Reconfiguration and Combinatorial Games

Marc Heinrich

under the supervision of:

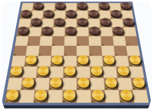
Eric Duchêne	Director
Sylvain Gravier	Co-director
Nicolas Bousquet	Co-supervisor

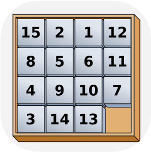
July 9th, 2019

Games

- No hidden information.
- No randomness.
- 1 or 2 players.



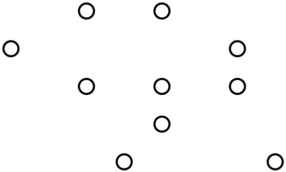




Definition

Graph:

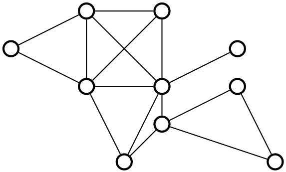
- A set of vertices.



Definition

Graph:

- A set of vertices.
- A set of edges: links between the vertices.

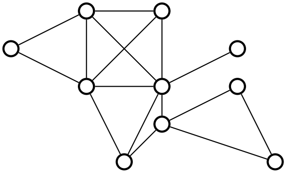


Graphs

Definition

Graph:

- A set of vertices.
- A set of edges: links between the vertices.



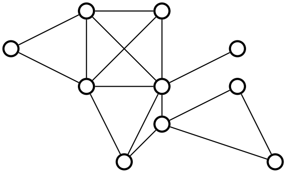
- Some games are played on a graph.

Graphs

Definition

Graph:

- A set of vertices.
- A set of edges: links between the vertices.



- Some games are played on a graph.

- Games can be also be represented by a graph.

General Organisation

Reconfiguration problems

(1-player games)

Reconfiguration of graph colourings

Reconfiguration of perfect matchings

General Organisation

Reconfiguration problems

(1-player games)

Reconfiguration of graph colourings

Reconfiguration of perfect matchings

Algorithmic Applications

Sampling Colourings

Online colouring

General Organisation

Reconfiguration problems

(1-player games)

- Reconfiguration of graph colourings
- Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

- Partizan Subtraction Games
- Rules composition

Algorithmic Applications

- Sampling Colourings
- Online colouring

General Organisation

Reconfiguration problems

(1-player games)

Reconfiguration of graph colourings

Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

Partizan Subtraction Games

Rules composition

Algorithmic Applications

Sampling Colourings

Online colouring

General Organisation

Reconfiguration problems

(1-player games)

- Reconfiguration of graph colourings
- Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

- Partizan Subtraction Games
- Rules composition

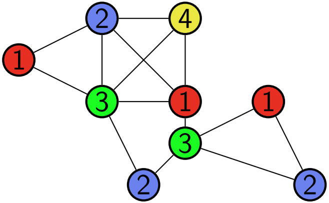
Algorithmic Applications

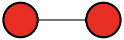
- Sampling Colourings
- Online colouring

Colouring reconfiguration

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.

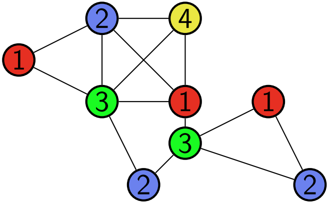


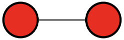
forbidden: 

Graph colouring

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.



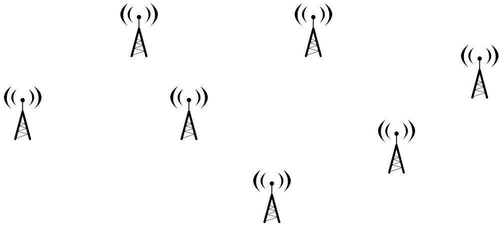
forbidden: 

Applications:

- Frequency assignment problem

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.



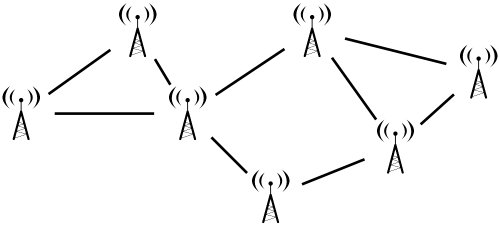
Applications:

- Frequency assignment problem

Graph colouring

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.



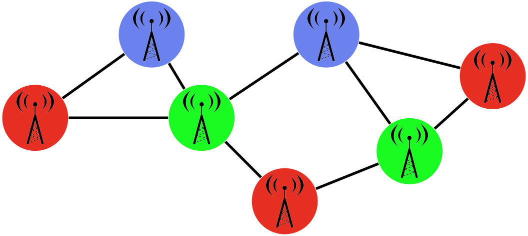
Applications:

- Frequency assignment problem

Graph colouring

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.



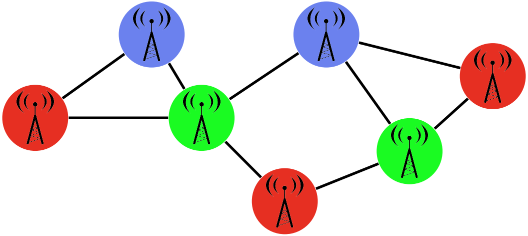
Applications:

- Frequency assignment problem

Graph colouring

Definition

k-colouring: assignment of colours between 1 and k to the vertices of a graph such that there is no monochromatic edge.



Applications:

- Frequency assignment problem
- Scheduling
- Statistical Physics.

Reconfiguration problems

Definition

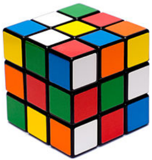
Reconfiguration problem: Finding transformations between solutions of a given problem.

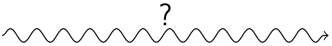
Reconfiguration problems

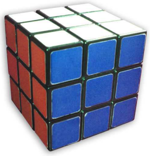
Definition

Reconfiguration problem: Finding transformations between solutions of a given problem.

- Examples:
  - Rubik's cube



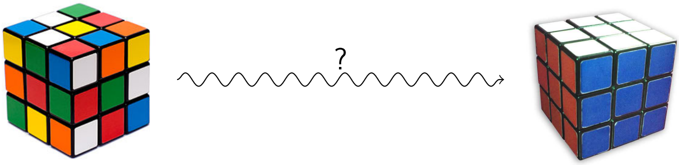




Definition

Reconfiguration problem: Finding transformations between solutions of a given problem.

- Examples:
  - Rubik's cube, 15 puzzle, Rush-Hour (combinatorial puzzles)

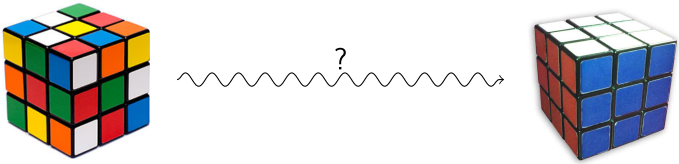


Reconfiguration problems

Definition

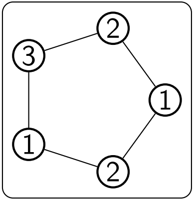
Reconfiguration problem: Finding transformations between solutions of a given problem.

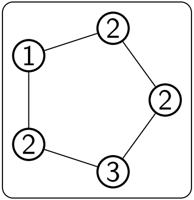
- Examples:
  - Rubik's cube, 15 puzzle, Rush-Hour (combinatorial puzzles)



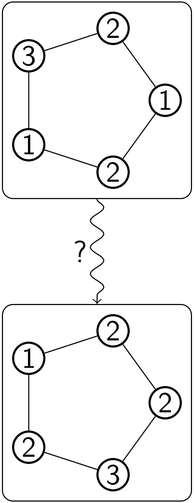
  - Colouring reconfiguration.

Reconfiguration

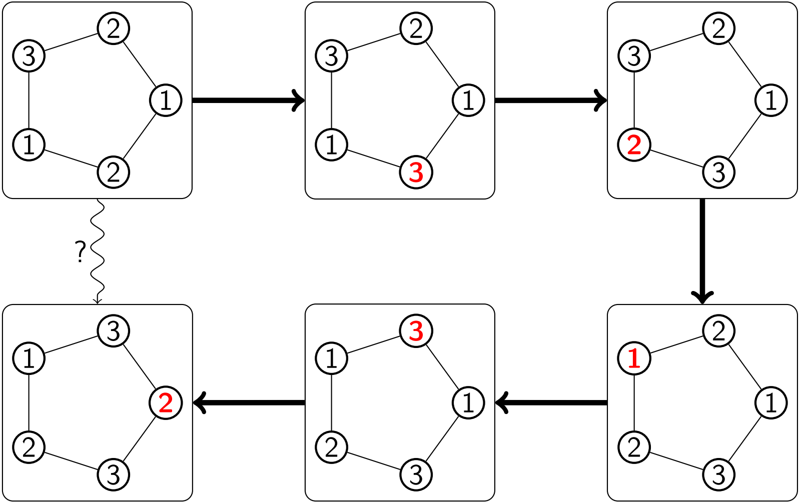




Reconfiguration



Reconfiguration



Definition

Reconfiguration graph:

- Vertices: all the possible colourings of a given graph.

Reconfiguration graph

Definition

Reconfiguration graph:

- Vertices: all the possible colourings of a given graph.
- Edges: Transformations between the colourings.

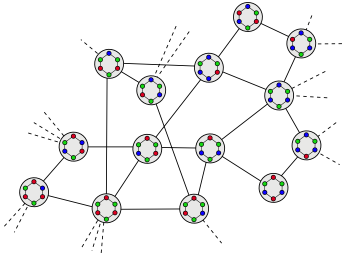
Reconfiguration graph

Definition

Reconfiguration graph:

- Vertices: all the possible colourings of a given graph.
- Edges: Transformations between the colourings.

$\mathcal{G}(k, G)$: reconfiguration graph of k-colourings of G.



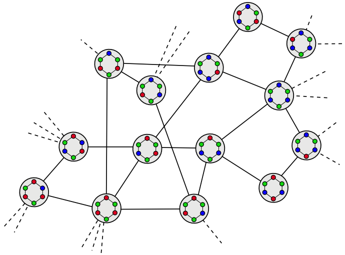
Reconfiguration graph

Definition

Reconfiguration graph:

- Vertices: all the possible colourings of a given graph.
- Edges: Transformations between the colourings.

$\mathcal{G}(k, G)$: reconfiguration graph of k-colourings of G.



Combinatorial puzzles (1-player games) can also be defined with this formalism.

Motivations

- Adapt a solution already in place
  $\rightarrow$ Operational research

Motivations

- Adapt a solution already in place
  $\rightarrow$ Operational research

- Understand the performance of local search algorithms.
  $\rightarrow$ Analysis of algorithms

Motivations

- Adapt a solution already in place
  $\rightarrow$ Operational research
- Understand the performance of local search algorithms.
  $\rightarrow$ Analysis of algorithms
- Enumeration problems

Motivations

- Adapt a solution already in place
  $\rightarrow$ Operational research

- Understand the performance of local search algorithms.
  $\rightarrow$ Analysis of algorithms

- Enumeration problems

- Generating random colourings
  $\rightarrow$ Statistical physics

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?

- CONNECTIVITY: can we transform any α into any β?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?
- CONNECTIVITY: can we transform any α into any β?
  $\longrightarrow$ Is $\mathcal{G}(k, G)$ connected?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?

- CONNECTIVITY: can we transform any α into any β?
  $\longrightarrow$ Is $\mathcal{G}(k, G)$ connected?

- How many steps are required (in the worst case)?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?
- CONNECTIVITY: can we transform any α into any β?
  $\longrightarrow$ Is $\mathcal{G}(k, G)$ connected?
- How many steps are required (in the worst case)?
  $\longrightarrow$ What is the diameter of $\mathcal{G}(k, G)$?

Reconfiguration problems

- REACHABILITY: can we transform a colouring α into another β?
  $\longrightarrow$ Is there a path from α to β in $\mathcal{G}(k, G)$?

- CONNECTIVITY: can we transform any α into any β?
  $\longrightarrow$ Is $\mathcal{G}(k, G)$ connected?

- How many steps are required (in the worst case)?
  $\longrightarrow$ **What is the diameter of $\mathcal{G}(k, G)$?**

Degeneracy

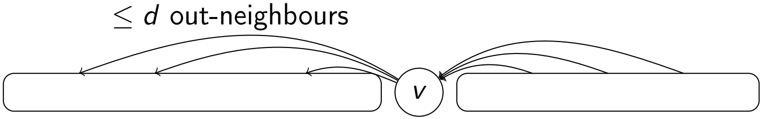
Definition

A graph is *d*-degenerate if there is an ordering such that each vertex has at most *d* previous neighbours.

Degeneracy

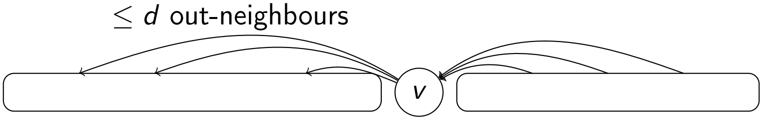
Definition

A graph is d-degenerate if there is an ordering such that each vertex has at most d previous neighbours.

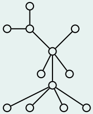


Definition

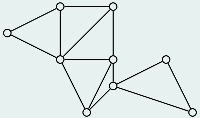
A graph is d-degenerate if there is an ordering such that each vertex has at most d previous neighbours.



- Trees are 1-degenerate



- Planar graphs are 5-degenerate,



Lemma

If G is d-degenerate and $k \geq d + 2$, then $\mathcal{G}(k, G)$ is connected.

Lemma

If G is d-degenerate and $k \geq d+2$, then $\mathcal{G}(k, G)$ is connected.
$\longrightarrow$ Diameter at most 2^n.

Lemma

If G is d-degenerate and $k \geq d+2$, then $\mathcal{G}(k, G)$ is connected.
$\longrightarrow$ Diameter at most 2^n.

Cereceda's Conjecture (2007)

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d+2$.

Lemma

If G is d-degenerate and $k \geq d+2$, then $\mathcal{G}(k, G)$ is connected.
$\longrightarrow$ Diameter at most 2^n.

Cereceda's Conjecture (2007)

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d+2$.

Even a polynomial upper bound is open.

Known Results

Cereceda's Conjecture

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d + 2$.

Known Results

Cereceda's Conjecture

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d + 2$.

number of colours	Diameter	Reference
$k \geq \Delta + 2$	$O(\Delta n)$	[Cer07]
$k \geq 2d + 1$	$O(n^2)$	[Cer07]
$k \geq 2d + 2$	$O(dn)$	[BP16]

Known Results

Cereceda's Conjecture

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d + 2$.

number of colours	Diameter	Reference
$k \geq \Delta + 2$	$O(\Delta n)$	[Cer07]
$k \geq 2d + 1$	$O(n^2)$	[Cer07]
$k \geq 2d + 2$	$O(dn)$	[BP16]

The conjecture holds if we replace the degeneracy by:

- $\text{mad}(G)$ the maximum average degree [BP16, Feg19],
- $\text{tw}(G)$ the treewidth [BB14, Feg19],
- $\chi_g(G)$ the Grundy chromatic number [BB14].

Known Results

Cereceda's Conjecture

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d + 2$.

number of colours	Diameter	Reference
$k \geq \Delta + 2$	$O(\Delta n)$	[Cer07]
$k \geq 2d + 1$	$O(n^2)$	[Cer07]
$k \geq 2d + 2$	$O(dn)$	[BP16]

The conjecture holds if we replace the degeneracy by:

- $\text{mad}(G)$ the maximum average degree [BP16, Feg19],
- $\text{tw}(G)$ the treewidth [BB14, Feg19], $\Rightarrow$ conjecture true for $d = 1$
- $\chi_g(G)$ the Grundy chromatic number [BB14].

Known Results

Cereceda's Conjecture

The diameter of $\mathcal{G}(k, G)$ is $O(n^2)$ if $k \geq d + 2$.

number of colours	Diameter	Reference
$k \geq \Delta + 2$	$O(\Delta n)$	[Cer07]
$k \geq 2d + 1$	$O(n^2)$	[Cer07]
$k \geq 2d + 2$	$O(dn)$	[BP16]

Planar graphs:

- diameter $O(n^2)$ if $k \geq 10$ [Feg19],
- diameter poly(n) if $k \geq 8$ [BP15, Feg19],
- diameter $2^{\sqrt{n}}$ if $k \geq 7$ [EF18].

Results [BH19]

number of colours	Diameter
$k \geq d+2$	$O(n^{d+1})$
$k \geq (1+\varepsilon)(d+1)$	$O(n^{\frac{1}{\varepsilon}})$
$k \geq \frac{3}{2}(d+1)$	$O(n^2)$

[Bousquet, Heinrich, 2019]

Results [BH19]

number of colours	Diameter
$k \geq d+2$	$O(n^{d+1})$
$k \geq (1+\varepsilon)(d+1)$	$O(n^{\frac{1}{\varepsilon}})$
$k \geq \frac{3}{2}(d+1)$	$O(n^2)$

Planar graphs:

- diameter $O(n^2)$ if $k \geq 9$
- diameter $O(n^6)$ if $k \geq 7$

[Bousquet, Heinrich, 2019]

Results [BH19]

number of colours	Diameter
$k \geq d+2$	$O(n^{d+1})$
$k \geq (1+\varepsilon)(d+1)$	$O(n^{\frac{1}{\varepsilon}})$
$k \geq \frac{3}{2}(d+1)$	$O(n^2)$

Planar graphs:

- diameter $O(n^2)$ if $k \geq 9$
  $\Rightarrow$ improved by Feghali to $O(n \log^c n)$ if $k \geq 8$
- diameter $O(n^6)$ if $k \geq 7$

[Bousquet, Heinrich, 2019]

Results [BH19]

number of colours	Diameter
$\mathbf{k \geq d + 2}$	$\mathbf{O(n^{d+1})}$
$k \geq (1+\varepsilon)(d+1)$	$O(n^{\frac{1}{\varepsilon}})$
$k \geq \frac{3}{2}(d+1)$	$O(n^2)$

Planar graphs:

- diameter $O(n^2)$ if $k \geq 9$
  $\Rightarrow$ improved by Feghali to $O(n \log^c n)$ if $k \geq 8$
- diameter $O(n^6)$ if $k \geq 7$

[Bousquet, Heinrich, 2019]

Idea

Proceed recursively on d.

Idea

Proceed recursively on d.

- G a d-degenerate graph.
- Two colourings α and β.

Idea

Proceed recursively on d.

- G a d-degenerate graph.
- Two colourings α and β.

- Allowed $O(n)$ recursive calls to recolour a $(d-1)$-degenerate graph.
- $F(n,d) \approx n \cdot F(n,d-1)$

Idea

Proceed recursively on d.

- G a d-degenerate graph.
- Two colourings α and β.

- Allowed $O(n)$ recursive calls to recolour a $(d-1)$-degenerate graph.
- $F(n,d) \approx n \cdot F(n,d-1)$
- Remove all the vertices coloured c.

Full colours

Definition

A colour c is **full** if for all v, either:

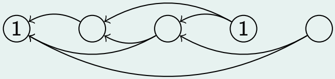
- v is coloured c.
- v has an out-neighbour coloured c.

Full colours

Definition

A colour c is **full** if for all v, either:

- v is coloured c.
- v has an out-neighbour coloured c.

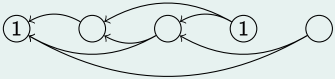


Full colours

Definition

A colour c is **full** if for all v, either:

- v is coloured c.
- v has an out-neighbour coloured c.



Remark

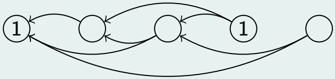
- Removing a full colour decreases d by 1:
  - Remove all vertices with this colour.
  - Forbid this colour for the other vertices.

Full colours

Definition

A colour c is **full** if for all v, either:

- v is coloured c.
- v has an out-neighbour coloured c.



Remark

- Removing a full colour decreases d by 1:
  - Remove all vertices with this colour.
  - Forbid this colour for the other vertices.

Questions

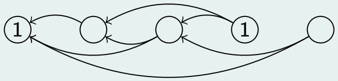
- What to do with a full colour?
- How to create a full colour?

Full colours

Definition

A colour c is **full** if for all v, either:

- v is coloured c.
- v has an out-neighbour coloured c.



Remark

- Removing a full colour decreases d by 1:
  - Remove all vertices with this colour.
  - Forbid this colour for the other vertices.

Questions

- What to do with a full colour? Easy
- How to create a full colour?

How to create a full colour?

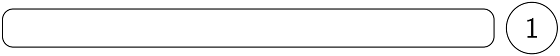
Idea

Proceed iteratively (using the degeneracy ordering).

How to create a full colour?

Idea

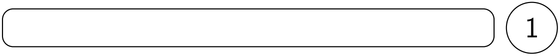
Proceed iteratively (using the degeneracy ordering).



How to create a full colour?

Idea

Proceed iteratively (using the degeneracy ordering).

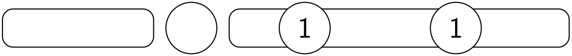


Full colour: 1.

How to create a full colour?

Idea

Proceed iteratively (using the degeneracy ordering).

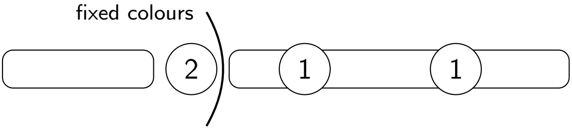


Full colour: 1.

How to create a full colour?

Idea

Proceed iteratively (using the degeneracy ordering).

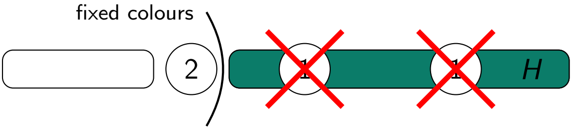


Full colour: 1.

How to create a full colour?

Idea

Proceed iteratively (using the degeneracy ordering).



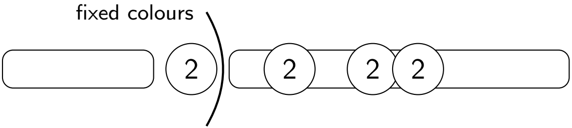
Full colour: 1.

- Remove colour 2 from H.

How to create a full colour?

Idea

Proceed iteratively (using the degeneracy ordering).



Full colour: 2.

- Remove colour 2 from H.
- Greedily recolour vertices with 2.

Reconfiguration problems

(1-player games)

Reconfiguration of graph colourings

Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

Partizan Subtraction Games

Rules composition

Algorithmic Applications

Sampling Colourings

Online colouring

Definition

Glauber Dynamics:

Definition

Glauber Dynamics:

1. Select a random vertex.

Definition

Glauber Dynamics:

1. Select a random vertex.
2. Recolour it with a random colour if possible.

Definition

Glauber Dynamics:

1. Select a random vertex.
2. Recolour it with a random colour if possible.
3. Repeat.

Definition

Glauber Dynamics:

1. Select a random vertex.
2. Recolour it with a random colour if possible.
3. Repeat.

- If the reconfiguration graph is connected, it produces an almost uniform random colouring.

Generating random colourings

Definition

Glauber Dynamics:

1. Select a random vertex.
2. Recolour it with a random colour if possible.
3. Repeat.

- If the reconfiguration graph is connected, it produces an almost uniform random colouring.
- Mixing time: how long do we have to repeat this?

Theorem (DHP18)

Glauber dynamics for edge colourings of a tree with $\Delta + 1$ colours mixes in polynomial time.

[Delcourt, Heinrich, Perarnau, 2018], [Poon, 2016], [Vigoda, 2000]

Theorem (DHP18)

Glauber dynamics for edge colourings of a tree with $\Delta + 1$ colours mixes in polynomial time.

- Improves:
  - [Vig00], with $\frac{11}{3}\Delta$ colours
  - [Po16] with 2Δ colours.

[Delcourt, Heinrich, Perarnau, 2018], [Poon, 2016], [Vigoda, 2000]

Theorem (DHP18)

Glauber dynamics for edge colourings of a tree with $\Delta + 1$ colours mixes in polynomial time.

- Improves:
  - [Vig00], with $\frac{11}{3}\Delta$ colours
  - [Po16] with 2Δ colours.
- The number of colours is tight.

[Delcourt, Heinrich, Perarnau, 2018], [Poon, 2016], [Vigoda, 2000]

Theorem (DHP18)

Glauber dynamics for edge colourings of a tree with $\Delta + 1$ colours mixes in polynomial time.

- Improves:
  - [Vig00], with $\frac{11}{3}\Delta$ colours
  - [Po16] with 2Δ colours.
- The number of colours is tight.
- The exponent is independent from Δ.

[Delcourt, Heinrich, Perarnau, 2018], [Poon, 2016], [Vigoda, 2000]

Further work

- Cereceda's conjecture:
  - Planar graphs with 7 colours.
  - Triangle-free planar graphs with 5 colours.

Further work

- Cereceda's conjecture:
  - Planar graphs with 7 colours.
  - Triangle-free planar graphs with 5 colours.

  - Improve the $\frac{3}{2}(d+1)$ bound for the quadratic diameter.

Further work

- Cereceda's conjecture:
  - Planar graphs with 7 colours.
  - Triangle-free planar graphs with 5 colours.

  - Improve the $\frac{3}{2}(d+1)$ bound for the quadratic diameter.
- How many colours to get a linear diameter?

Further work

- Cereceda's conjecture:
  - Planar graphs with 7 colours.
  - Triangle-free planar graphs with 5 colours.

  - Improve the $\frac{3}{2}(d+1)$ bound for the quadratic diameter.
- How many colours to get a linear diameter?
- Lower bounds when $k \geq d+3$.

Further work

- Cereceda's conjecture:
  - Planar graphs with 7 colours.
  - Triangle-free planar graphs with 5 colours.

  - Improve the $\frac{3}{2}(d+1)$ bound for the quadratic diameter.
- How many colours to get a linear diameter?
- Lower bounds when $k \geq d+3$.
- Glauber dynamics with $\Delta + 2$ colours.

Combinatorial Games: Rules Composition

Combinatorial games:

- 2-player games,

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,
- winner determined by the last move:
  - last player wins: **normal convention**,
  - last player loses: **misère convention**,

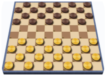
Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,
- winner determined by the last move:
  - last player wins: **normal convention**,
  - last player loses: **misère convention**,
- impartial: same moves for both players.

Combinatorial Games

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,
- winner determined by the last move:
  - last player wins: **normal convention**,
  - last player loses: **misère convention**,
- impartial: same moves for both players.



Combinatorial Games

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,
- winner determined by the last move:
  - last player wins: **normal convention**,
  - last player loses: **misère convention**,
- impartial: same moves for both players.

Combinatorial games:

- 2-player games,
- no randomness,
- no hidden information,
- alternate play,
- winner determined by the last move:
  - last player wins: **normal convention**,
  - last player loses: **misère convention**,
- impartial: same moves for both players.

Example

Definition

Subtraction games, $\textsc{Sub}(S)$:

- Parametrized by a set S: the subtraction set.
- Played on heaps of tokens.
- Each player can remove $x \in S$ tokens from a single heap.

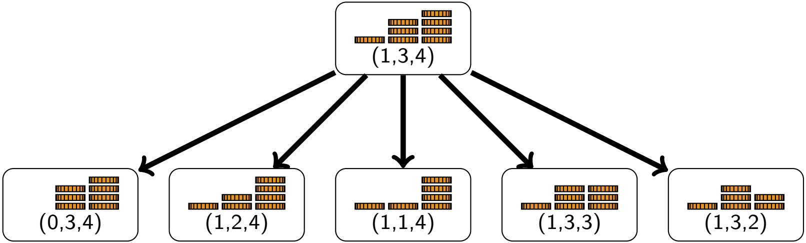
Example

Definition

Subtraction games, $\textsc{Sub}(S)$:

- Parametrized by a set S: the subtraction set.
- Played on heaps of tokens.
- Each player can remove $x \in S$ tokens from a single heap.

$S = \{1, 2\}$



Example

Definition

Subtraction games, $\textsc{Sub}(S)$:

- Parametrized by a set S: the subtraction set.
- Played on heaps of tokens.
- Each player can remove $x \in S$ tokens from a single heap.

Definition

NIM: Players can remove as many tokens as they want ($S = \mathbb{N}^+$).

Two main types of results:

- Study of particular instances of games:

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

Two main types of results:

- Study of particular instances of games:
  - explicit characterization of the outcomes,
  - algorithmic results,
  - hardness proofs.

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

Two main types of results:

- Study of particular instances of games:
  - explicit characterization of the outcomes,
  - algorithmic results,
  - hardness proofs.
- Combinations of games:

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

Two main types of results:

- Study of particular instances of games:
  - explicit characterization of the outcomes,
  - algorithmic results,
  - hardness proofs.

- Combinations of games:
  - combine existing games to create new ones,
  - decompose known games into simpler smaller games.

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

Two main types of results:

- Study of particular instances of games:
    - explicit characterization of the outcomes,
    - algorithmic results,
    - hardness proofs.

- Combinations of games:
    - combine existing games to create new ones,
    - decompose known games into simpler smaller games.

    - Examples: disjunctive sum, sequential compound [SU93], ordinal sum [CNS18].

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

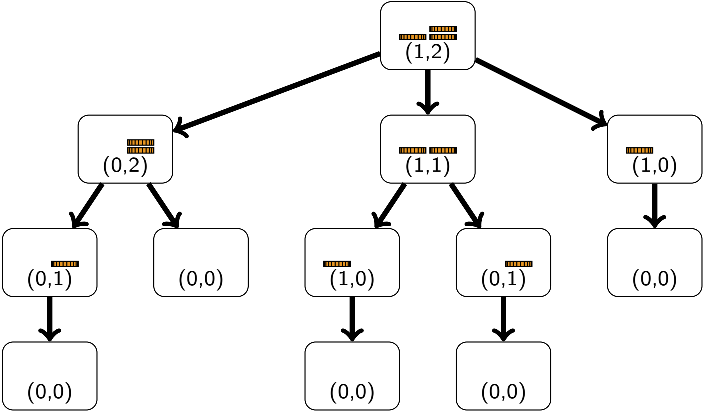
Two main types of results:

- Study of particular instances of games:
  - explicit characterization of the outcomes,
  - algorithmic results,
  - hardness proofs.
- Combinations of games:
  - combine existing games to create new ones,
  - decompose known games into simpler smaller games.
  - Examples: disjunctive sum, sequential compound [SU93], ordinal sum [CNS18].
  - Misère play, pass moves [HN03].

[Stromquist, Ullman, 1993], [Carvalho, Nto, Santos, 2018], [Horrocks, Nowakowski, 2003]

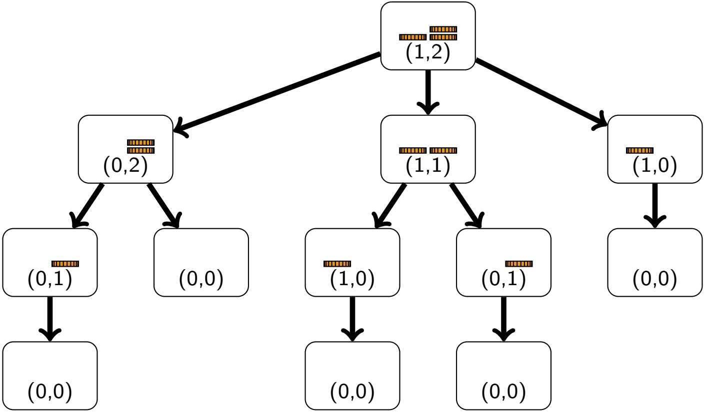
Game tree

$S = \{1, 2\}$



Game tree

$S = \{1, 2\}$



- $\mathfrak{G}$: set of all possible games.

Outcome

Goal: decide who wins under perfect play.

Outcome

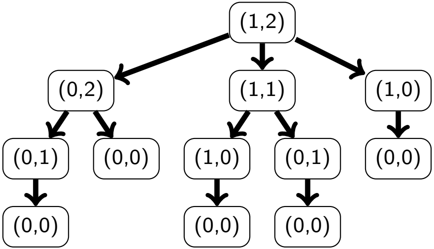
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in red),
- second player wins (in blue).

$o(G)$ denotes the outcome of a game.



Outcome

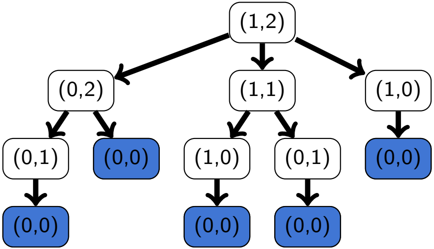
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in red),
- second player wins (in blue).

$o(G)$ denotes the outcome of a game.



Outcome

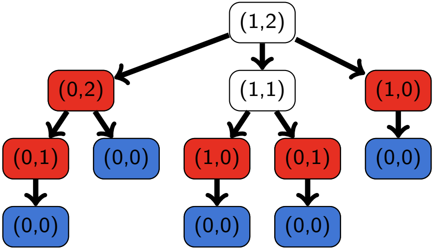
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in **red**),
- second player wins (in **blue**).

$o(G)$ denotes the outcome of a game.



Outcome

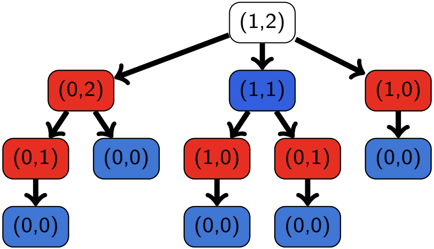
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in **red**),
- second player wins (in **blue**).

$o(G)$ denotes the outcome of a game.



Outcome

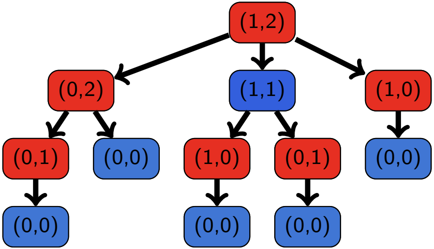
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in **red**),
- second player wins (in **blue**).

$o(G)$ denotes the outcome of a game.



Outcome

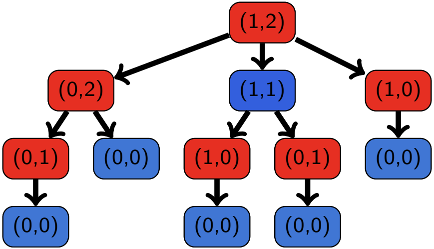
Goal: decide who wins under perfect play.

Definition

Outcome of a game:

- first player wins (in red),
- second player wins (in blue).

$o(G)$ denotes the outcome of a game.



- **Problem:** Computing the whole game tree is usually too expensive

Disjunctive sum

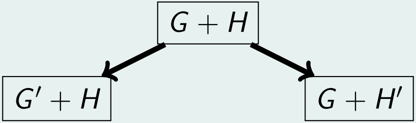
- **Idea:** decompose a game into smaller components.

Disjunctive sum

- **Idea:** decompose a game into smaller components.

Definition

disjunctive sum:

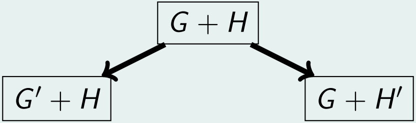


Disjunctive sum

- **Idea:** decompose a game into smaller components.

Definition

disjunctive sum:



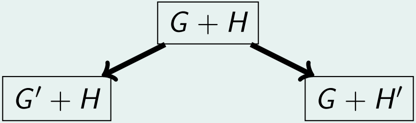
- Example: subtraction games on multiple heaps.

Disjunctive sum

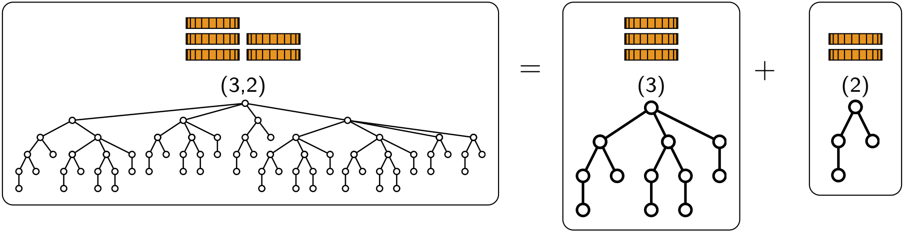
- **Idea:** decompose a game into smaller components.

Definition

disjunctive sum:



- Example: subtraction games on multiple heaps.



Sprague-Grundy Theory

- Goal: determine the outcome of a disjunctive sum by studying the components individually.

Sprague-Grundy Theory

- Goal: determine the outcome of a disjunctive sum by studying the components individually.
- However, $o(G + H)$ cannot be determined by $o(G)$ and $o(H)$.

Sprague-Grundy Theory

- Goal: determine the outcome of a disjunctive sum by studying the components individually.
- However, $o(G+H)$ cannot be determined by $o(G)$ and $o(H)$.

Definition

Grundy value $\mathcal{GV}(G)$: non-negative value attributed to a game.

Computed from the game tree.

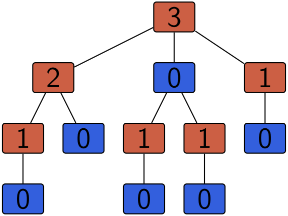
Sprague-Grundy Theory

- Goal: determine the outcome of a disjunctive sum by studying the components individually.
- However, $o(G+H)$ cannot be determined by $o(G)$ and $o(H)$.

Definition

Grundy value $\mathcal{GV}(G)$: non-negative value attributed to a game.

Computed from the game tree.



- Goal: determine the outcome of a disjunctive sum by studying the components individually.
- However, $o(G+H)$ cannot be determined by $o(G)$ and $o(H)$.

Definition

Grundy value $\mathcal{GV}(G)$: non-negative value attributed to a game.

Computed from the game tree.

- $\mathcal{GV}(G) = 0 \Leftrightarrow$ second player wins on G,

Sprague-Grundy Theory

- Goal: determine the outcome of a disjunctive sum by studying the components individually.
- However, $o(G+H)$ cannot be determined by $o(G)$ and $o(H)$.

Definition

Grundy value $\mathcal{GV}(G)$: non-negative value attributed to a game.

Computed from the game tree.

- $\mathcal{GV}(G) = 0 \Leftrightarrow$ second player wins on G,
- $\mathcal{GV}(G+H) = \mathcal{GV}(G) \oplus \mathcal{GV}(H)$ where $\oplus$ is the bit-wise XOR.

Grundy value

Definition

Game equivalence:

$$G \equiv H \Leftrightarrow \forall X \in \mathfrak{G}, o(G+X) = o(H+X)$$

Grundy value

Definition

Game equivalence:

$$G \equiv H \Leftrightarrow \forall X \in \mathfrak{G}, o(G+X) = o(H+X)$$

Theorem (Sprague, Grundy, 1936)

Every game G is equivalent to a single NIM *heap of size $\mathcal{GV}(G)$.*

Grundy value

Definition

Game equivalence:

$$G \equiv H \Leftrightarrow \forall X \in \mathfrak{G}, o(G+X) = o(H+X)$$

Theorem (Sprague, Grundy, 1936)

Every game G is equivalent to a single Nim *heap of size $\mathcal{GV}(G)$.*

- Not true for misère.

Example

Subtraction games:

Example

Subtraction games:

Theorem (Folklore)

The sequence of Grundy values for finite subtraction games on one heap is ultimately periodic.

Subtraction games:

Theorem (Folklore)

The sequence of Grundy values for finite subtraction games on one heap is ultimately periodic.

Examples

- $S = \{1, 2\}$, $\mathcal{GV}$-sequence: $0, 1, 2, 0, 1, 2, 0, 1, 2, \ldots$

Example

Subtraction games:

Theorem (Folklore)

The sequence of Grundy values for finite subtraction games on one heap is ultimately periodic.

Examples

- $S = \{1, 2\}$, $\mathcal{GV}$-sequence: $0, 1, 2, 0, 1, 2, 0, 1, 2, \ldots$
- $S = \{2, 4, 5, 8\}$, period: 17, preperiod: 12.

Example

Subtraction games:

Theorem (Folklore)

The sequence of Grundy values for finite subtraction games on one heap is ultimately periodic.

Examples

- $S = \{1, 2\}$, $\mathcal{GV}$-sequence: $0, 1, 2, 0, 1, 2, 0, 1, 2, \dots$
- $S = \{2, 4, 5, 8\}$, period: 17, preperiod: 12.

Corollary

The outcome of a position for a given subtraction game can be computed in polynomial time.

Rules compound

- Combine rulesets instead of games.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Rules compound

- Combine rulesets instead of games.

Definition [DHLP18]

$\mathcal{R}_1$ and $\mathcal{R}_2$ two rulesets. Push-compound $\mathcal{R}_1 \circledcirc \mathcal{R}_2$:

- start by playing according to $\mathcal{R}_1$,
- during the game, one of the player can change the rules to $\mathcal{R}_2$,
- changing the rules counts as a move.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Rules compound

- Combine rulesets instead of games.

Definition [DHLP18]

$\mathcal{R}_1$ and $\mathcal{R}_2$ two rulesets. Push-compound $\mathcal{R}_1 \circledcirc \mathcal{R}_2$:

- start by playing according to $\mathcal{R}_1$,
- during the game, one of the player can change the rules to $\mathcal{R}_2$,
- changing the rules counts as a move.

- generalisation of pass moves,
- variations of classical games.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Example

Push-subtraction games:

- Rulesets of the form $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$
  S_1 and S_2 two subtraction sets.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Example

Push-subtraction games:

- Rulesets of the form $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$
  S_1 and S_2 two subtraction sets.

Theorem (DHLP18)

Given S_1 and S_2 two finite sets, then $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$ played on a single heap has an ultimately periodic outcome sequence.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Example

Push-subtraction games:

- Rulesets of the form $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$
  S_1 and S_2 two subtraction sets.

Theorem (DHLP18)

Given S_1 and S_2 two finite sets, then $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$ played on a single heap has an ultimately periodic outcome sequence.

Question: What about multiple heaps?

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Example

Push-subtraction games:

- Rulesets of the form $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$
  S_1 and S_2 two subtraction sets.

Theorem (DHLP18)

Given S_1 and S_2 two finite sets, then $\textsc{Sub}(S_1) \circledcirc \textsc{Sub}(S_2)$ played on a single heap has an ultimately periodic outcome sequence.

Question: What about multiple heaps?

- This is not a disjunctive sum.
- Pushing the button changes the rules in both components.

[Duchêne, Heinrich, Larsson, Parreau, 2018]

Example

Example

Let $\mathcal{R} = \textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$. The second player has a winning strategy on $(n_1, \ldots, n_k)$ if and only if:

$$\bigoplus_{i=1}^{k} (n_i \bmod 4) = 1$$

Example

Example

Let $\mathcal{R} = \textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$. The second player has a winning strategy on $(n_1, \ldots, n_k)$ if and only if:

$$\bigoplus_{i=1}^{k} (n_i \bmod 4) = 1$$

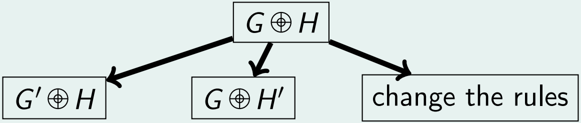
- Similar to the Grundy values.

Almost disjunctive sum

Almost disjunctive sum

Definition

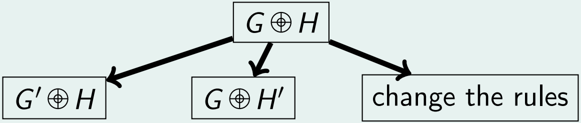
push-sum ⊕:



Almost disjunctive sum

Definition

push-sum $\oplus$:



Definition

push-equivalence:

$$G \overset{\circledcirc}{\equiv} H \Leftrightarrow \forall X \in \mathfrak{G}^{\circledcirc}, o(G \oplus X) = o(H \oplus X)$$

Push-button canonical forms

Theorem

- *For all push-game G, $G \oplus G \overset{\circledcirc}{\equiv} \textcircled{0}$.*

Push-button canonical forms

Theorem

- *For all push-game* G, $G \oplus G \overset{\circledcirc}{\equiv} \textcircled{0}$.
- *There are infinitely many equivalence classes which are winning for 2nd player.*

Push-button canonical forms

Theorem

- *For all push-game G, $G \oplus G \overset{\circledcirc}{\equiv} \textcircled{0}$.*
- *There are infinitely many equivalence classes which are winning for 2nd player.*
- *Canonical representative can be computed.*
  - *By simplifying the game tree*
  - *Adaptation of the procedure for the normal disjunctive sum.*

Push-button canonical forms

Theorem

- *For all push-game G, $G \oplus G \overset{\circledcirc}{\equiv} \textcircled{0}$.*
- *There are infinitely many equivalence classes which are winning for 2nd player.*
- *Canonical representative can be computed.*
  - *By simplifying the game tree*
  - *Adaptation of the procedure for the normal disjunctive sum.*

- Easier than misère.

Lemma

The sequence of canonical representatives for $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$ *on a single heap has infinitely many values.*

Lemma

The sequence of canonical representatives for $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$ *on a single heap has infinitely many values.*

Observation

For $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$, removing one token is never a good move.

Lemma

The sequence of canonical representatives for $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$ *on a single heap has infinitely many values.*

Observation

For $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$, removing one token is never a good move.

- The values for $\mathcal{R} \circledcirc \textsc{Sub}(\{x\})$ can be further simplified.

Lemma

The sequence of canonical representatives for $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$ *on a single heap has infinitely many values.*

Observation

For $\textsc{Sub}(\{1,2\}) \circledcirc \textsc{Sub}(\{1\})$, removing one token is never a good move.

- The values for $\mathcal{R} \circledcirc \textsc{Sub}(\{x\})$ can be further simplified.

Theorem

Let S be a finite set. The 'simplified values' of $\textsc{Sub}(S) \circledcirc \textsc{Sub}(\{x\})$ *are ultimately periodic.*

Perspectives

- General solution for multi-heap push-subtraction games?
- Find applications to other games.
- Restrictions on the rulesets.

Conclusion

Reconfiguration problems

(1-player games)

- Reconfiguration of graph colourings
- Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

- Partizan Subtraction Games
- Rules composition

Algorithmic Applications

- Sampling Colourings
- Online colouring

Conclusion

Reconfiguration problems

(1-player games)

- Reconfiguration of graph colourings
- Reconfiguration of perfect matchings

Combinatorial Games

(2-player games)

- Partizan Subtraction Games
- Rules composition

Algorithmic Applications

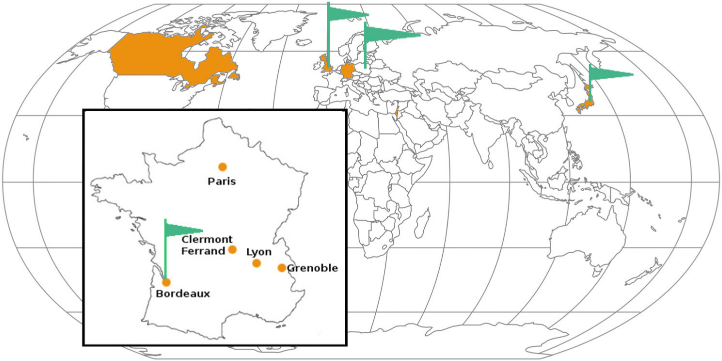
- Sampling Colourings
- Online colouring

Accepted papers

Submitted papers

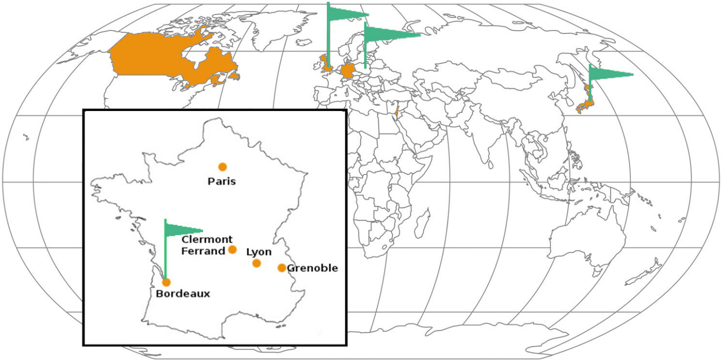
Writing in progress

Collaborations



17 co-authors

Collaborations



17 co-authors

Research stays:

- Warsaw
- Birmingham
- Bordeaux
- Tokyo

Other works

Computing maximum cliques in B_2-EPG graphs, Nicolas Bousquet, Marc Heinrich, WG, 2017.

A generalization of Arc-Kayles, Antoine Dailly, Valentin Gledel, Marc Heinrich, International Journal of Game Theory, 2018.

Enumerating minimal dominating sets in triangle-free graphs, Marthe Bonamy, Oscar Defrain, Marc Heinrich, Jean-Florent Raymond, STACS, 2019.

Thank You!