Exact Algorithms for Kayles

H. L. Bodlaender[1] D. Kratsch[2]

[1]Utrecht University
3508 TB Utrecht
The Netherlands

[2]LITA
Université Paul Verlaine - Metz
France

Workshop on Graph-Theoretic Concepts in Computer Science
Tepla, Czech Republic, june 21-24, 2011

I. Introduction

KAYLES : The Game

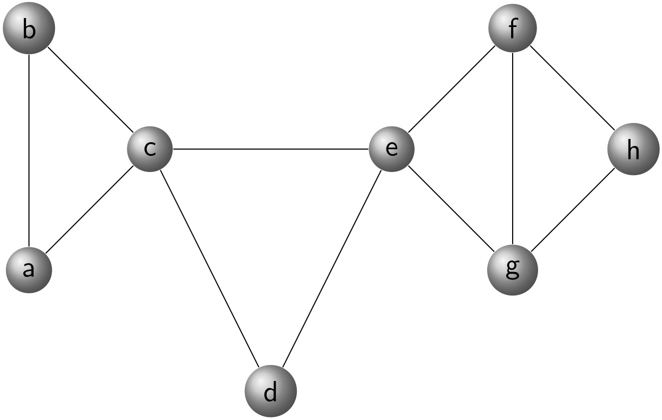
Rules

- two player game played on an undirected graph $G = (V, E)$
- players select alternatingly a vertex from G
- a player may never choose a vertex that is adjacent or equal to an already chosen vertex
- the last player that can select a vertex wins the game

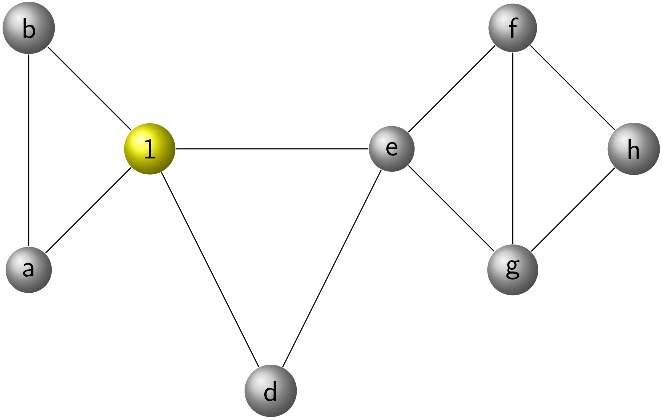
Alternative descriptions

- players build together an independent set in G and the player turning the independent set into a maximal independent set wins the game
- the chosen vertex and its neighbors are removed and a player wins when his move empties the graph.

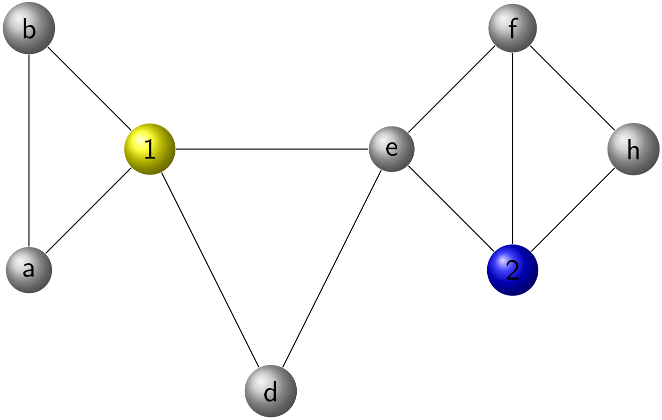
Playing KAYLES I



Playing KAYLES I

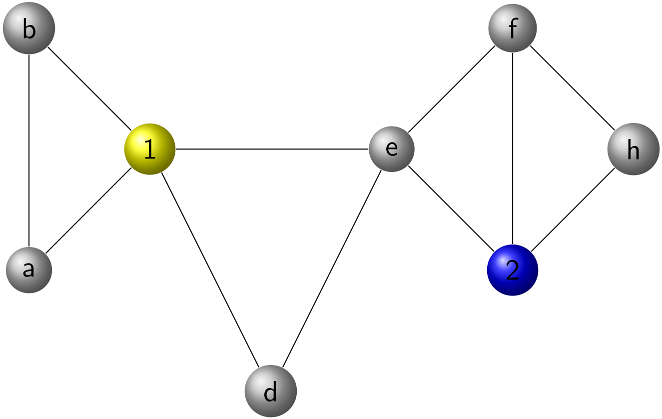


Playing KAYLES I



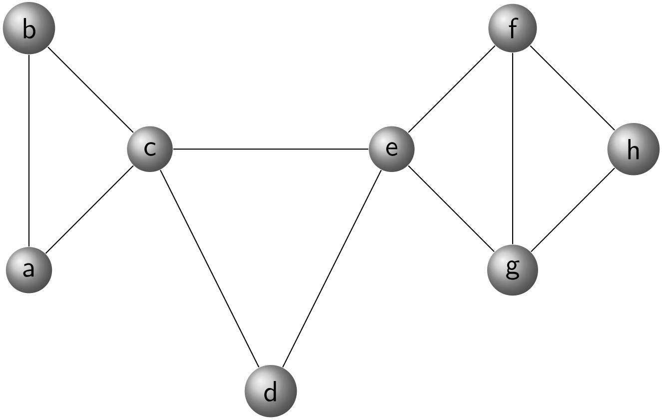
Player 2 wins !

Playing KAYLES I

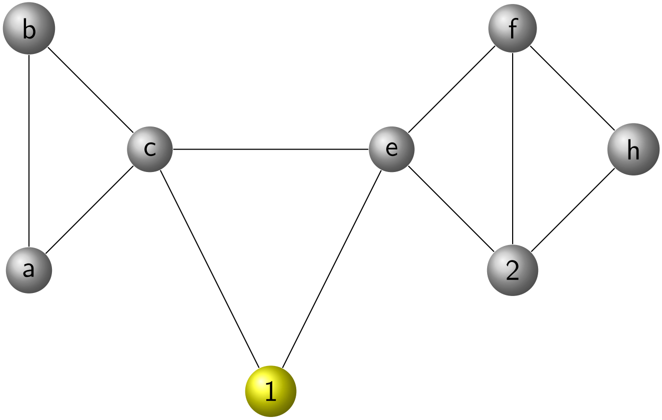


Player 2 wins !

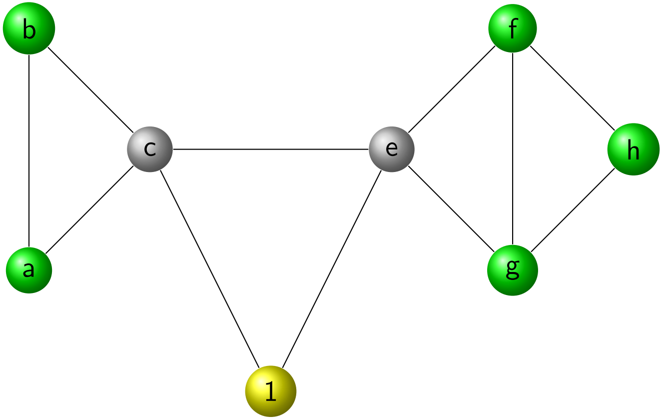
Playing KAYLES II



Playing KAYLES II

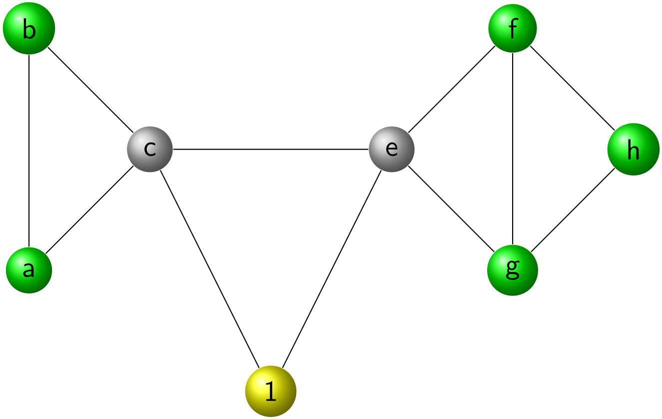


Playing KAYLES II



Player 1 wins !

Playing KAYLES II



Player 1 wins !

KAYLES : The PSPACE-complete Problem

KAYLES

- INPUT : an undirected graph $G = (V, E)$
- QUESTION : Has player 1 a winning strategy when the game is played on graph G ?

[Schaefer 1978]

The problem KAYLES is PSPACE-complet.

Known Results

Polynomial Time Algorithms

- on graphs of bounded asteroidal number (including AT-free graphs, interval graphs, cocomparability graphs and cographs) [Bodlaender Kratsch 2002]
- on stars of bounded degree [Fleischer Trippen 2004]
- on paths [Guignard Sopena 2009] (variants of KAYLES)

Algorithm solving KAYLES in $O^*(2^n)$ time :

“tabulate for each induced subgraph of G which player has a winning strategy from that position’ ’

Can we say more about the complexity of KAYLES ?

NP vs. PSPACE

- NP $\subseteq$ PSPACE
- every NP-complete problem can be solved in polynomial space
- polynomial-time hierarchy

What is the Time Complexity of KAYLES ?

- Could it be that there is a PSPACE-complete problem X and a NP-complete problem Y such that X can be solved faster than Y ?
- Could it be that there is an algorithm solving KAYLES of a running time faster than the best known one for SAT ?
- Could it be that the best algorithm for KAYLES is faster than the best algorithm (not yet known) for SAT ?

II. Our Results

K-sets

Let $S \subseteq V$. We denote by $N[S]$ the set of all vertices v satisfying

- either $v \in S$
- or v has a neighbor in S.

Fundamental Notion
A nonempty set of vertices $W \subseteq V$ is a K-set in a graph $G = (V, E)$, if $G[W]$ is connected and there exists an independent set X such that $W = V - N[X]$.

Hence a K-set is a connected component of the graph that remains after the players selected an independent set X; thereby removing each chosen vertex and its neighbors.

Maximum Number of K-sets : Upper Bounds

Upper bound on graphs

- A graph G on n vertices has (at most) $O(1.6052^n)$ K-sets.

Upper Bound on trees

- A tree on n nodes has at most $n \cdot 3^{n/3}$ K-sets.

Exact algorithms

The running time of our exact algorithm solving KAYLES is bounded by a polynomial factor times the number of K-sets in G.

KAYLES

- KAYLES can be solved in time $O(1.6052^n)$ for graphs on n vertices.

KAYLES on trees

- KAYLES can be solved in time $O(1.4423^n)$ for trees on n nodes.

Maximum Number of K-sets : Lower Bounds

Lower bound on graphs

- For any $t \geq 1$, there is a graph on $n = 3t$ vertices with at least $3^{n/3}$ different K-sets.

Lower Bound on trees

- For any $t \geq 1$, there is a tree on $n = 3t + 1$ nodes with at least $3^{(n-1)/3}$ different K-sets.

Consequence

- large gap between upper bound $O(1.6052^n)$ and lower bound $\Omega(1.4422^n)$ for graphs
- tight bounds (up to polynomial factor) $O(1.4423^n)$ resp. $\Omega(1.4422^n)$ for trees

III. Sprague-Grundy Theory

Sprague-Grundy Theory I

For a good introduction to Sprague-Grundy theory we refer to :

Berlekamp, E. R., Conway, J. H., and Guy, R. K. *Winning Ways for your mathematical plays, Vol. 1 : Games in General*. Academic Press, 1982.

Conway, J. H. *On Numbers and Games*. Academic Press, 1976.

Sprague-Grundy Theory II

Sprague-Grundy theory can be applied to KAYLES since ...

... KAYLES is ...

- impartial
- deterministic
- finite
- full-information
- two player game
- ‘last player wins rule’

Sprague-Grundy Theory III

A nimber is an integer belonging to $\mathbf{N} = \{0, 1, 2, \ldots\}$. For a finite set of nimbers $S \subseteq \mathbf{N}$, define the minimum excluded nimber of S as $mex(S) = \min\{i \in \mathbf{N} \mid i \notin S\}$.

Assigning nimbers to positions (of KAYLES)

- no move possible (and player who must move looses) in position p : $nb(p) := 0$
- otherwise $nb(p)$ is the minimum excluded nimber of the set of nimbers of positions that can be reached in one move

THEOREM [Berlekamp et al. 1982, Conway 1976]

There is a winning strategy for player 1 from a position, if and only if the nimber of that position is at least 1.

Sprague-Grundy Theory IV

The sum of two games (of KAYLES) $\mathcal{G}_1$ and $\mathcal{G}_2$ denoted $\mathcal{G}_1 + \mathcal{G}_2$ is the game where a move consists of choosing $\mathcal{G}_1$ or $\mathcal{G}_2$ and then making a move in that game. A player that cannot make a move in $\mathcal{G}_1$ nor in $\mathcal{G}_2$ looses the game $\mathcal{G}_1 + \mathcal{G}_2$.

Binary XOR operation is denoted by $\oplus$, i.e., for nimbers i_1, i_2, $i_1 \oplus i_2 = \sum\{2^j \mid (\lfloor i_1/2^j \rfloor \text{ is odd}) \Leftrightarrow (\lfloor i_2/2^j \rfloor \text{ is even})\}$.

THEOREM [Berlekamp et al. 1982, Conway 1976]

Let p_1 be a position in $\mathcal{G}_1$, p_2 a position in $\mathcal{G}_2$. The nimber of position (p_1, p_2) in $\mathcal{G}_1 + \mathcal{G}_2$ equals $nb((p_1, p_2)) = nb(p_1) \oplus nb(p_2)$.

Consequence for KAYLES : For disjoint graphs G_1 and G_2 (possibly disconnected) : $nb(G_1 \cup G_2) = nb(G_1) \oplus nb(G_2)$

IV. An Upper Bound on the Number of K-sets in Graphs

Combinatorial Upper Bound

THEOREM :

The number of K-sets in an n-vertex graph is $O(1.6052^n)$.

- Main fact in time analysis of exact algorithm for KAYLES

How to establish the upper bound ?

- branching algorithm generating all K-sets (possibly also non connected sets) of input graph
- upper bound the number of leaves of the search tree (corresponding to an execution)
- using linear recurrences, branching vectors, a measure etc. to analyse branching rules
- tailoring algorithm (branching and reduction rules) and measure to achieve best possible upper bound

Approach

non-trivial K-set

- A K-set is nontrivial, if it has at least three vertices ; otherwise we call it trivial.
- number of trivial K-sets is at most $|V| + |E|$

via branching

- construct an independent set X
- construct a non-trivial K-set W containing v_0
- select a vertex to be in X
- forbid a vertex to be in X
- remove a vertex from the graph

Colors and weights

Four types of vertices

- **White** or free vertices.
  Weight 1
- **Red** vertices. Not to be selected into the independent set X. Might be removed later.
  Weight $\alpha = 0.5685$
- **Green** vertices. Will never be removed. Belongs to (final) W.
  Weight 0
- **Removed vertices**. Not existing anymore. Removed by being selected into X or by being neighbor of a vertex in X.

Measure

- an instance is a (typically connected) induced subgraph G' of the input graph G with vertices of color white, red or green
- the measure $\mu(G')$ is the total weight of all vertices

Reduction Rules

START : Fix an arbitrary vertex v_0 color it green.
GOAL : Generate all non-trivial K-sets containing v_0

- **Rule 1** : If a red vertex v has no white neighbor, we can color it green. This is valid, as we can no longer place a neighbor of v in X.
- **Rule 2** : If a green vertex v has a white neighbor w, we can color w red. This is valid, as placing w in X would remove v, which we are not allowed by the green color of v.
- **Rule 3** : If G has more than one connected component, then remove all vertices from components that do not contain the green vertex v_0.

Branching Rules

Main type of branching rule : vertex branching

- $v \in V$ a white vertex.
- **Case 1** : v is selected into X. Remove $N[v]$. Measure decreases by the total weight of all white and red vertices in the closed neighborhood of v.
- **Case 2** : v is discarded from X. Color v red. Measure decreases by $1 - \alpha$.

Case 1 and 2

Case 1 : There is a white vertex *v* with at least three white neighbors.

- vertex branch on *v*
- decrease of measure at least 4 for select *v*
- decrease of measure $1 - \alpha$ for discard *v*
- branching vector $(4, 1 - \alpha)$

Case 2 : There is a white vertex *v* with two white neighbors and at least one red neighbor.

- vertex branch on *v*
- decrease of measure at least $3 + \alpha$ for select *v*
- decrease of measure $1 - \alpha$ for discard *v*
- branching vector $(3 + \alpha, 1 - \alpha)$

V. The Exact Algorithm

Recursive algorithm using memorization

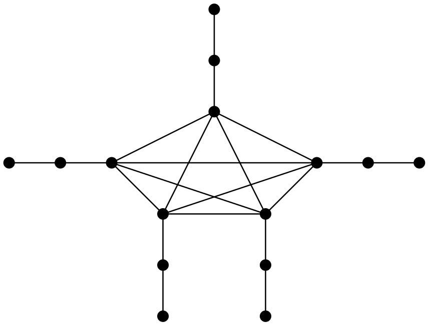
Procedure `compute_nimber`**(G[W]).**

if *nb*(W) *already computed* **then**
  return *nb*(W)
else
  $M := \emptyset$;
  for *all* $w \in W$ **do**
    let $Z_1, Z_2, \ldots, Z_r$ ($r \geq 1$) be the components of $G - N[w]$;
    $nim := 0$;
    for $i \leftarrow 1$ **to** r **do**
      $nim := nim \oplus$ `compute_nimber`$(G[Z_i])$;
      $M := M \cup \{nim\}$
    $answer :=$ `mex`(M);
  $nb(W) := answer$;
  return *answer*

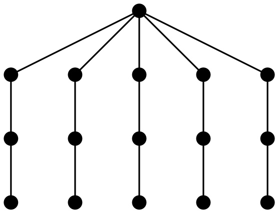
Algorithm calls procedure `compute_nimber` with input $G = (V, E)$
Running time : number of K-sets times a polynomial in n

VI. Lower Bounds

Example a lower bound graph with $t = 5$



Example a lower bound tree with $t = 5$



VII. Conclusions

Summary of results

- exact algorithm solving PSPACE-complete problem KAYLES
- introduced notion of K-sets
- upper and lower bounds for maximum number of K-sets in n-vertex graphs
- tight bound for the maximum number of K-sets in n-node trees

Open Questions

- complexity of KAYLES on trees longstanding open problem
- Could there be a subexponential algorithm for KAYLES on trees, say of form $O(c^{\sqrt{n}})$?
- Is there a polynomial space algorithm solving KAYLES with a running time of $O^*(2^n)$?
- Find an algorithm with a running time $O^*(c^n)$ with $c < 2$ for other PSPACE-complete problems, e.g. combinatorial games or even QUANTIFIED 3-SATISFIABILITY.