns. ∎

LEMMA 3. *If Bob colors an edge of a permitted subtree, at most one of the two subtrees formed is unpermitted.*

Proof of Lemma 3. Suppose T is a permitted subtree and that Bob is going to color one edge of T. Since, on the one hand, any subtree with at most three colored edges is permitted, so two unpermitted subtrees must have at least eight colored edges in total. On the other hand, after Bob has colored an edge of T, the two newly formed subtrees should have two more colored edges than that were in T, because the edge which has just been colored is double counted in the two subtrees. Therefore, the lemma holds if the total number of colored edges of T is less than 6. We can also see that each of types 1, 2, 6 and 7 subtrees has less than six colored edges.

Suppose T is of type 3. Bob would only color any edge of the uncolored path of length at most three. At least one of the subtrees then formed is completely colored or of type 3.

Suppose T is of type 4. When Bob colors any one edge of P, one of the two newly formed subtrees must be completely colored, of type 1 or of type 3; when Bob colors other edges, one of the two newly formed subtrees must contain only one colored edge, and so is permitted.

Suppose T is of types 5 or 8. No matter which edge Bob colors, one of the two newly formed subtrees must contain at most three colored edges, and so is permitted. ∎

Proof of Lemma 2 and Alice's strategy

Owing to the property shown in Lemma 3 that Bob will generate at most one unpermitted subtree in each of his move, Alice's task is to turn F, the unpermitted subtree if any, or, otherwise, a permitted subtree, to permitted or completely colored subtrees, in each of her turns in order to maintain all subtrees being permitted. In the following, we will first introduce Alice's strategies for handling types 1 and 2 subtrees and the unpermitted subtrees which would be generated by Bob in his last move on subtrees of types 1 and 2. After that, we will give, one by one, Alice's strategies for handling subtrees of each of types 3 to 8 and the unpermitted subtrees which were generated by Bob in his last move on subtrees of types 3 to 8, respectively. We note that 1) all subtrees are permitted if the color just added by Bob is removed; 2) some subtrees may belong to two permitted types simultaneously, for example, types 2 and 3. In this case, Alice may use any one of the strategies for dealing with these two types of subtrees.

Types 1 and 2 subtrees

We first consider unpermitted subtrees generated from a subtree of type 1 or 2 by Bob in his last move. For those from type 1, at most one of them contains a star-node. For those from type 2, any 5-LCT obtained by coloring an edge of a 4-LCT with one 4-SN must consist of one 4-SN and one 3-SN. In addition, all types 1 and 2 subtrees contain zero and one star-node, respectively. Therefore, F, the subtree Alice is going to put a color on its edge, may contain zero, one or two star-nodes. The following is a proposed strategy of Alice for dealing with these cases with respect to the number of star-nodes in F. Unless specified otherwise, Alice may use any available colors which can form the desired types of subtrees.

- F does not have star-nodes.

  Alice may color any edge to make subtrees which are completely colored, of type 1 or type 2.

- F has exactly one star-node v.

  (I) F is a 3-LCT:

  - If F has three colored star-edges, Alice may color the remaining star-edge to make a completely colored and a type 1 subtrees.
  - If F has at most two colored star-edges, Alice may color a star-edge to make one type 1 and one type 2 subtrees.

  (II) F is a 4-LCT:

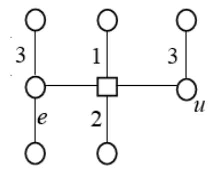


Figure III: An exception in Alice's strategy: under the constraint that $d(u) \geq 3$.

- If F has exactly three colored star-edges, Alice may color the remaining star-edge to make one completely colored and one type 1 subtrees.
- Given that F has exactly two colored star-edges. If F is the subtree in Figure III, Alice may color edge e with 1 to make one type 1 and one type 6 subtrees. Otherwise, Alice may color a star-edge to make one type 1 and one type 2 subtrees. After Alice colored that star-edge, a 4-LCT with one star-node and three colored star-edges will be formed, and this 4-LCT must be permitted in this case.
- Given that F has exactly one colored star-edge. The 4-SN v is adjacent to at most one 4-vertex. If this 4-vertex exists, say u, Alice may color the star edge vu to to make one type 1 subtree and a 4-LCT with the 4-SN v and exactly two colored star-edges. Because v is not adjacent to any 4-vertex in the 4-LCT, the 4-LCT is neither the one in Figure I(b) nor I(c), which implies it is of type 2. Similarly, if the 4-vertex adjacent to v doesn't exist, Alice may color any star-edge to make one type 1 subtree and a 4-LCT with exactly two colored star-edges of type 2.
- If F has no colored star-edges, Alice may color any star-edge to make one type 1 and one type 2 subtrees.

- F has exactly two star-nodes: Two 3-SN when F is a 4-LCT, or a 4-SN v and a 3-SN u when F is a 5-LCT.

  (I) F is a 4-LCT:

  Alice may color any edge on the star-path to form two 3-LCTs. Both of them are of type 2.

  (II) F is a 5-LCT:

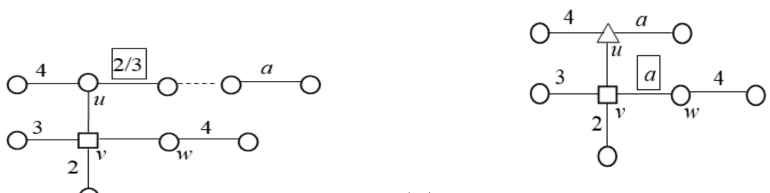
- Suppose v and u are not adjacent. If v is incident with three colored star-edges, Alice may color the remaining star-edge incident with v to make one completely colored and one type 2 subtrees. If v is incident with two colored star-edges, she may color the star-edge incident with v and on the star-path to form two type 2 subtrees. If v is incident with one or no colored star-edge, she may color the star-edge incident with u and on the star-path to form two type 2 subtrees because the formed 4-LCT with a 4-SN has at most one colored star-edge.
- Suppose v and u are adjacent. There is always an available color for uv, i.e., uv should not be surrounded by edges with five colors on them. If not, before Bob's last move, at least four colors were on the edges surrounding uv such that the subtree containing uv was not permitted. Then, Alice may put an available color to uv only if no unpermitted subtrees would form. It is clear that when uv is colored, F would be separated into one 4-LCT with a 4-SN v and one 3-LCT with a 3-SN u, where the former is type 2 (permitted) unless it is a subtree shown in Figures I(a), I(b) or I(c) while the latter must be type 2 (permitted). That means, Alice should not color uv only if one of the subtrees in Figures I(a), I(b) and I(c) would turn up. In the following, we introduce alternative strategies of Alice in dealing with those three cases. We first consider that the subtree in Figures I(a) shows up after Alice has colored uv. Referring to Figures IV(a) and IV(b), you can see 1) if no edge with color 4 is incident with u, Alice can put color 4 on uv instead, and 2) if u in IV(b) is of degree 4 and $a = 2$ or 3, before Alice putting color 1 on uv, F is of type 6 (permitted); 3) if u in IV(b) is of degree 4 and $a = 1$ or 5, by removing the color from any one colored edge, the subtree would become unpermitted. This contradicts that F was generated from a permitted subtree by Bob in his last move. Hence, the two cases shown in Figures IV(a) and IV(b) are the only possibilities of unpermitted F that the subtree in Figures I(a) would appear if Alice colors uv of it.

  Then Alice can adjust her strategies for these two cases by putting colors on other edges as shown in Figures V(a) and V(b), respectively, instead of coloring uv, to make one type 1 and one type 6

(a) $d(w) \geq 3$.

(b) $d(w) \geq 3$.

Figure IV: Subtree in Figure I(a) appears if Alice colored uv with 1.



(a) Coloring the edge with 2 or 3.

(b) Coloring the edge with a.

Figure V: Adjusted strategies of Alice for dealing with subtrees in Figure IV(a) and Figure IV(b). Alice will generate one type 1 and one type 6 subtrees for (a); type 1 and type 3 for (b).

subtrees, and one type 1 and one type 3 subtrees, respectively.

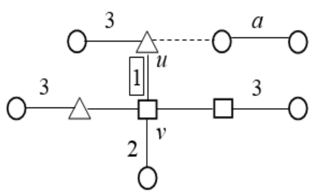
If the subtree in Figure I(b) turns up after Alice has colored uv on F, Figure VI(a) shows the only possible case while Figure VI(b) shows the adjusted strategy of Alice, which generates one subtree of type 1 and the other of types 4 or 7.

Similarly, if the subtree in Figure I(c) turns up after Alice has colored uv on F, Figure VII(a) shows the only possible case while Figure VII(b) shows the adjusted strategy of Alice, which generates one subtree of type 1 and the other type of 4.

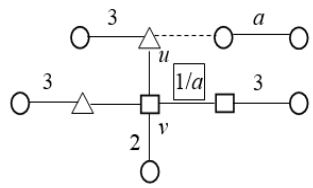
Type 3-8 subtrees

We then introduce Alice's strategy, one by one, for types 3 to 8 in the following. In the rest of the proof, Alice may need to use an appropriate color to generate the desired types of subtrees.

- Type 3 subtrees: all uncolored edges inducing a path P of length at most m, $1 \leq m \leq 3$, with at least two available colors for each uncolored



(a)



(b) Coloring the edge with a or 1 if a is unavailable.

Figure VI: Subtree in Figure I(b) appears if Alice colored uv with 1, and the corresponding adjusted strategy of Alice. Alice will generate one type 1 and one type 4 subtrees if the path represented by the dashed edge is trivial; otherwise, type 1 and type 7.

(a) (b)

Figure VII: Subtree in Figure I(c) appears if Alice colored uv with 1, and the corresponding adjusted strategy of Alice. Alice will generate one type 1 and one type 4 subtrees.

edge if $m > 1$ and one available color for the only uncolored edge if $m = 1$.

Alice may color the edge in the middle if the path induced by uncolored edges is of length 3 or may color any edge if the length of the path is shorter to form completely colored and/or type 3 subtrees. Bob can generate an unpermitted subtree only if P is of length 3 and Bob colors one of its end edges. In this case, Alice may color the middle edge to leave one uncolored edge with at least one available color.

- Type 4 subtrees: all uncolored edges inducing the union of a path, $v_0v_1 \dots v_m$ with $m = 3$ or 4, and a tree at v_m where no colored edges are incident with the vertices of the tree, and there are at least two available colors for $v_0v_1, \dots, v_{m-2}v_{m-1}$ and four for $v_{m-1}v_m$.

  We first note that, in all type 4 subtrees, v_{m-1} is of degree at most 3 and incident with at most one colored edge. Alice may color $v_{m-1}v_m$ with an appropriate color to generate from a type 4 subtree one type 1 and one type 3 subtrees.

  When Bob colors an edge of a subtree of this type, Alice may respond as shown in the following table to generate permitted subtrees.

Bob's act	Alice's response	Generated subtrees
v_0v_1	v_2v_3	Completely colored, Type 1 and Type 3
v_2v_3	v_1v_2	
v_3v_4		One Type 1 and Two Type 3
v_1v_2	$v_{m-1}v_m$	Completely colored, Type 1 and Type 3
Others		One Type 1 and one Type 3 plus one completely colored or one Type 1

- Type 5 subtrees.

  It can be checked that only types 1, 3 and 4 subtrees will be generated if Bob colored any one of the edges incident with s, except us. If Bob colored us, Alice may color vu to generate one completely colored and one type 3 subtrees. If Bob colored other edges, Alice may color us to generate types 1, 3 and/or completely colored subtrees. Moreover, Alice can color us with color 2 on a type 5 subtree to generate one type 1 and one type 3 subtrees.

- Type 6 subtrees.

  Since v is of degree 4, at most one of w and u is of degree 4. Suppose w is a 3-vertex. When vu is colored, a type 6 subtree is turned to one type 1 and one type 3. If Bob colors any edge other than vu, Alice may color vu to generate completely colored, types 1, 2 and/or 3 subtrees.

  Suppose w is a 4-vertex. Then r is of degree at most 3 and u is a 3-vertex. Hence, Alice can color us with 1 to generate one type 5 and one type 1 subtrees and respond to Bob's acts as shown in the following table.

<table>
<tr><th>Bob's act</th><th>Alice's response</th><th>Generated subtrees</th></tr>
<tr><td>us</td><td>wv</td><td rowspan="2">Type 1, Type 2 and Type 3</td></tr>
<tr><td>wv</td><td>us with the same color</td></tr>
<tr><td>vu</td><td>rw with the same color</td><td rowspan="2">Two Type 1 and one Type 3</td></tr>
<tr><td>rw</td><td>vu</td></tr>
<tr><td>Others incident with r</td><td rowspan="2">us with 1 or 2</td><td>Two Type 1 and one Type 4</td></tr>
<tr><td>Others of an uncolored s-branch</td><td>Two Type 1 and one Type 5</td></tr>
<tr><td>Others of an uncolored r-branch</td><td>rw with 2</td><td>Two Type 1 and one Type 4</td></tr>
</table>

- Type 7 subtrees.

  If Bob colors rw or wv, Alice may color vu to generate one Type 1 and one Type 2 plus one completely colored or one Type 3 subtrees, respectively. If Bob colors vu or us, Alice may color wv to make completely colored, types 1, 2 and/or 3 subtrees. In the remaining cases, if color 1 or 2 is available for us, Alice may color us with 1 or 2 to generate one type 6 or one type 8 plus other subtrees of types 1 or 2; if color 1 and 2 are both unavailable for us, Alice may use the move shown in Figure VIII.

- Type 8 subtrees.

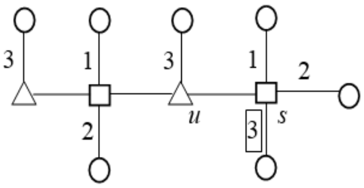


Figure VIII: When color 1 and 2 are both unavailable for us, Alice can color the edge incident with s with 3 to make one type 1 and one type 4 subtrees.

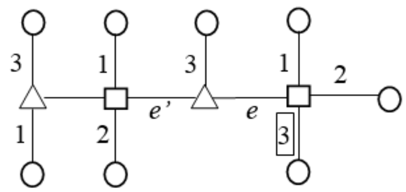


Figure IX: When e is surrounded by edges with colors 1, 2 and 3, Alice can color the edge adjacent to e with 3 to generate one type 1 and one type 3 subtrees.

If Bob colored any edge in his turn such that e is surrounded by edges with colors 1, 2 and 3 as shown in Figure IX, Alice can respond accordingly as shown in the same figure. If Bob colored e, Alice can color e'. In all other cases, Alice may color e with 1 or 2. ∎

References

[1] S. D. Andres, Spieltheoretische Kantenfärbungsprobleme auf Wäldern und verwandte Strukturen (German), Diplomarbeit, Universität zu Köln (2003), 49–50; 55–56; 58; 61; 83–95; 97–123.

[2] S. D. Andres, The game chromatic index of forests of maximum degree $\Delta \geq 5$, Discrete Applied Mathematics, 154 (2006), no. 9, 1317–1323.

[3] H. L. Bodlaender, On the complexity of some colouring games. Graph-theoretic concepts in computer science (Berlin, 1990), 30–40, Lecture Notes in Comput. Sci., 484, Springer, Berlin, 1991.

[4] L. Cai and X. Zhu, Game chromatic index of k-degenerate graphs, J. Graph Theory 36 (2001), 144–155.

[5] W.H. Chan and G. Nong. The game chromatic index of some trees of maximum degree 4. Discrete Appl. Math., 170 (2014), 1-6.

[6] P. Erdős, U. Faigle, W. Hochstättler and W. Kern, Note on the game chromatic index of trees, Theoretical Computer Science 313 (2004), 371–376.