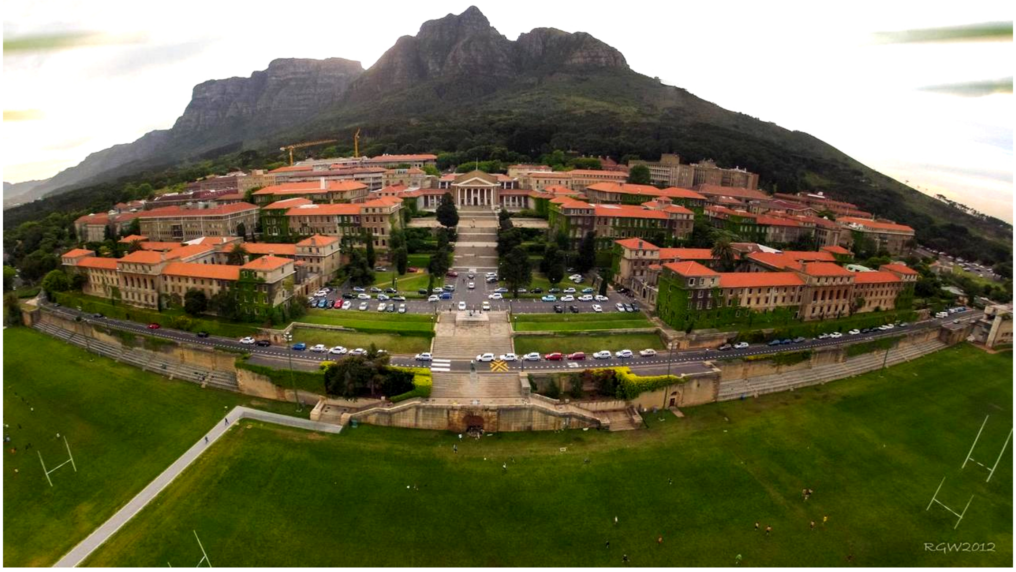
Ecole Normale Supérieure de Rennes

MATHIEU MARI

Game Labelling Number

Supervisor : DAVID ERWIN

Discrete Mathematics and Theoretical Computer Science



University of Cape Town.

Abstract.

This internship report presents a part of my work in the DMTCS Lab. The major part work was to introduce independently the game labelling number because when I had the idea (thanks to David Erwin) to add a competitive aspect at the $L(2,1)$-labelling number, I didn't know that it had been already introduced by Chia, Hsu, Kuo, Liaw and Xu in 2012. In this paper, I present some new results. I have calculated the game labelling number of families of paths, cycles and I give a different proof for complete graphs. I also give some good upper bounds for the families of trees and outerplanar graphs.

May - June 2015

1 Introduction

I did an internship of two months in the *Discrete Mathematics and theoretical Computer Science* laboratory (DMTCS) at the university of Cape Town in South Africa, with Prof David Erwin, a researcher mainly specialized in Graph Theory. This is a field of mathematics that I didn't really know before and I wanted to discover new areas mixing mathematics and computer science. I think that graph theory was a perfect candidate. Since I didn't know this theory I spent the first weeks in discovering the basics of this area. I read among others books the lectures notes for master students which were written by David Erwin. In particaular I was interested in problems linked to graph coloring and graph labelling.

1.1 The chromatic number

The easiest way to color a graph is to color every vertices with a color (red, blue, green, ...) such that two adjacent vertices have not the same color. The first questions are :

- Given a graph and a set of colors, is it possible to color all vertices of the graph with that colors respecting the rule above ?
- Given a graph, what is the minimum number of color one has to choose to color the graph properly ?

The minimum number of color needed to color a given graph G is called his *chromatic number*, and it's denoted by $\chi(G)$. It's very easy to understand this problem, but in some cases, it's very difficult to give an answer. Indeed, to find the chromatic number of a arbitrary graph is an NP-complete problem [4]. Also, it took more than one hundred years to prove the famous *Four colors theorem*, recently proved with the help of computers, which tells us that it's always possible to color an arbitrary planar graph with only four colors.

1.2 The game chromatic number

Imagine now that we add a competitive aspect in graph coloring. Imagine a game in which two players, say Alice and Bob choose a graph G and a set of colors X, and they color turn by turn an uncolored vertex with a color in X such that none of the its colored neighbors is colored with the same color. Alice wins the if every vertices is finally colored, and at the contrary, Bob wins if one of the player is reached in an impasse, i.e. if during the game, there exists an uncolored vertex such that for any colors in X, it has at least one neighbor colored with this color.

Given a graph G, its *game chromatic number* denoted by $\chi_g(G)$ is the minimum number m of colors such that Alice has a winning strategy playing on G with a set of m colors.

The *game chromatic number* of a set of graphs is the highest game chromatic numbers of the graphs of this set.

$$\chi_g(\mathcal{S}) = \max_{G \in \mathcal{S}} \chi_g(G)$$

The game chromatic number was first introduced by Bodlaender in [1]. It's proved in [3] that the game chromatic number of trees is 4. For the family of planar graphs, it is very difficult to find its exact value, and it's not determined yet. The best upper bound that we have is $\chi_g(\mathcal{P}) \leq 17$ where $\mathcal{P}$ is the family of planar graphs [11].

The first part of my work was to study the different strategies to find a good upper bound of the game chromatic number of the family of planar graphs and of its sub-family, the outerplanar graphs.

1.3 The marking game

It's good to introduce an other game called the *marking game*, which was used for the first time in [3] and was formally introduced in [10] to study the game chromatic number.

Suppose $G = (V, E)$ a graph. In this game where Alice plays first, Alice and Bob play turn by turn by *marking* an unmarked vertex. The game ends when all vertices are marked. For each vertex v, let $s(v)$ denote the number of vertices that are marked before v. The *score* of the game is $1+\max_{v\in V} s(v)$. Alice's goal is to minimize the score and at the contraty, Bob try to maximize it.

- The *game coloring number* is the minimum number m such that Alice has a strategy to get a score equal to m. The game coloring number of a graph G is denoted by $col_g(G)$.
- The *game coloring number* of a family of graphs is the highest game chromatic numbers of the graphs of this set.

It's easy to see that $\chi_g(G) \leq col_g(G)$ for any graph G. Indeed, if Alice has a strategy to get a score equal to m, then by coloring the vertices in the same order with a feasible color, it's impossible that a uncolored vertex has more than $m - 1$ colored vertices, and then it's always possible to color this vertex with one of the m colors.

Rem. This game is easier that the chromatic game to study because, we only deal with the order to mark the vertices and not the colors to use. The previous inequality allows us to find upper bounds of the game chromatic number.

The best bound ever found of the game coloring number of planar graph family are $11 \leq \chi_g(\mathcal{P}) \leq 17$ thanks to [11, 9], where $\mathcal{P}$ denotes the family of planar graphs.

Definition. *An* outerplanar graph *is a planar graph in which every vertex belongs to the infinite face.*

Now, the problem of the game coloring number of outerplanar graphs has been settled since Guan and Zhu [6] proved that $col_g(\mathcal{O}) \leq 7$ (where $\mathcal{O}$ denotes the family of outerplanar graphs), and Kierstead and Yang [8] found an outerplanar graph with an game coloring number equal to 7.

Definition. *A* line graph *is a outerplanar graph for which any face (except the infinite face) has at most two inner-edges, and it's denoted by $\mathcal{L}$.*

Rem. I chose the name *line* because its dual graph (after removing the vertex corresponding to the infinite face) is a path. I was studying these graphs because it seemed to be possible to make a planar graph or an outerplanar graph as a special union of some line graphs. And then, it would have been possible to apply an strategy on every different line graphs to get a new upper bound for the planar graphs.

Prop. $\chi_g(\mathcal{L}) \leq 6$

Proof. We use the following result proved by Guan and Zhu in [6].

Lemma. Suppose $G = (V, E)$ is a graph and $E = E_1 \cup E_2$. Let $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$. If $\Delta(G_1) = d$ and $col_g(G_2) \leq k$, then $col_g(G) \leq k + d$.

The strategy for Alice is simple. She applies the strategy that she would use on the sub-graph G_2 that guarantees a score lower than or equal to k. At every move, an unmarked vertex has at most d neighbors already colored in G_1. Since $E = E_1 \cup E_2$, the final score is lower than or equal to $k + d$.

In [6], they show an algorithm to built a spanning sub-tree of any planar graph. Then for an outerplanar graph, the resulting graph G_2 is a tree and $\Delta(G_1) \leq 3$. If we use exactly the same algorithm with a line graph, we easily see that G_1 is an union of paths, and then $\Delta(G_1) \leq 2$. Since the game coloring number of trees is 4 (see [3]), we have $col_g(\mathcal{L}) \leq 6$.

□

2 Game labelling number

Given a graph $G = (V, E)$, we call $L(2,1)$-*labelling* of G a function $f : V(G) \to \mathbb{N}$ such that if two vertices u and v are adjacent then $|f(u) - f(v)| \geq 2$ and if the distance between u and v is 2 then $f(u) \neq f(v)$.

The $L(1,2)$-*labelling number* of G denoted $\lambda(G)$ is the minimum number m such that there exists a $L(1,2)$-labelling f of G, where $f(v) \leq m, \forall v \in V$.

The $L(2,1)$-labelling was first introduced in [5] in which Griggs and Yeh show some properties of this number and calculate it for some families like the families of paths, cycles, tree and completes graphs.

David Erwin told me about this labelling number and advised me to see what we would find if Alice and Bob would play with the rules given by the labelling number instead of the ones given by the chromatic number. Then, taking example on the game chromatic number which is made from the chromatic number, we can also define a game $L(2,1)$-labelling number from the labelling number.

Given a graph $G = (V, E)$ and $m \in \mathbb{N}$, consider the following two-person game which is played on G. Alice (player 1) and Bob (player 2) play turn by turn by assigning a label $l \in \{0, \dots, m\}$ to an unlabelled vertex v such that if v has a neighbor labelled with l' then $|l - l'| \geq 2$ and if there a vertex at distance 2 from v which is labelled with l'' then $l \neq l''$. Alice wins the game if after $|V|$ moves, every vertices are labelled properly and Bob wins if an impasse is reached during the game. The *game* $L(2,1)$-*labelling number* is the minimum number m for which Alice has a winning strategy.

Since the parameters $(2,1)$ are constant in this paper, we will simply call it *game labelling number* and it's denoted by $\lambda_g(G)$.

Rem. I heard after writing this article that this game had been already introduced by Chia, Hsu, Kuo, Liaw and Xu in [2]. They give the more general definition which is the game $L(d,1)$-labelling number, and they calculate this number for the family of complete graphs.

In this paper, I calculate the game labelling number for the family of paths, cycles and complete graphs with a different proof that the one the reader would find in [2]. I also present a upper bound for the family of trees and outerplanar graphs.

2.1 Preliminary

Theorem 1. *For any graph G with maximum degree Δ,*

$$\Delta + 1 \leq \lambda(G) \leq \lambda_g(G) \leq \Delta^2 + 2\Delta$$

We define the *weight* of an unlabelled vertex as the number of labels with which this vertex can't be labelled. One can interpret the game labelling number as the maximum number m for which Bob can find a strategy to create a unlabelled vertex with weight equal to m.

Proof. First, it's easy to see that $\lambda(G) \leq \lambda_g(G)$. Indeed by playing the game, Alice and Bob make a labelling f' and

$$\lambda_g(G) = \max_{v \in V(G)} f'(v) \geq \min_{f:\ \text{labelling}} \left(\max_{v \in V(G)} f(v) = \lambda(G) \right)$$

Let's prove now that $\lambda_g(G) \leq \Delta^2 + 2\Delta$. An unlabelled vertex v has at most Δ neighbors (at distance 1) and there are at most $\Delta(\Delta - 1)$ vertices at distance 2 from v. Every neighbors increase the weight by v by at most 3 and every vertices at distance 2 increase the weight by at most 1. Then the maximum weight of v is $3\Delta + 1\Delta(\Delta - 1) = \Delta^2 + 2\Delta$. Thus, $\lambda_g(G) \leq \Delta^2 + 2\Delta$. □

2.2 Paths and cycles

Griggs and Yeh proved in [5] that the $L(2,1)$-labelling number of the path on n vertices is 4 when $n \geq 5$, and the $L(2,1)$-labelling number of the cycle on n vertices is 4 for any n. We now calculate the game labelling number for the both families.

Let P_n be the path on n vertices and C_n be the cycle on n vertices. If we apply the previous theorem to the family of paths and cycles for which $\Delta = 2$, we obtain the upper bound $\lambda_g(P_n) \leq 8$ and $\lambda_g(C_n) \leq 8$, for any n. Nonetheless, we can find a better upper bound.

Lemma. *For any n, $\lambda_g(P_n) \leq 7$ and $\lambda_g(C_n) \leq 7$.*

Proof. We will prove this result for paths but the proof for cycles is exactly the same.

To facilitate the proof, vertices of P_n are represented by $v_1, v_2, \ldots, v_n$ corresponding to the order naturally induced by the path, and if a vertex v_i is labelled then its label is denoted by c_i. For cycles, Alice has to decide arbitrarily the vertex v_1 and the direction of the cycle. We'll present the strategy for Alice to complete the game such that any unlabelled vertex during the game has weight strictly lower than 8. Note that by Theorem 1, $\lambda_g(P_n) \leq 8$. That means that if a vertex v_i has weight equal to 8 then the neighbors $v_{i-2}, v_{i-1}, v_{i+1}$ and v_{i+2} are all labelled, and their labels respect the following properties :

1. $|c_{i-1} - c_{i+1}| \geq 3$
2. $|c_{i\pm 2} - c_{i\pm 1}| \geq 2$
3. $c_{i-2} \neq c_{i+2}$

Since the labels belong to $[0,7]$, one can check that the only possibles quadruplets are :

$$\begin{aligned}(c_{i-2}, c_{i-1}, c_{i+1}, c_{i+2}) = (6,1,4,7) &\text{ or } (7,1,4,6) \text{ or } (3,1,5,7) \text{ or } (7,1,5,3)\\ \text{or } (3,1,6,4) &\text{ or } (4,1,6,3) \text{ or } (0,2,5,7) \text{ or } (7,2,5,0)\\ \text{or } (0,2,6,4) &\text{ or } (4,2,6,0) \text{ or } (0,3,6,1) \text{ or } (1,3,6,0)\end{aligned}$$

We will designate by *wrong couple*, a couple cc' of labels such that if the vertices v_i and v_{i+1} are respectively labelled by c and c', then the weight of v_{i-1} and v_{i+2} will always be strictly lower than 8 whatever the situation of the game. By seeing the above list of quadruplets, we can make the list of wrong couples :

$$\{04, 07, 15, 25, 26, 37, 40, 52, 51, 62, 70, 73\}$$

Note that if the couple ab is wrong couple then ba is a also a wrong couple.

Observation 1. *For any label c, there exists at least* one *label c' such that cc' (or $c'c$) is a wrong couple.*

Note also that for these couples, the two vertices don't have the same role. Indeed, one is the direct neighbor and the other is the vertex at distance 2. Then we call *wrong directed couple* the couple cc' such that if the vertices v_i and v_{i+1} are respectively labelled by c and c', then the weight of v_{i+2} will always be strictly lower than 8, whatever the situation of the game. There's the list of wrong directed couples :

$$\{04, 07|14, 15, 17|20, 24, 25, 26, 27|30, 37|40, 47|50, 51, 52, 53, 57|60, 62, 63|70, 73\}$$

Note that if cc' is a wrong directed, then $c'c$ is not necessarily a wrong directed couple (example : 47). Plus, if $c'c$ is a wrong directed couple as well then this is a wrong couple.

Observation 1'. *For any label c, there exists at least* two *labels c' such that cc' is a* wrong directed couple.

We now describe the strategy for Alice such the two following observations are true at any time during the game.

Observation 2. *After any Alice's move, every labelled vertex has at least one neighbor labelled, except one of the two end-vertices which can have no labelled neighbors.*

Observation 3. *Suppose that v_i and v_{i+1} are labelled.*

1. *If the three vertices v_{i+2}, v_{i+3} and v_{i+4} (resp. v_{i-1}, v_{i-2} and v_{i-3}) are unlabelled then $c_i c_{i+1}$ (resp. $c_{i+1} c_i$) is a wrong directed couple.*
2. *If the two vertices v_{i+2}, v_{i+3} (resp. v_{i-1}, v_{i-2}) are unlabelled and v_{i+4} (resp. v_{i-3}) is labelled then it's not possible to label v_{i+3} (resp. v_{i-2}) such that v_{i+2} (resp. v_{i-1}) get a weight equal to 8.*

Alice plays her first move by labelling v_1 with label 0. Observation 2 and observation 3 are true after this first move.

Now, suppose that after several steps, Bob labels the vertex v_i with label c_i. We also suppose that observation 2 and observation 3 are true just before Bob's move.

Case 1 : Two neighbors unlabelled. Suppose that v_{i-1} and v_{i+1} are unlabelled.

1.1 : v_{i-2} and v_{i+2} are labelled. In this case, because of observation 2 and observation 3.1, $c_{i-3}c_{i-2}$ and $c_{i+3}c_{i+2}$ are wrong directed couples, then $v_{i\pm1}$ can't pretend to get weight equal to 8. Thus, Alice can pick up v_{i-1} with one of the 8 labels.

1.2 : v_{i-2} and v_{i+3} are labelled and v_{i+2} is unlabelled. Alice can pick up v_{i+1} with one of the 8 labels. Because of observation 2 and observation 3.1, $c_{i-3}c_{i-2}$ and $c_{i+4}c_{i+3}$ are wrong directed couples, then v_{i-1} and v_{i+2} can't pretend to get weight equal to 8.

1.2' : v_{i-3} and v_{i+2} are labelled and v_{i-2} is unlabelled. Same case than 1.2 because of the symmetry of the path.

1.3 : $v_{i\pm2}$ are unlabelled and $v_{i\pm3}$ are labelled. $c_{i+4}c_{i+3}$ is a wrong oriented couple by hypothesis, then whatever the label Alice chooses to label v_{i+1}, the weight of v_{i+2} will be strictly lower than 8. If there exist at least two labels (say a and b) such that ac_i and bc_i are a wrong directed couple then Alice can choose $c_{i+1} = a$ or $c_{i+1} = b$ depending on whether c_{i+3} is equal to a or b. The only labels c for which there exist an unique c' such that $c'c$ is a wrong directed couple are 1 and 6 (such that $c' = 5$ and $c' = 2$ respectively). These two cases are symmetric actually because of the symmetry of labels ($6 = 7 - 1$), then we are going to study only the case where $c_i = 1$ and $c_{i+3} = 5$.

In this case it's not possible to label v_{i+1} with label 5, and then Alice's going to create a "well directed couple". If $c_{i-3} \neq 5$ then Alice labels v_{i+1} with label c_{i-3} and otherwise she can choose any feasible label to label v_{i+1}. One can check that v_{i-1} can't get a weight strictly greater than 7.

Finally, the vertex v_{i-2} is not dangerous for Alice because $c_{i-4}c_{i-3}$ is a wrong directed couple by assumption.

1.4 : v_{i+1}, v_{i+2} and v_{i+3} are unlabelled. Because of observation it's possible to choose a label c_{i+1} for v_{i+1} such that $c_i c_{i+1}$ is a wrong couple.

1.4 : v_{i-1}, v_{i-2} and v_{i-3} are unlabelled. Same case than 1.4 because of the symmetry of the path.

Case 2 : One neighbor labelled. Because of the symmetry, we will only consider the case where v_{i-1} is labelled and v_{i+1} is unlabelled.

2.1 : v_{i+2} **is labelled.** Observation 3.2 permits to conclude that the weight of v_{i+1} doesn't exceed 7. Thus, Alice can choose one of the eight labels to label this vertex.

2.2 : v_{i+2} **is unlabelled and** v_{i+3} **is labelled.** While observation 3.1 is true just before Bob's move, $c_{i+4}c_{i+3}$ is a wrong directed couple, then Alice can label v_{i+1} with any feasible label.

2.3 : v_{i+2} **and** v_{i+3} **are unlabelled.** Because of observation 1', Alice can find a label c_{i+1} such that $c_i c_{i+1}$ is a wrong directed couple.

Case 3 : Both neighbors are labelled. Since observation 2 is true just before Bob's move, $v_{i\pm 2}$ are both labelled and then Bob's move has changed nothing for the rest of the game. Thus Alice can pick up any vertex, choosing by example to label the smaller unlabelled vertex, and she decides the label by seeing case 2.

We can see that in any case, observations 2 and 3 are still true after Alice's move. Thus, for any n, $\lambda_g(P_n) \leq 7$.

□

Theorem 2. *Let P_n be the path on n vertices.*

1. $\lambda_g(P_2) = 2$. $\lambda_g(P_3) = 3$. $\lambda_g(P_4) = 6$. $\lambda_g(P_5) = 6$.
2. $\lambda_g(P_n) = 7, \forall n \geq 6$.

Proof. 1. For $n \leq 5$, it's easy to calculate the corresponding game labelling number by programming the min-max algorithm. I wrote a program in OCaml (the reader could find a simplified version in annex) to calculate the game labelling number for small values of n ($n \leq 6$). Alice's first move is always to label v_2 with label 0.

2. By using the previous lemma, we can already say that $\lambda_g(P_n) \leq 7$ for all $n \geq 6$. By programming the algorithm of min-max (annex) we can see that with maximum label equal to 6, Bob has a winning strategy for P_6, and since P_6 is a subpath of P_n for all $n \geq 6$, Bob has also a winning strategy on P_n with maximum label equal to 6. Thus $\lambda_g(P_n) = 7$ for any $n \geq 6$.

□

Rem. Note that for the family of paths, the game labelling number is strictly greater than the labelling number since Griggs and Yeh [5] have proved that $\lambda(P_n) = 4$ for $n \geq 5$.

Theorem 3. *Let C_n be the cycle on n vertices.*

1. $\lambda_g(C_3) = 4$.
2. $\lambda_g(C_n) = 7, \forall n \geq 4$.

Proof. 1. We have $\lambda_g(C_3) \geq \lambda(C_3) = 4$. If Alice picks a vertex up with the label 2 then, because of the symmetry of the graph and the symmetry of the labels (we are only interested in the distance between labels), any Bob's move is actually the same, then let imagine for example that he labels one of the two unlabelled vertices with 0, then Alice can label the last vertex with label 4. Thus Alice has a winning strategy with maximum label equal to 4, so $\lambda_g(C_3) \leq 4$.

2. By the lemma we know that $\lambda_g(C_n) \leq 7$ for all $n \geq 4$. By programming the min-max algorithm (annex) we can see that Bob has a winning strategy with 7 labels for $C_n, 4 \leq n \leq 6$, then $\lambda_g(C_n) = 7$ for $4 \leq n \leq 6$. Finally, P_6 is a subgraph of C_n for $n \geq 7$ then $\lambda_g(C_n) \geq \lambda_g(P_6) = 7$. Thus, for any $n \geq 4$, $\lambda_g(C_n) = 7$.

□

2.3 Trees and forests

We now look at the game labelling number of the family of trees with maximal degree Δ. If Δ is equal to 1, this kind of tree are simply paths and then refer to previous section.

Theorem 4. *Let T a tree with maximal degree $\Delta \geq 2$. Then $\lambda_g(T) \leq \Delta + 10$.*

Proof. We will give a strategy for Alice such that at every step of the game, any unlabelled vertex has a weight at most equal to $\Delta + 10$.

For her first move, Alice chooses and label an arbitrary vertex v_0, and creates a set of vertices T' with initially, $T' := \{v_0\}$. Now, suppose that Bob has just moved by labelling vertex v. Then,

1. If $v \in T'$ then, Alice labels an arbitrary unlabelled vertex in T', and if every vertices of T' are labelled, then Alice labels an vertex w adjacent to T', and updates $T' := T' \cup \{w\}$.
2. Otherwise, let P be the unique path between v_0 and v. And let v_1 be the last vertex P has in common with T'.
   (a) If v_1 is unlabelled then Alice moves by labelling this vertex with a feasible label.
   (b) Otherwise, let v_2 be the adjacent vertex of v_1 in P (different that v). Then,
      i. If v_2 is unlabelled, then Alice labels this vertex with a feasible label.
      ii. Else, if v_2 is labelled then Alice labels an arbitrary unlabelled vertex in T', and if all vertices of T' are labelled, then Alice labels a vertex adjacent to T', and updates $T' := T' \cup \{w\}$.

   Alice updates $T' := T' \cup P$.

After any Bob's moves, an unlabelled vertex v has :

- A degree at most 3 in T' (2. (a)) and since every labelled vertices in T are in T', v has at most 3 adjacent vertices labelled. The vertices at distance 1 contribute to increase the weight of an unlabelled vertex at most by $3 * 3 = 9$.
- At most 3 "grandchildren" (i.e vertices w such that $d(v, w) = 2$ and $d(v_0, w) > d(v_0, v)$) in T' and then at most 3 labelled "grand-sons". If v has exactly 3 labelled "grandchildren" then for the next move Alice labels v (2. (b) i.) and then when it's just been labelled, v has at most 2 adjacent vertices labelled. Since, the vertices at distance 2 add 1 of the weight, the weight is increased at most by $\max(3 * 3 + 2 * 1, 3 * 2 + 3 * 1) = 11$.
- At most 1 "grandfather" (i.e. a vertex w such that $d(w, v) = 2$ and $d(v_0, w) < d(v_0, v)$) and $\Delta - 2$ "brothers" (i.e. vertices w such that $d(w, v) = 2$ and $d(v_0, w) = d(v_0, v)$). Then, all of them increase the weight at most by $\Delta - 1$.

The weight of v is finally increased at most by $11 + (\Delta - 1) = \Delta + 10$. Thus $\lambda_g(T) \leq \Delta + 10$. □

Rem. This strategy is inspired by the proof used by Faigle, Kern, Kierstead, and Trotter [3] to prove that the game chromatic number of the family of trees is lower than 4.

Rem. For some graphs, Bob can find a strategy such that if Alice respects the strategy described above, then they have to choose a maximal label equal to $\Delta + 10$. That means that $\lambda_g \leq \Delta + 10$ is the tightest bound Alice can hope with this strategy.

Corollary. *Let F be a forest with maximal degree Δ, then $\lambda_g(F) \leq \Delta + 10$.*

Proof. If Bob plays first, Alice can still apply her strategy. Since Alice first move in the algorithm described above is chosen arbitrarily, then Alice has just to consider that she played first where Bob has just moved, and picks up a adjacent vertex of the first one. After the two first moves, Alice applies the same strategy on every trees. □

2.4 Outerplanar graphs

The strategy used by Alice on outerplanar graphs is almost the same than the one used for trees. Indeed, we first show how to make a particular spanning tree and then we use the strategy for trees on this spanning tree. We finally try to find the tightest bound of the game labelling number by calculating the maximal weight possible for an arbitrary vertex.

The following lemma explains that we can color the edge of an connected outerplanar graph with two colors (Blue and Red, for example) such that the "blue-subgraph" is a spanning tree and the "red-subgraph" has maximal degree lower than or equal to 3.

Lemma. *Let $O = (V, E)$ be an outerplanar graph. Then there exist two graphs $G_B = (V, E_B)$ and $G_R = (V, E_R)$ such that :*

- *For any $e \in E$, $e \in E_B$ or $e \in E_R$.*
- *G_B is a tree.*
- *For any $v \in V$ there exist $w \in V$ such that $vw \in E_B$ (E_B is a spanning tree)*
- *$\Delta(G_R) \leq 3$*

Rem. This structure is used on by Guan and Zhu in [7] to calculate the game chromatic number of outerplanar graphs.

We explain here how to color edges in O such that we get two sub-graphs respecting the properties above. We will do that by choosing an order on vertices.

Let v_1 and v_2 be two arbitrary vertices in O such that v_1 is adjacent to v_2 and v_1v_2 belongs to the infinite face. Then pick up this edge with color *blue*, i.e. $E_B := \{v_1v_2\}$. We also initiate $E_R := \emptyset$.

Suppose now that we have already colored the edges of the subgraph generated by vertices $v_1, \ldots, v_k$, for $k \geq 2$ (i.e. $< v_1, \ldots, v_k >= E_B \cup E_R$). Let v_i ($i \leq k$) be a vertex with at least one adjacent vertex w which is not in $\{v_1, \ldots, v_k\}$.

- If v_i has an adjacent vertex $v_j \in \{v_1, \ldots, v_k\}$ such that v_i,v_j and w belong to the same inner face in O *and* v_j and w are adjacent in O then color the edge v_iw with color *blue* (i.e. $E_B := E_B \cup \{v_iw\}$) (resp. *red*) and the edge v_jw with color *red* (resp. *blue*) if $i < j$ (resp. $j < i$). Finally, let v_{k+1} be the vertex w.
- Otherwise, color the edge v_iw with color *blue* and create $v_{k+1} := w$.

At the end, we get two sub-graphs G_B and G_R respecting the properties of the lemma. For more details, the reader can find the proof of that result in [7].

Rem. This algorithm also gives us an order on vertices. We will say that $v < w$ if v was in $G_B \cup G_R$ before w.

Theorem 5. *Let O be an outerplanar graph with maximal degree $\Delta \geq 6$. Then, $\lambda_g(O) \leq 20 + 4\Delta$.*

Proof. In the proof, v is called a *blue-neighbor* of w if $vw \in G_B$ and a *red-neighbor* of w is $vw \in G_R$.

Alice has to use the strategy described for trees on the spanning tree (the blue subgraph G_B) given by the previous lemma. She considers that the root is the vertex v_1 used in the proof of the lemma. Note that v_1 can be chosen arbitrarily, and then we get the same result if Bob begins, because it's always possible to choose v_2 such that v_1v_2 belongs to the infinite face.

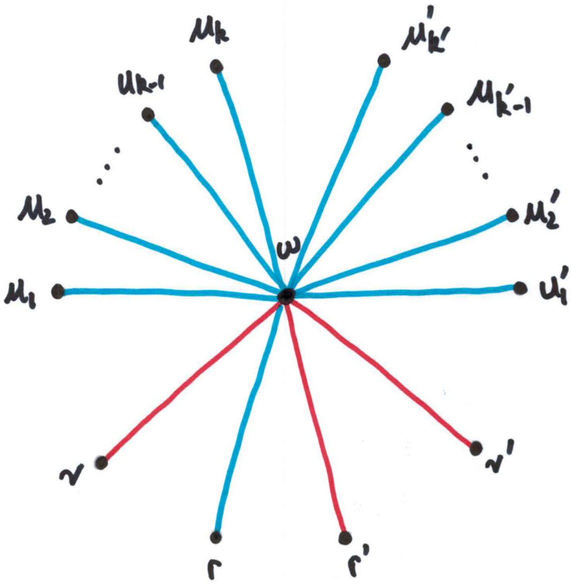


Figure 1: An arbitrary vertex w

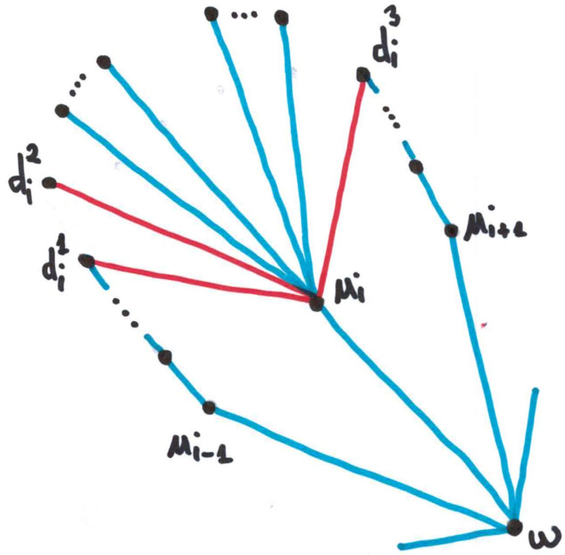


Figure 2: An arbitrary blue-neighbor u_i

Let w be a arbitrary vertex. The picture on the left shows an arbitrary vertex with exactly three red-neighbors. In that picture r,$r' < w$, and then in the process, w was created from r and r'. All others vertices on the diagram were created after w.

We now count the contribution of the different neighbors and vertices at distance 2 in the total weight of w :

+9 : If Alice applies the strategy used for trees, w has at most 3 labelled neighbors in G_B and all of them increase the weight by $3 \cdot 3 = 9$.

+9 : According to the lemma, w has at most 3 labelled neighbors in G_R and all of them increase the weight by $3 \cdot 3 = 9$.

$+3 \cdot (\Delta - 1)$: Each of the three red-neighbors (v,r' and v') can have $\Delta - 1$ neighbors different than w, these vertices at distance 2 from w contribute to increase its weight by $3 \cdot (\Delta - 1)$.

$+\Delta - 1$: In the algorithm for trees, it's not possible to limit the number of labelled neighbors of the "father" of an arbitrary vertex, and then r can have in the worst case $\Delta - 1$ labelled neighbors (we count the red-neighbors and the blue-neighbors at the same time) when w has just been labelled.

+2 : According to the algorithm for trees, there are at most two blue-neighbors of a labelled blue-neighbor u_i.

+4 : The most difficult is to count the weight brought by red-neighbors of blue-neighbors ($\{u_i, 1 \leq i \leq k\}$ and $\{u_i, 1 \leq i \leq k\}$) of w.

The figure on the right shows the diagram of a arbitrary blue-neighbor u_i of w, for $i \in \{1, \dots, k\}$. u_i has at most three red-neighbors d_i^1, d_i^2 and d_i^3. It's easy to see that d_i^1 and d_i^2 belong to a path from u_{i-1} which doesn't contain w, and d_i^3 belongs to a path from u_{i+1} which doesn't contain w neither.

Suppose u_i labelled, and $i \geq 2$. If Bob labels d_1^i, d_i^2 or d_i^3, then just after this move, Alice labels w according to the algorithm for trees. But the vertex labelled just before w is a blue-neighbor of w different than u_i. Then a blue-neighbor u_j or u'_j can have some red-neighbors only if $j = 1$. In this case, there are at most 4 vertices possibly labelled which are d_1^1, d_1^2, $d_1'^1$ and $d_1'^2$ where $d_i'^1$ and $d_i'^2$ are made by symmetry from the right diagram.

In final, the weight of w is at most $9+9+3\cdot(\Delta-1)+\Delta-1+2+4 = 4\Delta+20$. Thus, $\lambda_g(O) \leq 20+4\Delta$. $\square$

2.5 Complete graphs

It's easy to see that the labelling number of the complete graph on n vertices K_n is $2(n-1)$, by labelling the vertices with even labels 0,2,...,$2n-2$. Then a strategy for Bob to increase the number of labels would be to label a vertex with an odd label. The following theorem gives the exacts game labelling number depending on the rest of division of n by 3.

Theorem 6. *Let K_n be the complete graph with n vertices.*

- *If $n = 3m$ then $\lambda_g(K_n) = 7m - 3$*
- *If $n = 3m + 1$ then $\lambda_g(K_n) = 7m$*
- *If $n = 3m + 2$ then $\lambda_g(K_n) = 7m + 2$*

To prove this result, we introduce a new game which could be considered as the "dual game" of the labelling game on complete graphs.

In a complete graph, any vertex is at distance 1 from any other vertex, then during the labelling game, a label c is used only one time and the labels $c \pm 1$ can't be used to label an unlabelled vertex.

In the "dual game" we consider the labels $[0, \ldots, k-1]$ as a path with k vertices where two consecutive labels are represented in the path by two adjacent vertices. Bob or Alice plays by *marking* turn by turn an unmarked vertex and its two neighbors (the neighbors can be unmarked or already marked). The game ends when all vertices are marked. The *score* is the number of moves played to finish the game. Alice's goal is to maximise the score and at the contrary Bob wants to minimise it.

- We call $\sigma_A(k)$ the maximum number n such that there exists a strategy for Alice to get a score equal to n playing in a path with k vertices when she plays first.
- We call $\sigma_B(k)$ the minimum number n such that there exists a strategy for Bob to get a score equal to n playing in a path with k vertices when he plays first.

If there exists such a strategy for Alice to get a score equal to n in a path with k vertices, then by labelling the vertices of K_n corresponding to the vertices of the path, Alice has a winning strategy with k labels playing on the complete graph $K_{\sigma_A(k)}$, and then $\lambda_g(K_n) \leq k-1$.

Reciprocally, if Alice has a winning strategy on K_n with k labels, by applying the same strategy, she can get a score greater than or equal to n if she plays on a path with k vertices. Thus,

$$\lambda_g(K_n) = \min\{k, \sigma_A(k) = n\} - 1 \tag{1}$$

Rem. We could say that σ_A is a kind of inverse function of $n \to \lambda_g(K_n)$.

Lemma. *let δ_A and δ_B be the functions from $\{0, \ldots, 6\}$ to $\mathbb{N}$ given by :*

r	0	1	2	3	4	5	6
$\delta_A(r)$	0	1	1	2	2	3	3
$\delta_B(r)$	0	1	1	1	2	2	3

If $k \geq 0$ and $k = 7d + r$ then $\sigma_A(m) = 3d + \delta_A(r)$ and $\sigma_B(k) = 3d + \delta_B(r)$

Proof. One easily agree that the families $\{\sigma_A(k)\}_{k\geq 0}$ and $\{\sigma_B(k)\}_{k\geq 0}$ respect the following properties, for all $k \geq 0$:

$$\sigma_A(k) \geq \sigma_B(k) \tag{2}$$

$$\sigma_A(k+1) \geq \sigma_A(k) \text{ and } \sigma_B(k+1) \geq \sigma_B(k) \tag{3}$$

At any step, every unmarked vertex v belongs to a maximal connected component of unmarked vertices. If the order of this component is k, it will be called a *k-component*. Let $v_1, \ldots, v_k$ be the ordered vertices of a k-component. Suppose that Bob (resp. Alice) marks v_i (v_{i-1} and v_{i+1} are also marked just after this move).

If $i = 1$ or $i = k$: Bob (resp. Alice) has marked 2 vertices (v_1 and v_2 or v_{k-1} and v_k) of the k-component then Alice (resp. Bob) can play her next move in a $(k-2)$-component. In any case, we have, for any $k \geq 2$:

$$\begin{aligned}\sigma_A(k) &\geq 1 + \min\big(\sigma_A(k-2), \sigma_B(k-2)\big) \\ &\geq 1 + \sigma_B(k-2) && \text{according to (2)}\end{aligned}$$

and,

$$\begin{aligned}\sigma_B(k) &\leq 1 + \max\big(\sigma_A(k-2), \sigma_B(k-2)\big) \\ &\leq 1 + \sigma_A(k-2) && \text{again according to (2)}\end{aligned}$$

If $2 \leq i \leq k-2$: Bob (resp. Alice) has marked 3 vertices in the middle of the k-component, and then has created a $(i-2)$-component (vertices $v_1, \ldots, v_{i-2}$) and a $(k-i-1)$-component (vertices $v_{i+2}, \ldots, v_k$). We note that one of these two components can possibly have order 0. Thus, for the next move, Alice (resp. Bob) can choose to play either in the $(i-2)$-component or in the $(k-i-1)$-component or in any other component, depending on the best strategy to apply. Then by denoting $p := i-1$ and $q := k-i-1$ such that $p, q \geq 0$ and $p+q+3 = k$, we have in the worst case for Bob (resp. Alice) :

$$\begin{aligned}\sigma_A(k) &\geq 1 + \min\big(\sigma_A(p), \sigma_B(p)\big) + \min(\sigma_A(q), \sigma_B(q)\big) \\ &\geq 1 + \sigma_B(p) + \sigma_B(q)\end{aligned}$$

and,

$$\begin{aligned}\sigma_B(k) &\geq 1 + \max\big(\sigma_A(p), \sigma_B(p)\big) + \max(\sigma_A(q), \sigma_B(q)\big) \\ &\geq 1 + \sigma_A(p) + \sigma_A(q)\end{aligned}$$

In particular, since $\sigma_B(1) = 1$ and $\sigma_A(0) = 0$, we have :

$$\sigma_A(k) \geq 1 + \sigma_B(1) + \sigma_B(k-3-1) = 2 + \sigma_B(k-4) \tag{4}$$

and,

$$\sigma_B(k) \leq 1 + \sigma_A(0) + \sigma_B(k-3-0) = 1 + \sigma_A(k-3) \tag{5}$$

Bob (resp. Alice) can always play in one of the two components (p or q) but Alice (resp. Bob) chooses in which component Bob is going to play after her move according to her best strategy. Then, we have :

$$\sigma_A(k) \leq 1 + \max\left(\sigma_B(k-2), \max_{p+q+3=k}\left(\min\big(\sigma_A(p) + \sigma_B(p), \sigma_A(q) + \sigma_B(p)\big)\right)\right) \tag{6}$$

and,

$$\sigma_B(k) \geq 1 + \min\left(\sigma_A(k-2), \min_{p+q+3=k}\left(\max\big(\sigma_A(p) + \sigma_B(p), \sigma_A(q) + \sigma_B(p)\big)\right)\right) \tag{7}$$

Now, we prove the lemma by induction on d.

For $d = 0$, it's easy to calculate manually the winning strategy for Alice, and the winning strategy for Bob, and then one can see that for r in $\{1, \ldots, 6\}$, $\sigma_A(r) = \delta_A(r)$ and $\sigma_B(r) = \delta_B(r)$.

Let $K = 7D + R \in \mathbb{N}$, with $R \in \{0, \ldots, 6\}$. Suppose that for any $k < K$, $\sigma_A(k) = 3d + \delta_A(r)$ and $\sigma_B(k) = 3d + \delta_B(r)$ where $k = 7d + r$, with $r \in \{0, \ldots, 6\}$.

Note that for all $7 < k < K$, we have $\sigma_A(k-7) + 3 = \sigma_A(k)$ and $\sigma_B(k-7) + 3 = \sigma_B(k)$. Then, if $7d + r < 7D$ (we don't say that r has to belong to $\{0, \ldots, 6\}$), we have (using induction assumptions)

$$\sigma_A(7d + r) = 3(d + \phi(r)) + \delta_A(r \bmod 7) \tag{8}$$

and,

$$\sigma_B(7d + r) = 3(d + \phi(r)) + \delta_B(r \bmod 7) \tag{9}$$

where $\phi(r) = l$ when $7l \leq r < 7(l + 1)$. This result will simplify calculations later.

Using (4) and (9) we have :

$$\begin{aligned}
\sigma_A(K) &\geq 2 + \sigma_B(K - 4) \\
&= 2 + \sigma_B\big(7D + (R - 4)\big) \\
&= 2 + 3D + \delta_B\big((R - 4) \bmod 7\big) + 3 \cdot \phi(R - 4)
\end{aligned}$$

We easily check that $2 + \delta_B\big((R - 4) \bmod 7\big) + 3 \cdot \phi(R - 4) = \delta_A(R)$, for all $R \in \{0, \ldots, 6\}$. Thus, we have just proved that

$$\sigma_A(K) \geq 3D + \delta_A(R) \tag{10}$$

Using (5) and (8) we have :

$$\begin{aligned}
\sigma_B(K) &\leq 1 + \sigma_A(K - 3) \\
&= 1 + \sigma_A\big(7D + (R - 3)\big) \\
&= 1 + 3D + \delta_A\big((R - 3) \bmod 7\big) + 3 \cdot \phi(R - 3)
\end{aligned}$$

We easily check that

$$1 + \delta_A\big((R - 3) \bmod 7\big) + 3\phi(R - 3) = \delta_B(R), \text{ for all } R \in \{0, \ldots, 6\}$$

Thus, we have just proved that

$$\sigma_B(K) \leq 3D + \delta_B(R) \tag{11}$$

For $0 \leq p \leq 7D + R - 3$, where $p = 7d + r$ with $r \in \{0, \ldots, 6\}$, and $q = (7D + R) - p - 3$, we have :

$$\begin{aligned}
\sigma_A(p) + \sigma_B(q) &= \sigma_A(p) + \sigma_B\big((7D + R) - p - 3\big) \\
&= \sigma_A(7d + r) + \sigma_B\big(7(D - d) + (R - r - 3)\big) \\
&= 3 \cdot \big(d + \phi(r)\big) + \delta_A(r \bmod 7) \\
&\quad + 3 \cdot \big((D - d) + \phi(R - r - 3)\big) + \delta_B\big((R - r - 3) \bmod 7\big) \qquad \text{using (8) and (9)} \\
&= 3D + \delta_A(r) + 3 \cdot \phi(R - r - 3) + \delta_B\big((R - r - 3) \bmod 7\big)
\end{aligned}$$

Let $f(R, r) := \delta_A(r) + 3 \cdot \phi(R - r - 3) + \delta_B\big((R - r - 3) \bmod 7\big)$, depending only on the parameters R and r. Then, we can rewrite inequalities (6) and (7) :

$$\sigma_A(K) \leq 1 + \max\left(\sigma_B(K - 2), 3D + \max_{r \in \{0, \ldots, 6\}}\left(\min\Big(f(R, r), f\big(R, (K - r - 3) \bmod 7\big)\Big)\right)\right) \tag{12}$$

and,

$$\sigma_B(k) \geq 1 + \min\left(\sigma_A(k - 2), 3D + \min_{r \in \{0, \ldots, 6\}}\left(\max\Big(f(R, r), f\big(R, (K - r - 3) \bmod 7\big)\Big)\right)\right) \tag{13}$$

Since $1 + \sigma_A(K-2) \geq 3D + \delta_B(R)$ and $1 + \sigma_B(K-2) \leq 3D + \delta_A(K)$, to prove that $\sigma_A(K) \leq 3D + \delta_A(R)$ and $\sigma_B(K) \geq 3D + \delta_B(R)$, we have just have to check that

$$\delta_A(R) \leq \max_{r \in \{0,\dots,6\}} \min\Big(1 + f(R,r), 1 + f\big(R, (K-r-3) \bmod 7\big)\Big), \forall R \in \{0,\dots,6\} \tag{14}$$

and,

$$\delta_B(R) \leq \min_{r \in \{0,\dots,6\}} \max\Big(1 + f(R,r), 1 + f\big(R, (K-r-3) \bmod 7\big)\Big), \forall R \in \{0,\dots,6\} \tag{15}$$

The following tabular shows the calculation of couples $\Big(1 + f(R,r), 1 + f\big(R, (K-r-3) \bmod 7\big)\Big)$

R/r	0	1	2	3	4	5	6	min(max(.,.))	max(min(.,.))
0	$(0,0)$	$(1,0)$	$(0,0)$	$(0,1)$	$(0,0)$	$(0,1)$	$(1,0)$	0	0
1	$(1,0)$	$(1,1)$	$(1,0)$	$(0,1)$	$(1,1)$	$(1,2)$	$(1,1)$	1	1
2	$(1,1)$	$(2,1)$	$(1,1)$	$(1,1)$	$(1,1)$	$(1,2)$	$(1,1)$	1	1
3	$(1,1)$	$(2,2)$	$(2,1)$	$(1,2)$	$(2,1)$	$(1,2)$	$(2,2)$	1	2
4	$(2,2)$	$(2,2)$	$(2,2)$	$(2,2)$	$(2,2)$	$(2,2)$	$(2,2)$	2	2
5	$(2,2)$	$(3,3)$	$(2,2)$	$(2,3)$	$(3,2)$	$(2,3)$	$(3,2)$	2	3
6	$(3,2)$	$(3,3)$	$(3,3)$	$(2,3)$	$(3,3)$	$(3,3)$	$(3,3)$	3	3

The tabular permits to conclude that the formulas (14) and (15) are true. Thus, we proved that $\sigma_A(7D+R) \geq 3D + \delta_A(R)$ and $\sigma_A(7D+R) \geq 3D + \delta_A(R)$, for all $R \in \{0,\dots,6\}$. With (10) and (11), we have finally proved by induction the lemma.

$\square$

Now, it's easy to prove the theorem 6.

Proof. (Theorem 6)
According to 1, we have :

$$\lambda_g(K_n) = \min\{k, \sigma_A(k) = n\} - 1$$

If $n = 3m$. Using the lemma, we find that :

$$\begin{aligned}\lambda_g(K_n) &= \min\{k, \sigma_A(k) = 3m\} - 1 \\ &= \min\{7(m-1)+5, 7(m-1)+6, 7m\} - 1 \\ &= 7m - 3\end{aligned}$$

If $n = 3m+1$. Using the lemma, we find that :

$$\begin{aligned}\lambda_g(K_n) &= \min\{k, \sigma_A(k) = 3m+1\} - 1 \\ &= \min\{7m+1, 7m+2\} - 1 \\ &= 7m\end{aligned}$$

If $n = 3m+2$. Using the lemma, we find that :

$$\begin{aligned}\lambda_g(K_n) &= \min\{k, \sigma_A(k) = 3m+2\} - 1 \\ &= \min\{7m+3, 7m+4\} - 1 \\ &= 7m+2\end{aligned}$$

$\square$

3 Annex : Calculation of the game labelling number of small paths.

```
let rec init_pos n = (* make a list [n,n-1,...,0] *)
   match n with 0 -> [0]
   |n -> n::(init_pos (n-1));;

let snd (a,b) = b ;; let fst (a,b) = a ;;

let init_path m n = let l = init_pos n in Array.make m (-1,l);;
(* make an array [|(-1,l); ... ; (-1,l)|] where l = [n,n-1,...,0] |] *)
(* this array represent the state of the game. -1 means that the corresponding vertex is unlabelled,
and the list l represents the feasible labels for this vertex *)

let rec remove x l = match l with (* remove element x from the list l *)
   |[] -> []
   |e::r -> if x=e then r else e::(remove x r);;

let remove' x b = let (e,l) = b in (e,remove x l);;

let solution_path m n = (* m : path order, n : maximal label*)
   (* the result of this program is :
      - 0 if Alice has a winning strategy playing on a path with m vertices with labels lower
            than or equal to n.
      - 1 otherwise. *)
   let choice v = (* make a list of all possible moves deducted from the state of the game *)
      let rec ins x l rest = match l with (* [e1,e2,...] -> [(x,e1),(x,e2),...]@rest *)
         |[]-> rest
         |e::r -> (x,e)::(ins x r rest)
      in
      let rec aux i =
         if i = m then []
         else
         (
            if fst(v.(i)) = -1 then (ins i (snd(v.(i))) (aux (i+1))) else aux (i+1)
         )
      in aux 0
   in

   let rec aux w c l =
(* w : state of the game, c : number of moves already payed, l : list of feasible moves *)
      if c = m then 0 (* all vertices are labelled, Alice wins *)
      else (* some vertices are still unlabelled *)
      (
      match l with
         |[] -> 1-c (* there's no feasible moves anymore, ... *)
         |(i,color)::r ->
(* the player (c mod 2) (0:Alice, 1:Bob) chooses to label the vertex 'i' with label 'color' *)
            (
               let w' = Array.copy w in
               w'.(i) <- (color,[]);
            begin
               if i-2 >= 0 then w'.(i-2) <- remove' color w'.(i-2);
               if i+2 <= m-1 then w'.(i+2) <- remove' color w'.(i+2);
               if i+1 <= m-1 then w'.(i+1) <- remove' (color-1) (remove' color (remove' (color+1) w'.(i
                  +1)));
               if i-1 >= 0 then w'.(i-1) <- remove' (color-1) (remove' color (remove' (color+1) w'.(i
                  -1)));
            end;
      (* we have modified the state of the game according to the rules of the labelling game *)
               let b = ref false in
               for j = 1 to 2 do
                  b := (!b)||((fst(w'.(max (i-j) 0)) = -1)&&(snd(w'.(max (i-j) 0)) = []))||((fst(w'.(min
                        (i+j) (m-1))) = -1)&&(snd(w'.(min (i+j) (m-1))) = []));
               done;
      (* b is true is the move played has created an unlabelled vertex with no feasible label *)
```

```
            if (!b = true)&&(c+1 < m) then 1 (* In this case, Bob wins *)
            else
            (
              let l'= choice w' in
              let winner = aux w' (c+1) l' in
              if winner=(c mod 2) then winner (* In this case, at this step of the game, the player
                  (c mod 2) has a winning strategy if he/she labels the vertex 'i' with label '
                  color' *)
              else aux w c r (* otherwise, we try the next feasible move on the list *)
            )
          )
      )
  in let w = init_path m n in
  aux w 0 (choice w);;

let gln_path m = (* the result is the game labelling number of the path on m vertices *)
  let n = ref 0 in let b = ref 1 in
  while !b=1 do
    n := !n + 1;
    b := solution_path m !n;
  done;
!n;;

for m = 1 to 6 do
  print_string "The␣game␣labelling␣number␣of␣a␣path␣on␣"; print_int m;
  print_string "␣vertices␣is␣:␣"; print_int (gln_path m); print_string "\n";
done;;
```

References

[1] H. L. Bodlaender. On the complexity of some coloring games. *Graph Theorical Concepts in Computer Science, Lecture Notes in Comuter Science*, 484:30–40, 1991.

[2] Ma-Lian Chia, Huei-Ni Hsu, David Kuo, Sheng-Chyang Liaw, and Zi-teng Xu. The game l(d,1)-labelling problem of graphs. *Discrete Mathematics*, 312:3037–3045, 2012.

[3] U. Faigle, U. Kern, H. Kierstead, and W. T. Trotter. On the game chromatic number of some classes of graphs. *Ars Combin.*, 35:143–150, 1993.

[4] M. R. Garey, D. S. Johnson, and L. Stockmayer. Some simplified np-complete problems. *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, pages 47–63, 1974.

[5] Jerrold R. Griggs and Roger K. Yeh. Labeling graphs with a condition at distance 2. *Siam Journal on Discrete Mathematics*, 5(4):586–595, 1992.

[6] D. Guan and X. Zhu. Game chromatic number of outerplanar graphs. *J. Graph Theory*, 30:67–70, 1999.

[7] D. J. Guan and Xuding Zhu. Game chromatic number of outerplanar graphs. *J Graph Theory*, 30:67–70, 1999.

[8] H. A. Kierstead and Daqing Yang. Very asymmetric marking games. *Order*, 22(Issue 2):93–107, 2005.

[9] J. Wu and X. Zhu. Lower bounds for the game colouring number of partial k-trees and planar graphs. *Discrete Mathematics*, 308:2637–2642, 2008.

[10] X. Zhu. The game coloring number of planar graphs. *J. Combinatorial Theory (B)*, 75:245–258, 1999.

[11] Xuding Zhu. Refined activation strategy for the marking game. *Journal of Combinatorial Theory*, Ser. B 98:1–18, 2008.