ON THE COMPLEXITY OF SOME COLORING GAMES

Hans L. Bodlaender

RUU-CS-89-27
November 1989

Utrecht University

Department of Computer Science

Padualaan 14, P.O. Box 80.089,
3508 TB Utrecht, The Netherlands,
Tel. : ... + 31 - 30 - 531454

ON THE COMPLEXITY OF SOME COLORING GAMES*

Hans L. Bodlaender
Department of Computer Science, Utrecht University
P.O.Box 80.089, 3508 TB Utrecht, the Netherlands

Abstract

In this paper we consider the following game: players must alternately color the lowest numbered uncolored vertex of a given graph $G = (\{1, 2, \cdots, n\}, E)$ with a color, taken from a given set C, such that never two adjacent vertices are colored with the same color. In one variant, the first player which is unable to move, loses the game. In another variant, player 1 wins the game if and only if the game ends with all vertices colored. We show that for both variants, the problem to determine whether there is a winning strategy for player 1 is PSPACE-complete for any C with $|C| \geq 3$, but the problems are solvable in $\mathcal{O}(n + e\alpha(e, n))$, and $\mathcal{O}(n + e)$ time, respectively, if $|C| = 2$. We also give polynomial time algorithms for the problems with certain restrictions on the graphs and orderings of the vertices. We give some partial results for the versions, where no order for coloring the vertices is specified.

1 Introduction.

Games can be used to model situations where different parties have conflicting interests, or to model the "worst type of erroneous behavior of a system". In the latter case, we assume that an erroneous system is malicious and uses an intelligent strategy to try to prevent us from reaching our goal. If we are able to deal with this type of behavior, we are also able to deal with all weaker types of errors.

In this paper we concentrate on the question: given a certain game, how hard is it to determine whether there is a winning strategy for the first (or second) player. We will further only discuss 2-player games, where players have perfect information.

With the notion of "game", we usually mean a class of actual games (instances), where each game (instance) is distinguished from others in the class ony by its

*This research was partially supported by the ESPRIT II Basic Research Actions Program of the EC under contract no. 3075 (project ALCOM).

starting position and playing area, but uses the same type of moves. With the complexity of a game (class), we denote the complexity of the following problem: given a game (instance) from this class, does player 1 have a winning strategy in this game (instance)?

Several generalizations of well-known games, like chess, go, and checkers have been shown to be PSPACE-complete or EXPTIME-complete [5, 7, 11]. Also, PSPACE-completeness and EXPTIME-completeness results have been established for several more abstract games [1, 4, 6, 12, 14, 15]. For an overview, see e.g. [8, 9].

In this paper we consider the following game: given are an undirected graph $G = (V, E)$, a linear ordering of G (that is: a bijection $f : V \to \{1, \cdots, |V|\}$) and a finite set of colors C. Players 1 and 2 must alternately color the first uncolored vertex, i.e., player 1 starts with coloring vertex $f^{-1}(1)$, then player 2 colors vertex $f^{-1}(2)$, player 1 continues with coloring $f^{-1}(3)$, etc. When coloring a vertex v, the player must assign a color from C to v that has not already been assigned to a neighbor of v. In one variant of the game, the first player that is unable to move (either because no uncolored vertices are left, or because the vertex that must be colored has for each $c \in C$ a neighbor that is already colored with c) loses the game. We call this game the SEQUENTIAL COLORING GAME. In another variant of the game, player 1 wins, if and only if all vertices are colored at the end of the game. We call this variant the SEQUENTIAL COLORING CONSTRUCTION GAME.

One could also consider the following variants of the games: each vertex is owned by a player (i.e., we have a function *owner* : $V \to \{1, 2\}$). In the i'th move, the player that owns vertex $f^{-1}(i)$ must color this vertex. In this variant of the SEQUENTIAL COLORING GAME, we must specify which player wins if all vertices are colored. It is not hard to see that these variants have the same complexity as the original games (e.g., add an isolated vertex between each two successive vertices that are owned by the same player). NP-hardness of these games can be proved easily (by transformation from CHROMATIC NUMBER). Hence, SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME are also NP-hard.

However, a stronger result can be proved. In section 2 we show that both types of games are PSPACE-complete, for any fixed set of colors C with $|C| \geq 3$.

In sections 3 and 4 we look at special cases of the problems. In section 3 we consider the case that there are only 2 colors, and derive $\mathcal{O}(n+e\alpha(e,n))$ and $\mathcal{O}(n+e)$ time algorithms, respectively. (α is the inverse Ackermann-function.) In section 4 we give polynomial algorithms for the problems with certain restrictions on the graphs and linear orderings. In section 5 we give some partial results for versions, where no order is specified in which the vertices must be colored.

Throughout this paper we use the name of a game to denote the problem to determine whether there is a winning strategy for player 1 in a given instance of the game. In this paper, we use the notion of PSPACE-completeness under transformations that use logarithmic work space (log-space reductions).

2 PSPACE-completeness of SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME

In this section we show that the problems, to decide whether there is a winning strategy for player 1 in instances of SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME, are PSPACE-complete. We use a transformation from QUANTIFIED 3-SATISFIABILITY. This is the following, restricted version of QUANTIFIED BOOLEAN FORMULAS:

QUANTIFIED 3-SATISFIABILITY
Instance: Set $X = \{x_1, \cdots, x_n\}$ of variables, well-formed quantified Boolean formula $F = (Q_1 x_1)(Q_2 x_2) \cdots (Q_n x_n) F_0$, where F_0 is a Boolean expression in conjunctive normal form with three literals per clause, and each Q_i is either $\forall$ or $\exists$.
Question: Is F true ?

In [13] Schaefer proved that this problem is PSPACE-complete, but becomes solvable in polynomial when the problem is restricted to two literals per clause. It is not hard to see, that instances may also be restricted to instances of the form $\exists x_1 \forall x_2 \exists x_3 \cdots \forall x_n F_0$.

Theorem 2.1
SEQUENTIAL COLORING GAME with 3 colors is PSPACE-complete.

Proof.
It is easy to see that the problem is solvable in polynomial space, e.g. by using backtracking.

To show PSPACE-hardness, we use a transformation from QUANTIFIED 3-SATISFIABILITY.

Let a formula $F = \exists x_1 \forall x_2 \exists x_3 \cdots \exists x_{n-1} \forall x_n F_0$ be given, where F_0 is a boolean expression over $x_1 \cdots x_n$ in conjunctive normal form with 3 literals per clause. Let $C = \{c_1, \cdots, c_m\}$ be the set of clauses in F_0. We write $l_i \in c_j$, if literal l_i appears in the j'th clause of F_0.

We transform F as follows: Let $G = (V, E)$ be the graph, defined by

$$
\begin{aligned}
V = \{\ & \text{true, false}, X, x_1, \overline{x}_1, x_2, \overline{x}_2, \cdots, x_n, \overline{x}_n, c_1, c_2, \cdots, c_m, d_1, \cdots, d_m, e\} \\
E = \{\ & (\text{true, false}), (\text{true}, X), (\text{false}, X)\ \} \cup \\
& \{\ (X, x_i) \mid 1 \le i \le n\} \cup \{(X, \overline{x_i}) \mid 1 \le i \le n\} \cup \\
& \{\ (x_i, \overline{x}_i) \mid 1 \le i \le n\} \cup \\
& \{\ (l, c_i) \mid l \in c_i, l \in \{x, \cdots, x_n, \overline{x}_1, \cdots, \overline{x}_n\}, 1 \le i \le m\} \cup \\
& \{\ (c_i, e) \mid 1 \le i \le n\} \cup \{(\text{false}, e), (X, e)\} \cup \\
& \{\ (\text{false}, c_i) \mid 1 \le i \le m\}
\end{aligned}
$$

Let f be the linear ordering of G, given by

$$\begin{array}{lll} f(\text{true})=1, & f(\text{false})=2, & f(x) = 3, \\ f(x_i) = & i+3 & (1 \leq i \leq n) \\ f(\overline{x}_i) = & i+n+3 & (1 \leq i \leq n) \\ f(d_i) = & 2i+2n+2 & (1 \leq i \leq m) \\ f(c_i) = & 2i+2n+3 & (1 \leq i \leq m) \\ f(e) = & 2m+2n+4 & \end{array}$$

We now claim: there is a winning strategy for player 2, in the game, played on G with ordering f with 3 colors, if and only if F is true.

First note that true, false and X must be colored with different colors. We call the color, given to true "true", the color given to false "false", and the color given to X "X". Note that variables $x_i, \overline{x}_i$ must be colored with true or false.

Suppose F is true. Then player 2 can color the vertices x_i (i odd) in such a way, that each clause $c_i \in C$ contains a variable x_i that is colored true or a variable $\overline{x}_i$ with x_i colored false, no matter what strategy player 1 uses to color the vertices x_i with i even. Note that each vertex $\overline{x}_i$ is colored true, if x_i is colored false, and vice versa. Hence, player 2 can make sure that each clause $c_i \in C$ has at least one variable colored true. Now, player 1 must color all vertices c_i with X, as they are adjacent to false, and to a vertex, colored true. So player 2 can color e with true, and wins the game, as player 1 cannot move, because no vertices are left.

Suppose F is false. Player 1 now can make sure, that at least one clause c_i only contains vertices that are colored false. So player 2 can color c_i with true, and player 1 loses the game because he cannot color e. This ends the proof of our claim.

Finally notice that the transformation $F \rightarrow (G, f)$ can be carried out in logarithmic work space. □

With an almost identical proof one can show:

Theorem 2.2
SEQUENTIAL COLORING CONSTRUCTION GAME with 3 colors is PSPACE-complete.

With similar proofs, one can show that the PSPACE-completeness results also hold for any fixed number of colors ≥ 3. One can also restrict the graphs to graphs with bounded degree.

Theorem 2.3
SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME with 3 colors, are PSPACE-complete when restricted to graphs with maximum vertex degree ≤ 5.

Proof.
We indicate how to modify the construction in the proof of theorem 2.1.

e is replaced by m vertices $e_1, \cdots, e_m$. Between each pair e_i, e_{i+1}, we put a new isolated vertex f_i, such that player 2 must color all vertices e_i. Each e_i is made adjacent to false, X and c_i. If player 1 can color a vertex c_i with true, then player 2 loses the game, as he cannot color e_i, otherwise he wins the game.

Next we observe that QUANTIFIED 3-SATISFIABILITY is PSPACE-complete, also if we require that no literal appears in more than 3 clauses (use a similar transformation as is used for 3SAT.) This means that each $x_i, \overline{x}_i, c_i$ has degree at most 4.

Note that there are at most $2n+3m$ edges to {true,false, X}. We replace { true, false, X} by $3(2n + 3m)$ vertices true_i,false_i, $X_i (1 \leq i \leq 2n + 3m)$, with edges $(\text{true}_i, \text{false}_i)$, (true_i, X_i),(false_i, X_i), (false_i,true_{i+1}), (X_i, true_{i+1}), (X_i, false_{i+1}). Take $f(\text{true}_i) = 3i - 2$; $f(\text{false}_i) = 3i - 1$; $f(X_i) = 3i$. All edges (j, v), $j \in$ {true,false,X} $\not\ni v$ are replaced by an edge (j_i, v), such that each j_i ($j \in$ { true, false, X}) has at most one adjacent edge of this type. Now each j_i has degree ≤ 5. Note that all vertices true_i must be colored with the same color, and likewise for all false_i, and all X_i.

As in theorem 2.1 one now can show that the game on the resulting graph and ordering has a winning strategy for player 2, if and only if the formula F is true. Finally notice that the transformation still can be carried out in logarithmic work space. □

3 SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME with 2 colors.

In this section we show that SEQUENTIAL COLORING CONSTRUCTION GAME and SEQUENTIAL COLORING GAME can be solved in linear, or almost linear time, when we restrict ourselves to instances with only 2 colors.

We first introduce some terminology. To ease presentation, we assume $V = \{1, 2, \cdots, n\}$, and $f(i) = i$. The owner of a vertex i is the player that must color the vertex, i.e. $owner(i) = 1 \Leftrightarrow i$ is odd. A vertex $i \in V$ that has only edges to vertices with a larger number are called *sources*. Notice that the only moments when a player can make a decision is when he colors a source-vertex: when they must color a non-source vertex, then they either lose, or have exactly one possible color available to color the vertex. For a color $c \in C$, denote the other color in C with $\overline{c}$. A *predecessor* of a vertex j is a vertex i with $i < j$ and $(i, j) \in E$. The problem for SEQUENTIAL COLORING CONSTRUCTION GAME with 2 colors is easy to resolve.

Theorem 3.1
SEQUENTIAL COLORING CONSTRUCTION GAME with 2 colors is solvable in $\mathcal{O}(n+e)$ time.

Proof.
We claim that player 2 can win the game, if and only if G is not bipartite, or player 2 owns a source vertex that is not the first vertex of a connected component of G. Clearly, this property can be tested in $\mathcal{O}(n+e)$ time.

If G is not bipartite, then player 2 will always win the game. Suppose G is bipartite. If player 2 owns a source vertex v that is not the first vertex of a connected component, then he can color this vertex such that no coloring of G with 2 colors exists that extends the coloring so far. (Player 2 looks to a vertex w, in the same component as v with $w < v$. If the distance from v to w is even, then player 2 colors v with a different color from the color of w, otherwise he colors v with the same color as w.) So, in this case, player 2 wins the game.

If G is bipartite, and player 1 owns all source vertices, that are not the first vertex in a connected component, then he wins by using the following strategy: when coloring such a source vertex, find the smallest $w < v$ that belongs to the same component. If the distance from v to w is even, then color v with the same color as w, otherwise give it a different color. By induction, all vertices with even distance to w get the same color as w and all vertices with odd distance to w get a different color. Eventually, all vertices will be colored and player 1 wins the game. □

Next we discuss an algorithm for SEQUENTIAL COLORING GAME with 2 colors. We first give the algorithm, and then discuss its correctness and running time.

In the algorithm we keep a collection of sets of vertices C. Each set has associated with it a "buddy" (which is another set in C), a number "first" (which is the first vertex in the set or its buddy), and an "owner" (which is the owner of "first").

We use the following algorithm.

$C := \emptyset$; game-end:= false; $i := o$
while not (game-end) and $i < n$
do begin $i := i + 1$;
Let $S \subseteq C$ be the collection of sets, which contain predecessors of i.
if $S = \emptyset$ (i.e., i is a source)
then begin add set $a_i = \{i\}$ to C;
add set $b_i = \emptyset$ to C;
buddy$(a_i) := b_i$; buddy$(b_i) := a_i$;
first$(a_i) := i$; first$(b_i) := i$;
owner(a_i) := owner(i); owner(b_i) := owner(i)
end
else if $|S| = 1$
then begin suppose $S = \{s\}$;

add i to the set buddy(s)
end
else if $\exists s \in S$
then begin player owner(i) loses the game;
game-end := true
end
else if $\exists s_1 \in S : \exists s_2 \in S :$ first(s_2) > first(s_1) and
owner s_2) $\neq$ owner(i)
then begin player owner(i) loses the game;
game-end:= true,
end
else Take $s_1 \in S$ with $\forall s_2 \in S :$ first(s_2) $\geq$ first(s_1)
for all $s_2 \in S : s_2 \neq s_1$
do begin Union (s_1, s_2);
Union (buddy(s_1), buddy(s_2));
add i to the set buddy(s_1)
end
end
if not(game-end) then player 1 loses the game, if and only if n is even.

Here Union is a procedure that adds the vertices of the second set to the first set, and removes then the second set from C.
To analyze this algorithm, let C_i be the collection C of sets after the i'th iteration of the main loop, e.g., $C_0 = \emptyset, C_1 = (\{1\}, \emptyset)$. Let G_1 be the subgraph of G, induced by $\{1, \cdots, i\}$, i.e. $G_i = (\{1, \cdots, i\}, \{(v, w) \in E \mid 1 \leq v, w \leq j\})$.

Note the following: c_i forms a partition of $\{1, \cdots, i\}$. For all $s \in C_i$) : buddy(buddy(s)) = s. For each connected component of G_i, there are two sets s, buddy(s), that contain all vertices of the connected component. For every edge (v, w) in G_i, there are two sets s, buddy(s), with $v \in s$, $w \in$ buddy(s). Hence, as long as game-end is false, G_i is bipartite. Each of these properties can be proved with induction on i.

Lemma 3.2
The algorithm outputs the correct player that has a winning strategy.

Proof.
Suppose the algorithm outputs that player p loses the game, in the i'th iteration of the main loop. (Take $i = n + 1$, if the output is given after the main loop, i.e., when game-end stayed false.) Let q be the other player. We give a winning strategy for player q such that the game ends with at most $i - 1$ colored vertices.

Call a move *i-wrong*, if a player colors a vertex $v \in s \in C_{i-1}$ with a color c, such that

(i) $\exists w \in s$, which has been colored with color $\overline{c}$,
or

(ii) $\exists w \in$ buddy(s), which has been colored with color c.

A move is *i-good*, if it is not i-wrong. Let S_i be the set S, computed at the i-th iteration.

As long as player p makes only i-good moves, player q uses the following strategy: he makes only i-good moves, obeying the following: if $\exists s_1 \in S_i : \exists s_2 \in S_i :$ first$(s_2) >$ first(s_1), and owner$(s_2) \neq$ owner (i), then he considers fixed s_1, s_2 with this property. Clearly, player q owns first$(s_2$. When player q must color first(s_2), then

- if (first$(s_1) \in s_1 \Leftrightarrow$ first$(s_2) \in s_2)$, then color first(s_2) with a different color as first(s_1).
- otherwise color first(s_2) with the same color as first(s_1).

Suppose player p also makes only i-good moves. Then it follows that for each $s \in S_i$: all vertices in s are colored with the same color, which is different from the color of the vertices in buddy(s).

Now player p cannot color i: if $\exists s \in S$ with buddy$(s) \in S$, then i must be colored differently from a vertex in s, and from a vertex in buddy(s), but these will have different colors. If $\exists s_1 \in S : \exists s_2 \in S :$ first$(s_2) >$ first(s_1) and owner(s_2), then player q has colored first(s_2) in such a way, that the predecessor of i in s_2 is colored differently from the predecessor of i in s_1. In both cases p loses the game.

Next we suppose player p makes at least one i-wrong move. This will only have the effect that player p loses at an earlier move. Let the first i-wrong move of player p be at move j. j must be a source. Look at the smallest i', where in the i'th iteration a set s' with $j \in s' \cup$ buddy(s) is added (by Union (s'', s')) to a set s'' with first$(s'') <$ first(s') (i' exists and $i' < i$, because $\{j\} \in C_j$ and $\{j\} \subseteq s$ with first$(s) < j$ for some $s \in C_{i-1}$).

If both player 1 and 2 make only i'-good moves, then player p will lose when he must color i'. Notice that i-good moves are also i'-good. It follows that if $v, w \in s \in C_{i-1}$, then v and w are colored with the same color. Because the test " $\exists s_1 \in S : \exists s_2 \in S :$ first$(s_1) >$ first(s_2) and owner$(s_2) \neq$ owner(i')" is false, player p must own i'. i' cannot be colored, because its predecessor in s' is colored differently from its predecessor in s''.

Player q makes i'-good moves until player p makes an i'-wrong move. Then let i'' be the largest i'' such that all moves up to that point are i''-good. With an argument as above, it follows that if both players make only i''-good moves, then player p loses on or before move i''. If player p makes an i''-wrong move, then repeat the construction with i'' instead of i', etc.

It follows that the resulting strategy gives player q the victory in the game. □

Theorem 3.3

SEQUENTIAL COLORING GAME with 2 colors is solvable in $\mathcal{O}(n + e\alpha(e, n))$ time.

Proof.
We can implement the algorithm using the well-known Union-Find data structure of Tarjan ([16]). Our algorithm requires at most n Unions and e Finds. □

4 Special classes of graphs and linear orderings.

In this section we consider two special classes of graphs and corresponding linear orderings. In both cases we give polynomial time algorithms for the coloring problems.

Definition.
A linear ordering $f : V \rightarrow \{1, \cdots, |V|\}$ of a graph $G = (V, E)$ is said to have vertex separation number K, if $K = \max_{1 \leq i \leq |V|} |\{v \in V \mid f(v) \leq i \wedge \exists (v, w) \in E : f(w) > i\}|$.

The problem to determine a linear ordering with minimum vertex separation number is NP-complete, but becomes solvable in polynomial time if k is bounded by a constant (see e.g. [3, 10]).

Theorem 4.1
SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME are solvable in $\mathcal{O}(n)$ time, when restricted to graphs, that are given with a linear ordering with vertex separation number bounded by a constant K.

Proof.
First note that if $|C| \geq K + 1$, then a vertex can always be colored, when the game is played on a graph given with a linear ordering with vertex separation number at most K, as each vertex has at most K predecessors. So, in the remainder we suppose that $|C| \leq K$.

We first consider SEQUENTIAL COLORING GAME. A *state* (of a game, played on G with linear ordering f and set of colors C) is a function $c : \{1, \cdots, i\} \rightarrow C$, such that $\forall v, w : 1 \leq f(v), f(w) \leq i : (v, w) \in E \Rightarrow c(v) \neq c(w)$. Consider a game-instance of SEQUENTIAL COLORING GAME with vertex separation number $\leq K$. The set of all possible states (of a game, played on this instance) is denoted by X. We now make a directed acyclic graph $H = (X, F)$, with $F = \{(c, c') \mid c, c' \in X$ and $\exists i : c : \{1, \cdots, i\} \rightarrow C, c' : \{1, \cdots, i+1\} \rightarrow C, \forall j, 1 \leq j \leq i : c(j) = c'(j)\}$, i.e., we have an edge from a state to each state that is reached after one move. We assume that there is an edge from $0 : \emptyset \rightarrow C$ to each $c : \{1\} \rightarrow C$.

A state $c \in X$ with outdegree 0 is called a *losing* state (because a player, that must move from c loses the game, because no move is possible). A state with outdegree ≥ 1 is a winning state, if there is an edge to a losing state, and it is a losing state, if all outgoing edges lead to winning states. Player 1 wins the game, if and only if the start-state $0 : \emptyset \rightarrow C$ is a winning state. So the problem remains to determine whether $0 : \emptyset \rightarrow C$ is a winning state.

Hereto we define an equivalence relation $\equiv$ on states: a state $c : \{1, \cdots, i\} \to C$ is equivalent with state $c' : \{1, \cdots, i'\} \to C$ ($c \equiv c'$), if and only if $i = i'$ and for all $v \in V$ with $f(v) \leq i$ and $\exists (v, w) \in E : f(w) > i : c(v) = c'(v)$.

Claim 4.1.1
If $c \equiv c'$, then c is a winning state, if and only if c' is a winning state.

Proof.
This follows by induction, starting at states with outdegree 0. If $c \equiv c'$ and c has outdegree 0, then c' also has outdegree 0: suppose $c : \{1, \cdots, i\} \to C$ with $i < |V|$ has outdegree 0. Then $\{c(v) \mid f(v) \leq i,\ (v, f^{-1}(i+1)) \in E\} = C$. But then also $\{c'(v) \mid f(v) \leq i,\ (v, f^{-1}(i+1)) \in E\} = C$. So starting with $c', i+1$ cannot be colored, hence outdegree c' is 0.

If $c \equiv c'$ and c has an edge to a losing state c'', then c' has also an edge to a losing state. Let $c : \{1, \cdots, i\} \to C$. Now take $c''' : \{1, \cdots, i+1\} \to C$ as follows: $c'''(j) = c'(j)$ for $1 \leq j \leq i$, and $c'''(i+1) = c''(i+1)$. Now $c', c''') \in F$ and $c'' \equiv c'''$, hence, by induction, c''' is a losing state. □

It can be seen easily that equivalence classes of states can be characterized by pairs $(i, \tilde{c})$, with $\tilde{c}$ a function $\{v \mid f(v) \leq i \wedge \exists w : (v, w) \in E \wedge f(w) > i\} \to C$. We now can define a directed graph $\tilde{H}$ on the equivalence classes with edges from class $(i, \tilde{c})$ to class $(i+1, \tilde{c'})$, if $\exists c \in (i, \tilde{c}) : \exists c' \in \{i+1, \tilde{c'}\}$ with $(c, c') \in F$. This directed acyclic graph has at most $n \cdot |C|^k = \mathcal{O}(n)$ vertices, and each vertex in $\tilde{H}$ has outdegree $\leq K$ (different outgoing edges correspond to different colors for $f^{-1}(i+1)$, and $|C| \leq K$). A class is losing, if and only if it has outdegree 0, or has only outgoing edges to winning classes. It is now not hard to find a linear time algorithm that builds $\tilde{H}$, and determines for each class (vertex in $\tilde{H}$) whether it is losing or winning. (Start computing for each class $(n, \tilde{c})$ whether it is losing or winning, then classes $(n-1, \tilde{c})$, $(n-2, \tilde{c})$ etc.) In this way we determine whether the class containing $0 : \emptyset \to C$ is winning, and hence whether there is a winning strategy for player 1 in linear time.

We can use a similar argument, with some small modifications for the SEQUENTIAL COLORING CONSTRUCTION GAME. We only must change the notions of "winning" and "losing". A state $c : \{1, \cdots, i\} \to C$ is "winning for player 1", if and only if $i = n$ or c has an edge to a state, winning for player 1 and owner$(i+1) = 1$ or all outgoing edges from c go to states, winning for player 1 and owner$(i+1) = 2$. (If $i = n$, then player 1 has won the game. If $i < n$, and owner$(i+1) = 1$, then player 1 will move to a state that is winning. Player 2 can only move from states, winning for player 1 to other states, winning for player 1. If he can move from a state c to a state, losing for player 1, then c is also losing for player 1). One can now use arguments and an algorithm, similar to the case of SEQUENTIAL COLORING GAME, to obtain also a linear time algorithm for SEQUENTIAL COLORING CONSTRUCTION

GAME on graphs, given with a linear ordering with vertex separation number $\leq K$. □

Definition.
A linear ordering $f : V \rightarrow \{1, \cdots, |V|\}$ of a graph $G = (V, E)$ is said to have *bandwidth* K, if $K = \max\{|f(v) - f(w)| \mid (v, w) \in E\}$. It is said to have *cutwidth* K, if $K = \max_{1 \leq i \leq n} |\{(u, v) \in E \mid f(u) \leq i < f(v)\}|$.

Corollary 4.2
SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME can be solved in $\mathcal{O}(n)$ time for graphs with a linear ordering with bandwidth $\leq K$ or cutwidth $\leq K$, for constants K.

Proof.
If the bandwidth or cutwidth of a linear ordering is bounded by a constant, then the vertex separation number of the linear ordering is also bounded by a constant (this can easily be proved, similar to results in [2]). □

We now consider linear orderings, that are obtained by reversing a linear ordering with small vertex separation number.

Definition.
A linear ordering $f : V \rightarrow \{1, 2, \cdots, |V|\}$ of $G = (V, E)$ is said to have *reversed vertex separation number* K, if the linear ordering f^R of G, defined by $f^R(v) = |V| + 1 - f(v)$ has vertex separation number K.

Theorem 4.3
SEQUENTIAL COLORING GAME and SEQUENTIAL COLORING CONSTRUCTION GAME can be solved in polynomial time on graphs with a linear ordering with reversed vertex separation number bounded by a constant K.

Proof.
We use a method, similar to the method in the proof of theorem 4.1, but define equivalence classes of states in a different, slightly more complex way. Note that we cannot limit the number of colors to K in this case.

Use the same concept of states as in the proof of theorem 4.1. Let $\equiv$ be an equivalence relation of states, defined by $c : \{1, \cdots, i\} \rightarrow C$ is equivalent to $c' : \{1, \cdots, i'\} \rightarrow C$ ($c \equiv c'$) if and only if $i = i'$ and for all $j > i$: $\{c(v) \mid f(v) \leq i \wedge (v, f^{-1}(j)) \in E\} = \{c'(v) \mid f(v) \leq i \wedge (v, f^{-1}(j)) \in E\}$.

Claim 4.3.1
If $c \equiv c'$, then c is a winning state, if and only if c' is a winning state.

We omit the proof of this claim. (Use induction, starting at states with outdegree 0.)

Let $S_i = \{v \mid f(v) > i \wedge \exists j \leq i : (f^{-1}(j), v) \in E\}$. Assuming that f has reversed vertex separation number $\leq K$, it follows that $\forall i, 1 \leq i \leq n : |S_i| \leq K$. We can characterize each equivalence class of the relation $\equiv$ by a pair (i, x), with x a function $x : S_i \rightarrow \mathcal{P}(C)$. x denotes the "sets of forbidden colors", i.e. if the class containing c is characterized by (i, x), then for all $w \in S_i : x(w) = \{c(v) \mid f(v) \leq i$ and $(v, w) \in E\}$. Note that this set is empty for $w \neq S_i$.

To get a polynomial number of equivalence classes, we must introduce a second equivalence notion. Write $(i, x) \sim (i', y)$, if and only if $i = i'$ and there exists a bijection $\varphi : C \rightarrow C$, such that for all $v \in S_i : \varphi(x(vv)) = y(v)$, i.e., y can be obtained from x by renaming the colors. It is easy to see that if $(i, x) \sim (i', y)$, then each state in (i, x) is winning, if and only if each state in (i', y) is winning.

We can characterize $\sim$-equivalence classes by pairs (i, A), with A a function $A : \mathcal{P}(S_i) \rightarrow \{0, 1, \cdots, |C|\}$, as follows: if (i, x) belongs to the class, characterized by (i', A), then $i = i'$, and for all $S \subseteq S_i : A(S)$ equals the number of colors c, that belong to each set $x(v)$ for $v \in S$, and to no set $x(v)$ for $v \in S_i - S$. It is not hard to show that this is indeed a correct characterization of $\sim$-equivalence classes.

As there are at most $(|C|+1)^{2^K}$ functions $A : \mathcal{P}(S_i) \rightarrow \{0, 1, \cdots, |C|\}$, it follows that the total number of $\sim$-equivalence classes is polynomial. So we can make a directed acyclic graph with $\sim$-equivalence classes as vertices and then compute in time, linear to the number of $\sim$-equivalence classes, which $\sim$-equivalence classes correspond to winning states, and which correspond to losing states. There is a winning strategy for player 1, if the class $(0, \emptyset \rightarrow \{0, 1, \cdots, |C|\})$ corresponds to a (set of) winning states. □

5 Final remarks and related games.

There are several interesting open problems, concerning the complexity of SEQUENTIAL COLORING (CONSTRUCTION) GAME for special classes of inputs. For instance, what is the complexity of the problems, when G is required to be a tree?

Interesting variants arise when players are not required to color the vertices in a pre-specified order. In the COLORING GAME and COLORING CONSTRUCTION GAME, a move consists of choosing a vertex v and a color c, that has not been chosen already before for a neighbor of v. In the COLORING GAME, the first player that is unable to move loses the game. In the COLORING CONSTRUCTION GAME, player 1 wins if and only if the game ends with all vertices colored. (A game instance for these games now consists of an undirected graph and a set of colors.)

Theorem 5.1 [14]
COLORING GAME with 1 color is PSPACE-complete.

For the proof, see [14] for the game KAYLES.

Theorem 5.2
COLORING GAME with 2 colors is PSPACE-complete.

Proof.
Of course, the problem is in PSPACE. To prove PSPACE-hardness, we transform from the 1-color case.

Let $G = (V, E)$ be an undirected graph. Let $G' = (V', E')$ be defined as follows: $V = \{v_i \mid v \in V,\ i = 1 \vee i = 2\} \cup \{x\}$; $E = \{(v_1, v_2) \mid v \in V\} \cup \{(v_i, w_j) \mid (v, w) \in E,\ i, j \in \{1, 2\}\} \cup \{(v_i, x) \mid v \in V,\ i \in \{1, 2\}\}$. Now there is a winning strategy for player 1 for the 1-color game, played on G, if and only if there is a winning strategy for player 2 for the 2-color game on G'. First note that player 1 must start coloring x : if not then player 2 can win by answering each move of player 1: if player 1 colors v_i with color j, then player 2 colors v_{3-i} with color $\bar{j}$. After x has been colored, the game will be similar to the 1-color game on G : if a vertex v_i is colored, then neither v_{3-i}, nor any vertex w_j with w adjacent to v in G' can be colored. □

The complexity of COLORING GAME with any fixed number ≥ 3 of colors is open. Also, the complexity of COLORING CONSTRUCTION GAME is an interesting open problem.

Define the chromatic game number of a graph to be the smallest integer k, such that there is a winning strategy for player 1 in the COLORING CONSTRUCTION GAME with k colors.

Clearly, the game-chromatic number of a graph is at least its chromatic number and at most its maximum vertex degree plus 1.

Theorem 5.3
For every tree T: the game-chromatic number of T is at most 5.

Proof.
Player 1 considers T as a rooted tree. Let V_1 be the set of vertices with even distance to the root, let V_2 be the set of vertices with odd distance to the root. Player 1 uses the following winning strategy for the COLORING CONSTRUCTION GAME with colors $\{1, 2, 3, 4, 5\}$ on T :

- if no vertex is colored, then color the root with color 1.
- if the father v of the last vertex w that has been colored by player 2 is not yet colored, and the father of v is not colored, then player 1 colors v
  - with color 1, if $v \in V_1$ and color$(w) \neq 1$
  - with color 2, if $v \in V_1$ and color$(w) = 1$
  - with color 3, if $v \in V_1$ and color$(w) \neq 3$

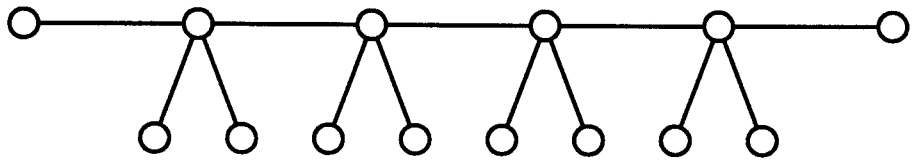

Figure 1

- with color 4, if $v \in V_1$ and color$(w) = 3$

- if the father v of the last vertex w that has been colored by player 2, is not yet colored, and the father of v is colored, then player 1 colors v with an arbitrary possible color.

- if the father of the last vertex w that has been colored by player 2, has been colored, then v choses an arbitrary vertex whose father has been colored, and gives it an arbitrary possible color.

We claim that this strategy always gives a possible, allowed move for player 1. It is important to note that with this strategy, after each move of v_1 : $\forall v \in V$: if v is colored and the father of v is not colored, then $v \in V_1 \Rightarrow v$ is colored with color 1 or color 2, and $v \in V_2 \Rightarrow v$ is colored with color 3 or color 4. With this observation, correctness easily follows. □

It is presently open whether there are trees with game-chromatic number 5. However, we can show the following:

Theorem 5.4
There exists a tree T, with game-chromatic number ≥ 4.

Proof.
Use the graph G, given in figure 1.

We give a strategy for player 2 to avoid a coloring with 3 colors.
In its first move, player 2 colors a vertex with distance 3 to the vertex player 1 colored in its first move, with the same color. Let this color be c. Note that G now has a subgraph of the type, shown in figure 2.

We consider 2 cases:
Case 1: Player 1 colors 2 or 3 in its second move. He must use a color $\neq c$. W.l.o.g. suppose is colored with c'. Then player 2 colors 7 with the color $\neq c, c'$, and wins the game, as 3 cannot be colored.

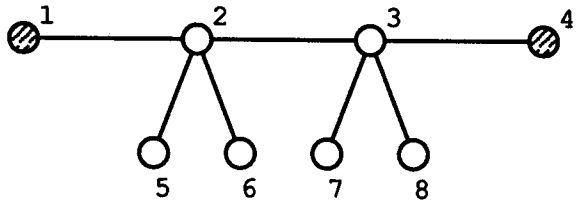

Figure 2

Case 2: Player 1 colors a vertex $v \neq 2,3$ in its second move. W.l.o.g. suppose $v \notin \{5,6\}$ and v is not adjacent to 5,6. Let c' be the color that player 1 used for v. Now player 2 colors 5 with c', if $c' \neq c$, and with a color $\neq c$, if $c = c'$. We consider 3 sub-cases.
Case 2a: Player 1 colors 2 or 3 in its third move. Player 2 now wins the game as in case 1.
Case 2b: Player 1 colors 6 or a neighbor of 6. Then player 2 colors 3 with a color $\neq c$, and $\neq$ the color of 5, and wins, because 2 cannot be colored.
Case 2c: Player 1 colors a vertex $v \neq 2,3,6$, or a neighbor of 6. Then player 2 colors 6 with a color $\neq c$ and wins the game, because 3 cannot be colored. □

It is an interesting open problem whether a result, similar to theorem 5.3 holds for planar graphs. This problem can be seen as "a game variant" to the famous 4-color theorem.

Conjecture
There exists a constant c, such that for all planar graphs G, the game-chromatic number of G is at most c.

In case the conjecture is true, what is the smallest c with this property?

Several other types of coloring games can be and have been proposed. (See e.g. [14] for the game SNORT ON GRAPHS and a PSPACE-completeness proof for this game.)

Acknowledgements

I would like to thank Richard Tan for some stimulating discussions.

References

[1] A. Adachi, S. Iwata, and T. Kasai. Some combinatorial game problems require $\Omega(n^k)$ time. *J. ACM*, 31:361–376, 1984.

[2] H. L. Bodlaender. Classes of graphs with bounded treewidth. Technical Report RUU-CS-86-22, Dept. of Computer Science, Utrecht University, Utrecht, 1986.

[3] H. L. Bodlaender. Improved self-reduction algorithms for graphs with bounded treewidth. Technical Report RUU-CS-88-29, Dept. of Computer Science, Utrecht University, 1988. To appear in: Proc. Workshop on Graph Theoretic Concepts in Comp. Sc. '89.

[4] S. Even and R. Tarjan. A combinatorial problem which is complete in polynomial space. *J. ACM*, 23:710–719, 1976.

[5] A. Fraenkel, M. Garey, D. Johnson, T. Schaefer, and Y. Yesha. The complexity of checkers on an $n \times n$ board — preliminary version. In *Proc. 19th Annual Symp. on Foundations of Computer Science*, pages 55–64, 1978.

[6] A. Fraenkel and E. Goldschmidt. Pspace-hardness of some combinatorial games. *J. Combinatorial Theory ser. A*, 46:21–38, 1987.

[7] A. Fraenkel and D. Lichtenstein. Computing a perfect strategy for n by n chess requires time exponential in n. *J. Combinatorial Theory ser. A*, 31:199–214, 1981.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, New York, 1979.

[9] D. Johnson. The NP-completeness column: An ongoing guide. *J. Algorithms*, 4:397–411, 1983.

[10] T. Lengauer. Black-white pebbles and graph separation. *Acta Inf.*, 16:465–475, 1981.

[11] D. Lichtenstein and M. Sipser. Go is polynomial-space hard. *J. ACM*, 27:393–401, 1980.

[12] J. Robson. N by N Checkers is exptime complete. *SIAM J. Comput.*, 13:252–267, 1984.

[13] T. J. Schaefer. The complexity of satisfiability problems. In *Proc. 10th Symp. on Theory of Computation*, pages 216–226, 1978.

[14] T. J. Schaefer. On the complexity of some two-person perfect-information games. *J. Comp. Syst. Sc.*, 16:185–225, 1978.

[15] L. J. Stockmeyer and A. K. Chandra. Provably difficult combinatorial games. *SIAM J. Comput.*, 8:151–174, 1979.

[16] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22:215–225, 1975.