A logical approach to Isomorphism Testing and Constraint Satisfaction

Oleg Verbitsky

Humboldt University of Berlin, Germany
and
IAPMM, Lviv, Ukraine

Joint Estonian-Latvian Theory Days at Ratnieki 2014, 3–5 October

Outline

Logical complexity of graphs

Applications to Graph Isomorphism

Applications to Constraint Satisfaction (Graph Homomorphism)

Outline

Logical complexity of graphs

Applications to Graph Isomorphism

Applications to Constraint Satisfaction (Graph Homomorphism)

First-order language of graph theory

Vocabulary:

$=$ equality of vertices
$\sim$ adjacency of vertices

Syntax:

$\wedge, \vee, \neg$ etc. Boolean connectives
$\exists, \forall$ quantification over vertices
(no quantification over sets).

First-order language of graph theory

Vocabulary:

$=$ equality of vertices
$\sim$ adjacency of vertices

Syntax:

$\wedge, \vee, \neg$ etc. Boolean connectives
$\exists, \forall$ quantification over vertices
(no quantification over sets).

Example

We can say that vertices x any y lie at distance no more than n:

$$\Delta_1(x,y) \stackrel{\text{def}}{=} x \sim y \vee x = y$$

$$\Delta_n(x,y) \stackrel{\text{def}}{=} \exists z_1 \ldots \exists z_{n-1} \Big(\Delta_1(x,z_1) \wedge \\ \wedge \Delta_1(z_1,z_2) \wedge \ldots \wedge \Delta_1(z_{n-2},z_{n-1}) \wedge \Delta_1(z_{n-1},y) \Big)$$

Succinctness measures of a formula Φ

Definition

The width $W(\Phi)$ is the number of variables used in Φ (different occurrences of the same variable are not counted).

Example

$W(\Delta_n) = n + 1$ but we can economize by recycling just three variables:

$$\begin{aligned} \Delta'_1(x, y) &\stackrel{\text{def}}{=} \Delta_1(x, y) \\ \Delta'_n(x, y) &\stackrel{\text{def}}{=} \exists z(\Delta'_1(x, z) \land \Delta'_{n-1}(z, y)). \end{aligned}$$

Succinctness measures of a formula Φ

Definition

The depth $D(\Phi)$ (or quantifier rank) is the maximum number of nested quantifiers in Φ.

- $\forall x(\forall y(\exists z(\ldots)))$ – depth 3; $(\forall x \ldots) \wedge (\forall y \ldots) \wedge (\exists z \ldots)$ – depth 1

Example

$D(\Delta'_n) = n - 1$ but we can economize using the halving strategy:

$$\begin{aligned} \Delta''_1(x,y) &\stackrel{\text{def}}{=} \Delta_1(x,y) \\ \Delta''_n(x,y) &\stackrel{\text{def}}{=} \exists z \left(\Delta''_{\lfloor n/2 \rfloor}(x,z) \wedge \Delta''_{\lceil n/2 \rceil}(z,y) \right). \end{aligned}$$

Now $D(\Delta''_n) = \lceil \log n \rceil$ and $W(\Delta''_n) = 3$.

Definition

A statement Φ defines a graph G if Φ is true on G but false on every non-isomorphic graph H.

Example

P_n, the path on n vertices, is defined by

$$
\begin{aligned}
&\forall x \forall y \Delta_{n-1}(x,y) \wedge \neg \forall x \forall y \Delta_{n-2}(x,y) \\
&\qquad\qquad\qquad\qquad \texttt{\% diameter = n-1} \\
&\wedge \forall x \forall y_1 \forall y_2 \forall y_3 (x \sim y_1 \wedge x \sim y_2 \wedge x \sim y_3 \\
&\qquad\qquad \rightarrow y_1 = y_2 \vee y_2 = y_3 \vee y_3 = y_1) \\
&\qquad\qquad\qquad\qquad \texttt{\% max degree < 3} \\
&\wedge \exists x \exists y \forall z \big(x \sim y \wedge (z \sim x \rightarrow z = y)\big) \\
&\qquad\qquad\qquad\qquad \texttt{\% min degree = 1}
\end{aligned}
$$

The logical depth and width of a graph

Definition

$D(G)$ is the minimum $D(\Phi)$ over all Φ defining G.
$W(G)$ is the minimum $W(\Phi)$ over all Φ defining G.

Example

- $W(P_n) \leq 4$
- $D(P_n) < \log n + 3$

Remark

$W(G) \leq D(G) \leq n + 1$, where $n = v(G)$

The logical depth and width of a graph

Definition

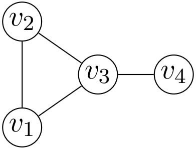
$D(G)$ is the minimum $D(\Phi)$ over all Φ defining G.
$W(G)$ is the minimum $W(\Phi)$ over all Φ defining G.

Example

- $W(P_n) \leq 4$
- $D(P_n) < \log n + 3$

Remark

$W(G) \leq D(G) \leq n + 1$, where $n = v(G)$



$$\exists x_1 \exists x_2 \exists x_3 \exists x_4 \forall y \Big(\bigwedge_{1 \leq i < j \leq 4} x_i \neq x_j \wedge \bigvee_{1 \leq i \leq 4} y = x_i \wedge x_1 \sim x_2 \wedge x_1 \sim x_3 \wedge x_2 \sim x_3 \wedge x_3 \sim x_4 \wedge \wedge x_1 \not\sim x_4 \wedge x_2 \not\sim x_4 \Big)$$

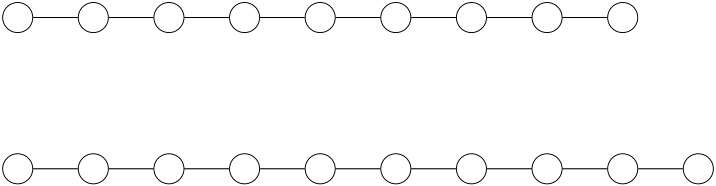
How to determine $W(G)$ or $D(G)$?

- $D(G) = \max_{H \not\cong G} D(G, H)$, where $D(G, H)$ is the minimum quantifier depth needed to distinguish between G and H. Similarly for $W(G)$.
- $D(G, H)$ and $W(G, H)$ are characterized in terms of a combinatorial game:

  G and H are distinguishable with k variables and quantifier depth r iff Spoiler wins the k-pebble Ehrenfeucht game on G and H in r rounds.

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



$G = P_9$

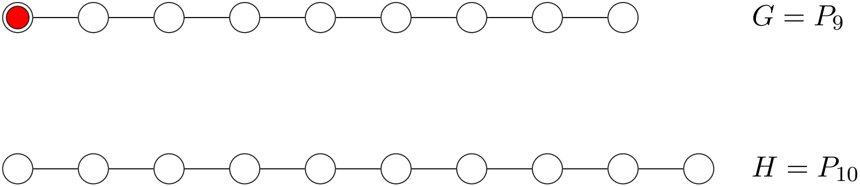
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$

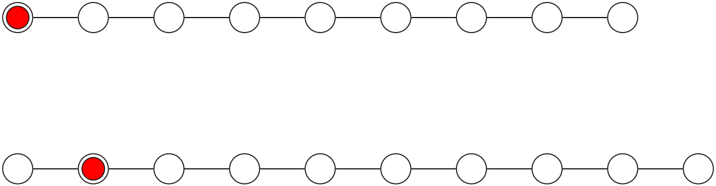


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



$G = P_9$

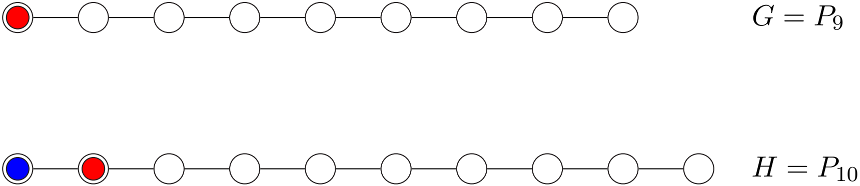
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

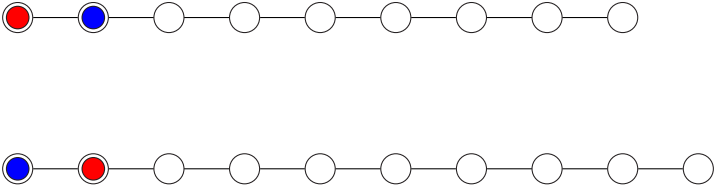


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



$G = P_9$

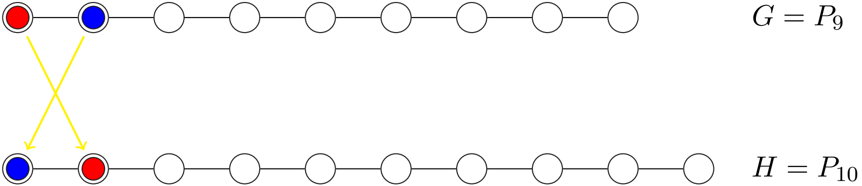
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

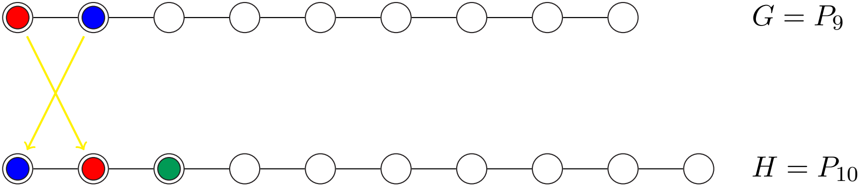


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$

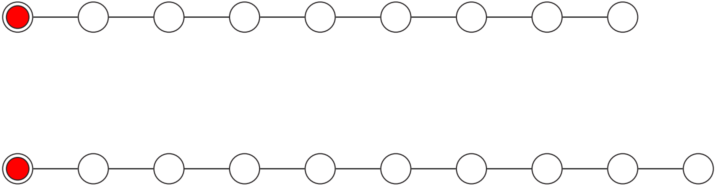


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



$G = P_9$

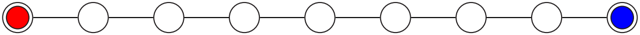
$H = P_{10}$

Two players: Spoiler and Duplicator

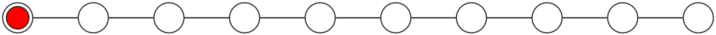
Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$



$G = P_9$



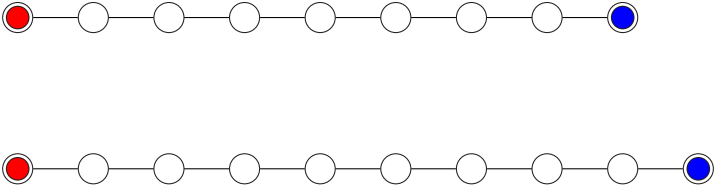
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



$G = P_9$

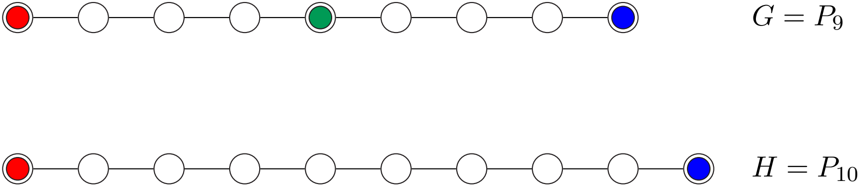
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

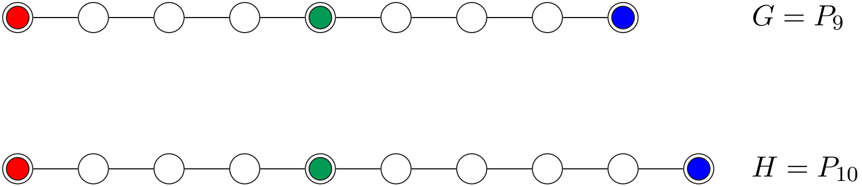


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

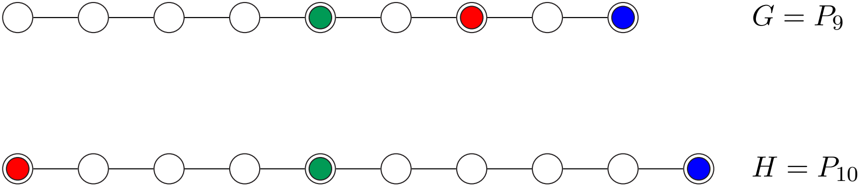


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$

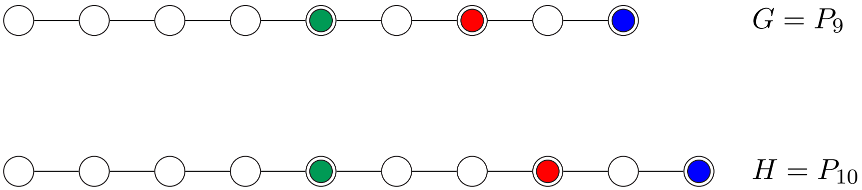


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

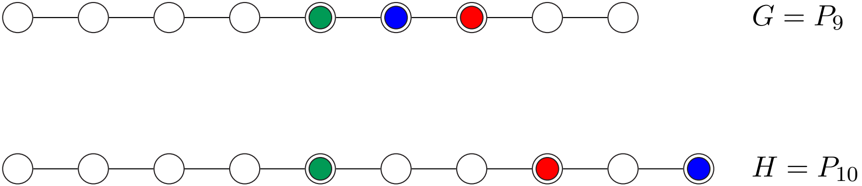


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



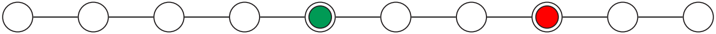
Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 1: $W(P_n, P_{n+1}) \le 3$, $D(P_n, P_{n+1}) \le \log_2 n + 3$

$G = P_9$



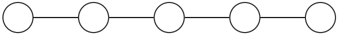
$H = P_{10}$

Two players: Spoiler and Duplicator

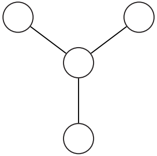
Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



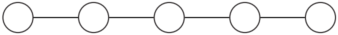
$G = P_5$



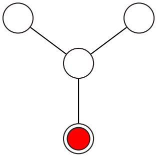
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



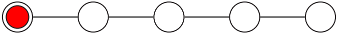
$G = P_5$



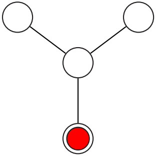
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



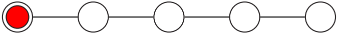
$G = P_5$



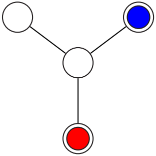
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



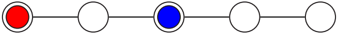
$G = P_5$



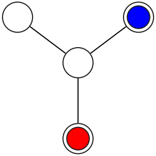
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



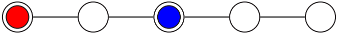
$G = P_5$



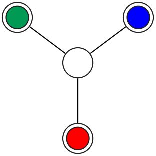
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



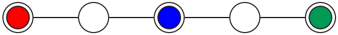
$G = P_5$



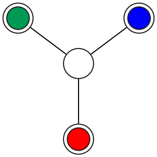
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



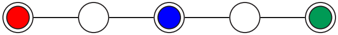
$G = P_5$



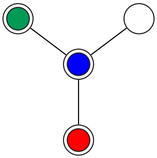
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



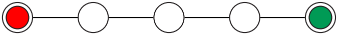
$G = P_5$



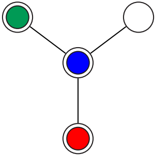
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

Example 2: $W(P_n) \leq 3$



$G = P_5$



$K_{1,3}$ in H

Left aside from this talk

Consider n-vertex graphs.

- If $G \ncong H$, then $D(G, H) \leq n$. Can this be improved?
- What is $D(G)$ if G is chosen at random?
- What is the minimum possible value of $D(G)$?
- How do the answers change if we restrict the number of quantifier alternations?

(joint work with Joel Spencer and Oleg Pikhurko)

k-variable logic

$D^k(G)$ denotes the logical depth of G in the k-variable logic (assuming $W(G) \leq k$).

For example, $D^3(P_n) \leq \log n + 3$.

k-variable logic

$D^k(G)$ denotes the logical depth of G in the k-variable logic (assuming $W(G) \leq k$).

For example, $D^3(P_n) \leq \log n + 3$.

A disturbing fact: We may need many variables even for very simple graphs.

For example, $W(K_{1,n}) \geq n$ because $W(K_{1,n}, K_{1,n+1}) \geq n$.

Logic with counting quantifiers

$\exists^m x \Psi(x)$ means that there are at least m vertices x having property Ψ.

The counting quantifier $\exists^m$ contributes 1 in the quantifier depth whatever m.

Example

$K_{1,n}$ can now be defined by

$$\exists^{n+1}(x = x) \wedge \neg\exists^{n+2}(x = x) \wedge \\ \exists x \forall y \forall z (y \neq x \wedge z \neq x \to y \sim x \wedge y \not\sim z)$$

Therefore, $W_{\#}(K_{1,n}) \leq 3$ and $D^3_{\#}(K_{1,n}) \leq 3$.

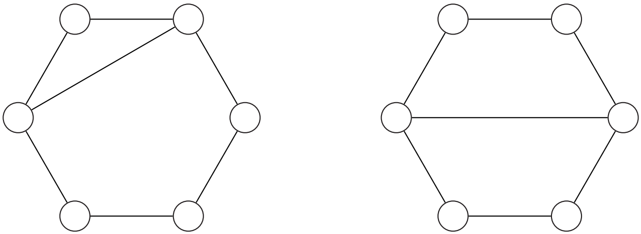
Outline

Logical complexity of graphs

Applications to Graph Isomorphism

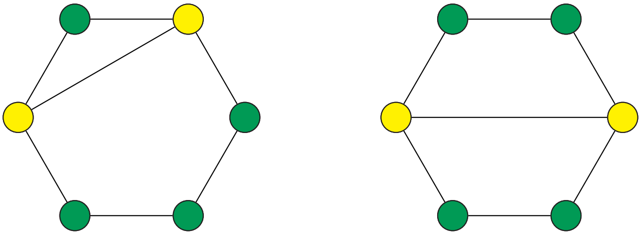
Applications to Constraint Satisfaction (Graph Homomorphism)

Color refinement algorithm

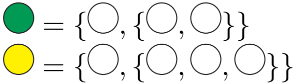


Initial coloring is monochromatic.

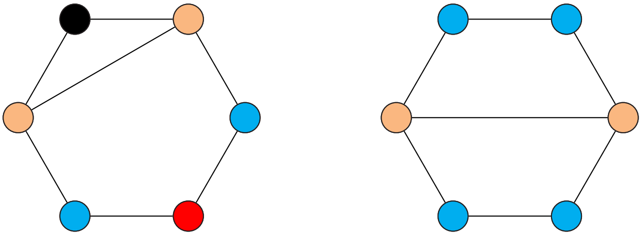
Color refinement algorithm



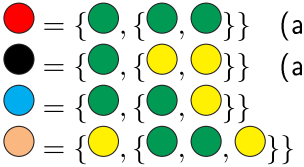
New color of a vertex = old color + old colors of all neighbours.



Color refinement algorithm



Next refinement.



(absent in the second graph)

(absent in the second graph)

Color refinement algorithm

Theorem (Immerman, Lander 90)

Color Refinement works correctly on G and every H iff $W_{\#}(G) \leq 2$.

This is the case for

- all trees [Edmonds 65]
- almost all graphs [Babai, Erdős, Selkow 82]

k-dimensional Weisfeiler-Lehman algorithm

- 1-dim WL = the color refinement algorithm
- k-dim WL colors $V(G)^k$
- Initial coloring: $C^1(\bar{u})$ = the equality type of $\bar{u} \in V(G)^k$ and the isomorphism type of the spanned subgraph
- Color refinement:
  $C^i(\bar{u}) = \{C^{i-1}(\bar{u}), \{(C^{i-1}(\bar{u}^{1,x}), \ldots, C^{i-1}(\bar{u}^{k,x}))\}_{x \in V}\}$,
  where $(u_1, \ldots, u_i, \ldots, u_k)^{i,x} = (u_1, \ldots, x, \ldots, u_k)$

The Weisfeiler-Lehman algorithm

- purports to decide if input graphs G and H are isomorphic:
  - If $G \cong H$, the output is correct,
  - if $G \ncong H$, the output can be wrong;
- has two parameters: dimension and number of rounds.
- Fixed dimension $k \implies \leq n^k$ rounds $\implies$ polynomial running time.
- Fixed dimension and $O(\log n)$ rounds $\implies$ parallel logarithmic time.

Theorem (Cai, Fürer, Immerman 92)

The r-round k-dim WL works correctly on G and every H if

$$k = W_{\#}(G) - 1 \text{ and } r = D_{\#}^{k+1}(G) - 1.$$

On the other hand, it is wrong on (G, H) for some H if

$$k < W_{\#}(G) - 1, \text{ whatever } r.$$

The Weisfeiler-Lehman algorithm

Corollary (Cai, Fürer, Immerman 92)

Let $\mathcal{C}$ *be a class of graphs* G *with* $W_{\#}(G) \leq k$ *for a constant* k*.*
Then Graph Isomorphism for $\mathcal{C}$ *is solvable in* P*.*

Corollary (Grohe, V. 06)

1. *Let* $\mathcal{C}$ *be a class of graphs* G *with* $D^k_{\#}(G) = O(\log n)$*.*
   Then Graph Isomorphism for $\mathcal{C}$ *is solvable in* $\mathrm{TC}^1 \subseteq \mathrm{NC}^2 \subseteq \mathrm{AC}^2$*.*
2. *Let* $\mathcal{C}$ *be a class of graphs* G *with* $D^k(G) = O(\log n)$*.*
   Then Graph Isomorphism for $\mathcal{C}$ *is solvable in* $\mathrm{AC}^1 \subseteq \mathrm{TC}^1$*.*

Classes of graphs: Trees

- $W_{\#}(T) \leq 2$ for every tree T.
- $D^2_{\#}(P_n) \geq \frac{n}{2} - 1$
- one extra variable $\implies$ logarithmic depth !

Theorem

If T is a tree on n vertices, then $D^3_{\#}(T) \leq 3 \log n + 2$.

Isomorphism of trees (history revision)

Theorem
If T is a tree on n vertices, then $D^3_{\#}(T) \leq 3\log n + 2$.

Testing isomorphism of trees is

- in Log-Space [Lindell 92]
- in AC^1 [Miller-Reif 91]
- in AC^1 if $\Delta = O(\log n)$ [Ruzzo 81]
- in Lin-Time by 1-WL ($W_{\#}(T) = 2$) [Edmonds 65]

Miller and Reif [SIAM J. Comput. 91]: *"No polylogarithmic parallel algorithm was previously known for isomorphism of unbounded-degree trees."*

However, the $3\log n$-round 2-WL solves it in TC^1 and is known since 68 !

Classes of graphs: Bounded tree-width, planar, interval

Theorem

For a graph G of tree-width k on n vertices

$W_{\#}(G) \leq k+2$ *[Grohe, Mariño 99];*

$D_{\#}^{4k+4}(G) < 2(k+1)\log n + 8k + 9$ *[Grohe, V. 06].*

Theorem

For a planar graph G on n vertices

$W_{\#}(G) = O(1)$ *[Grohe 98].*

If G is, moreover, 3-connected, then

$D^{15}(G) < 11 \log n + 45$ *[V. 07].*

Theorem

For an interval graph G on n vertices

$W_{\#}(G) \leq 3$ *[Evdokimov et al. 00, Laubner 10];*

$D_{\#}^{15}(G) < 9\log n + 8$ *[Köbler, Kuhnert, Laubner, V. 11].*

Graphs with an excluded minor

Theorem (Grohe 11)

For each F, if G excludes F as a minor, then

$$W_{\#}(G) = O(1).$$

Open problem

Is it then true that $D^k_{\#}(G) = O(\log n)$ for some constant k?

Outline

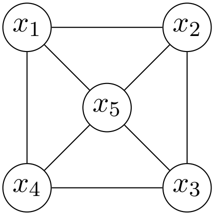
Logical complexity of graphs

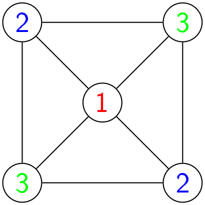
Applications to Graph Isomorphism

Applications to Constraint Satisfaction (Graph Homomorphism)

Constraint Satisfaction Problem (CSP)

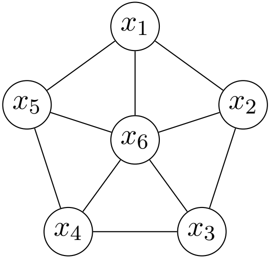
Variables	x_1, x_2, x_3, x_4, x_5
Values	$x_i \in \{1, 2, 3\}$
Constraints	$x_1 \neq x_2$, $x_2 \neq x_3$, $x_3 \neq x_4$, $x_4 \neq x_1$, $x_1 \neq x_5$, $x_2 \neq x_5$, $x_3 \neq x_5$, $x_4 \neq x_5$
Question:	Is there an assignment of values to the variables satisfying all constraints?





Constraint Satisfaction Problem (CSP)

Variables	$x_1, x_2, x_3, x_4, x_5, x_6$
Values	$x_i \in \{1, 2, 3\}$
Constraints	$x_1 \neq x_2$, $x_2 \neq x_3$, $x_3 \neq x_4$, $x_4 \neq x_5$, $x_5 \neq x_1$,
	$x_1 \neq x_6$, $x_2 \neq x_6$, $x_3 \neq x_6$, $x_4 \neq x_6$, $x_5 \neq x_6$
Question:	Is there an assignment of values to the variables satisfying all constraints?



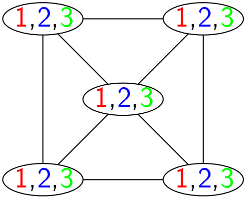
No!

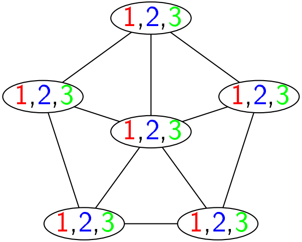
Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

Derivation rules: $\frac{x = 1, \ y \neq x}{y \neq 1}$ etc.



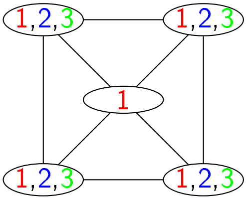


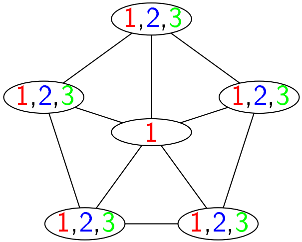
Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

Derivation rules: $\dfrac{x = 1,\ \ y \neq x}{y \neq 1}$ etc.



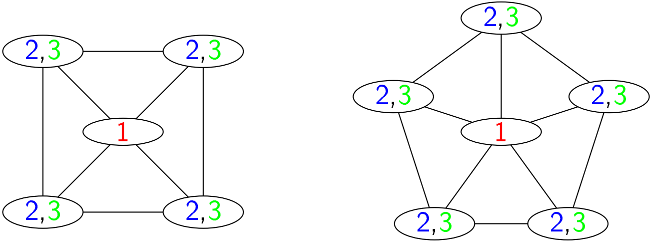


Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

Derivation rules: $\frac{x = 1,\ y \neq x}{y \neq 1}$ etc.

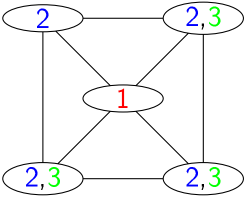


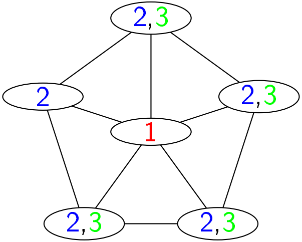
Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

Derivation rules: $\dfrac{x = 1,\ y \neq x}{y \neq 1}$ etc.



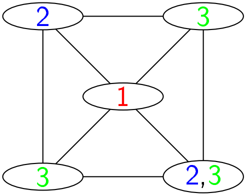


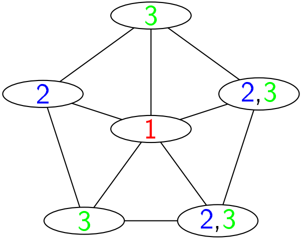
Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

Derivation rules: $\dfrac{x = 1,\ y \neq x}{y \neq 1}$ etc.



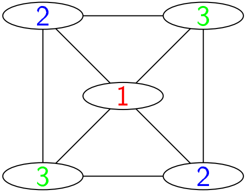


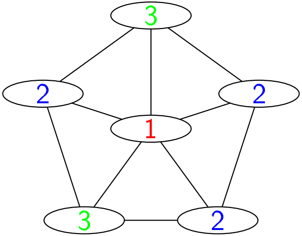
Constraint propagation

Methodolody: derivation instead of search

Example: 3-COLORABILITY.
We can choose an arbitrary edge and color it arbitrarily.

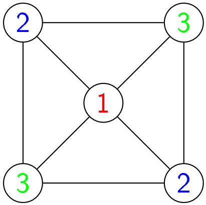
Derivation rules: $\dfrac{x = 1,\ y \neq x}{y \neq 1}$ etc.

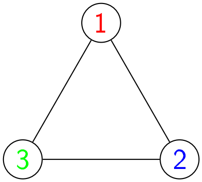




The Feder-Vardi paradigm: a CSP = a Homomorphism Problem

For example, a graph G is 3-colorable iff there is a homomorphism from G to K_3 (notation: $G \to K_3$).





A logic and a game for the Homomorphism Problem

The following three conditions are equivalent:

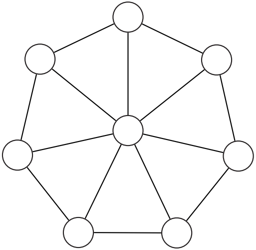
- $G \not\to H$,
- some existential-positive formula distinguishes G from H,
- Spoiler has a winning strategy in the existential k-pebble game on G and H for some k.

The existential k-pebble game on G and H is the version of the k-pebble Ehrenfeucht game where

- Spoiler moves always in G,
- Duplicator must keeps a partial homomorphism.

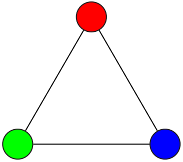
An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



Spoiler

1 2 3 4

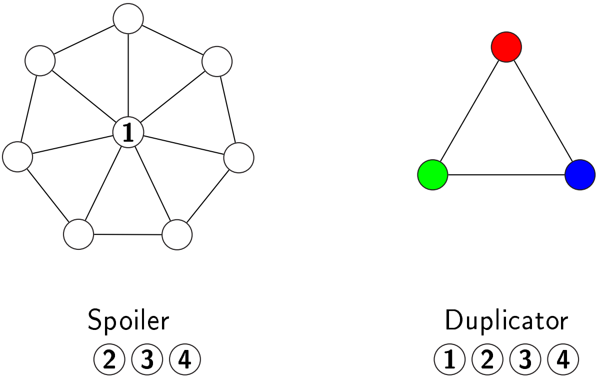


Duplicator

1 2 3 4

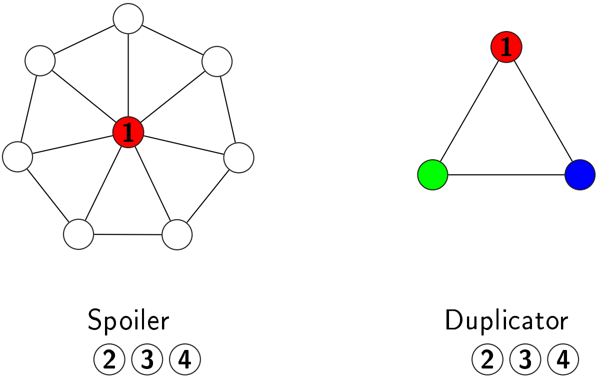
An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



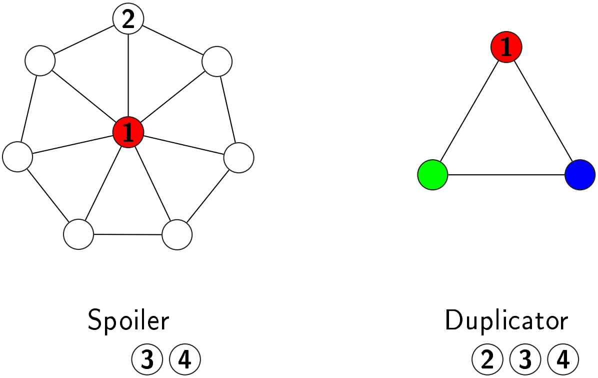
An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



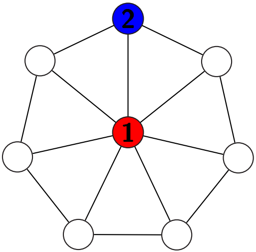
An example

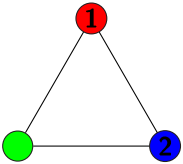
Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.





Spoiler

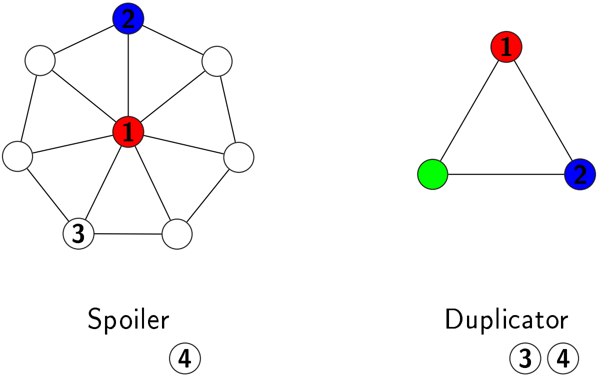
3 4

Duplicator

3 4

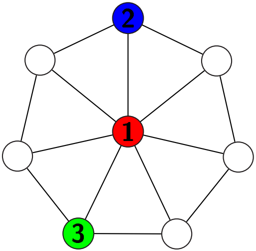
An example

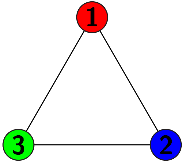
Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



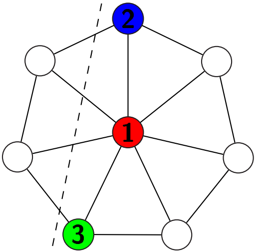


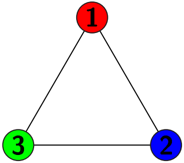
Spoiler
4

Duplicator
4

An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.





Spoiler

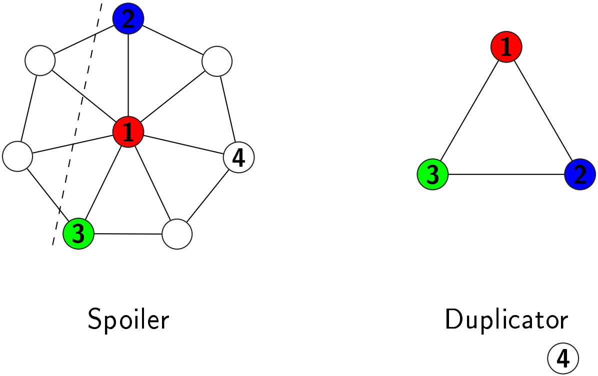
4

Duplicator

4

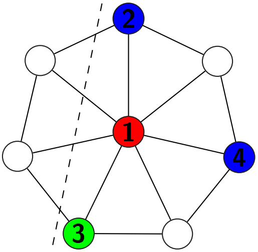
An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.

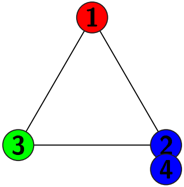


An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



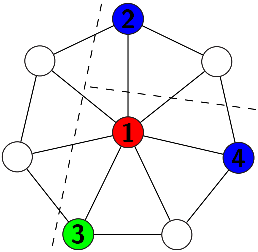
Spoiler



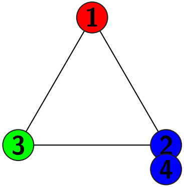
Duplicator

An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



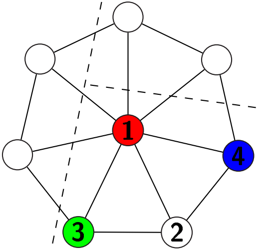
Spoiler



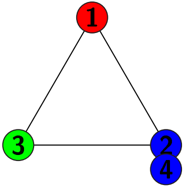
Duplicator

An example

Let G_n denote the wheel graph with n vertices.
If n is even, then Spoiler wins the existential game on G_n and K_3 with 4 pebbles.



Spoiler



Duplicator

The triad: A logic, a game, an algorithm

Theorem (Kolaitis, Vardi 95)

Suppose that $G \not\to H$. The the following three conditions are equivalent:

- $W_{\exists,+}(G,H) \leq k$, i.e., G is distinguishable from H by an existential-positive sentence with k variables;
- Spoiler wins the existential k-pebble game on G and H;
- k-Consistency Checking recognizes that $G \not\to H$.

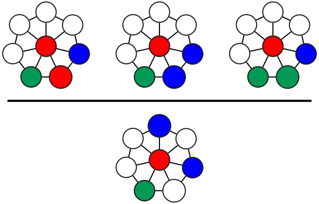
k-Consistency Checking (recasted)

Algorithmic problem

Given two finite structures G and H, does Spoiler win the existential k-pebble game on these structures?

- This is a relaxation of the homomorphism problem.
- For small k, it is commonly used as a heuristics approach.

A propagation-based algorithm makes derivations like



- winning positions for Spoiler

$\Downarrow$

- a winning position too

(a position is a mapping of $\leq k$ vertices from $V(G)$ into $V(H)$)

Spoiler has a winning stategy $\Leftrightarrow$ the uncolored graph is derivable. Since there are at most $N = v(G)^k v(H)^k$ positions, all derivations can be generated in time N^{k+1} (the wasteful version of k-consistency checking). If k is fixed, this takes polynomial time.

The time complexity of k-Consistency Checking

Theorem

The k-Consistency problem is solvable in

- *time* $O(v(G)^k v(H)^k) = O(n^{2k})$ *for each* k [Cooper 89]
- *but not in time* $O(n^{\frac{k-3}{12}})$ *for* $k \geq 15$ [Berkholz 12]

The time complexity of k-Consistency Checking

Theorem

The k-Consistency problem is solvable in

- *time $O(v(G)^k v(H)^k) = O(n^{2k})$ for each k [Cooper 89]*
- *but not in time $O(n^{\frac{k-3}{12}})$ for $k \geq 15$ [Berkholz 12]*

Question. What about arc consistency $(k = 2)$?

Remark. If $k = 2$, we consider directed graphs.

The time complexity of k-Consistency Checking

Theorem

The k-Consistency problem is solvable in

- *time $O(v(G)^k v(H)^k) = O(n^{2k})$ for each k [Cooper 89]*
- *but not in time $O(n^{\frac{k-3}{12}})$ for $k \geq 15$ [Berkholz 12]*

Question. What about arc consistency $(k = 2)$?

Remark. If $k = 2$, we consider directed graphs.

In practice: All arc consistency $(k = 2)$ algorithms

- such as AC-1, AC-3, AC-3.1 / AC-2001, AC-3.2, AC-3.3, AC-3_d, AC-4, AC-5, AC-6, AC-7, AC-8, AC-$*$
- and several parallel/distributed variants

are based on constraint propagation.

Bounds for the propagation approach

Upper bounds for Arc Consistency

- Sequential: $O(v(G)e(H) + e(G)v(H))$, which implies $O(n^3)$.
- Parallel: $O(\mathrm{depth}(G, H)) \leq O(v(G)v(H))$, implies $O(n^2)$.

Theorem (Berkholz, V. 13)

Any sequential propagation-based arc consistency algorithm takes time $\Omega(n^3)$, and any such parallel algorithm takes time $\Omega(n^2)$.

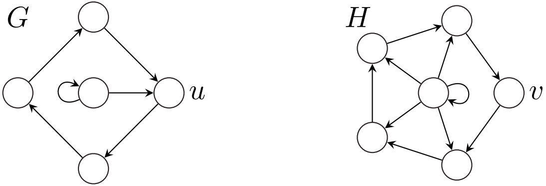
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



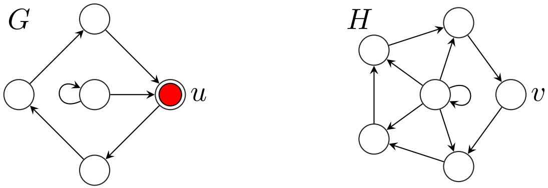
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



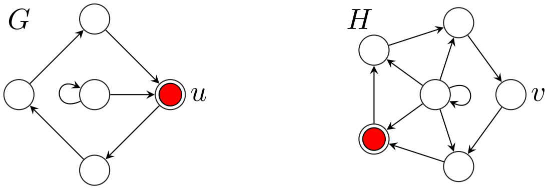
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



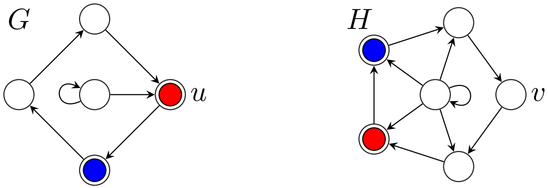
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



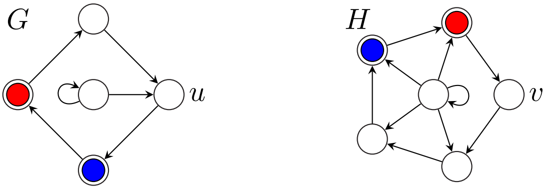
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



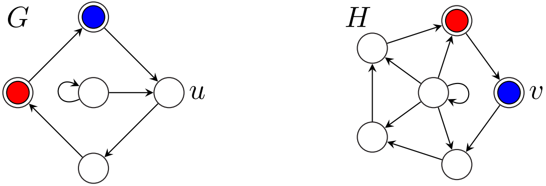
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



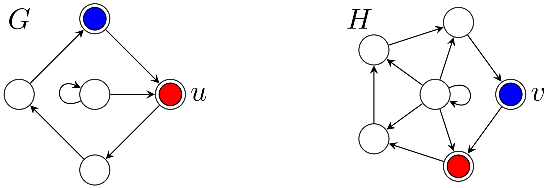
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



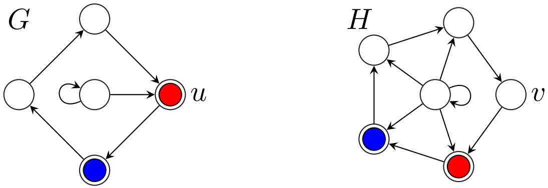
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



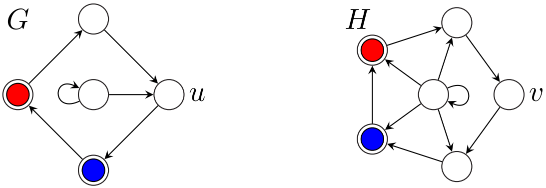
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



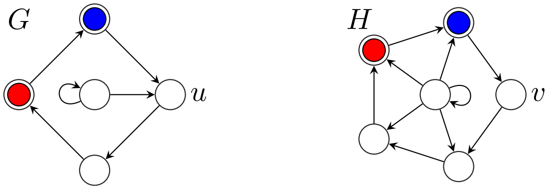
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



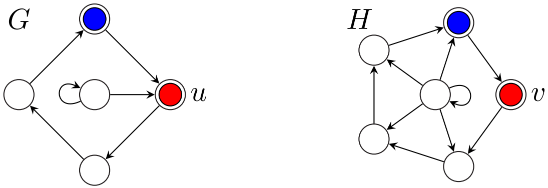
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



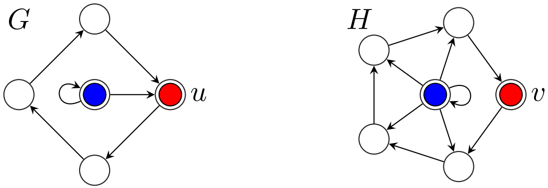
The core of the proof

Lemma

There are directed graphs G and H with $v(G) = v(H) - 1 = n$ such that

- Spoiler wins the existential 2-pebble game on G and H;
- Duplicator can resist $\Omega(n^2)$ rounds.

Remark. $n^2 + 1$ rounds always suffice for Spoiler.



The 3-COLORABILITY problem

Given: a graph G
Decide: $\chi(G) \leq 3$?

Solvable: in time $O(1.3289^n)$ (Beigel-Eppstein 05)

not in time $2^{o(n)}$, under the Exponential Time Hypothesis

A reminder: The non-3-colorability of wheel graphs with even number of vertices can be established by k-consistency checking with $k = 4$.

Question. What is the minimum $k = k(n)$ such that k-consistency checking is successful for all graphs with n-vertices?

Dynamic width of the 3-colorability problem

Definition

$W(n) = \{ W_{\exists,+}(G, K_3) : v(G) = n, \chi(G) > 3\}$

Remark

- If $W(n) \leq k(n)$, then 3COL is solvable in time $n^{O(k(n))}$.
- NP≠P $\Rightarrow W(n)$ is unbounded.

Theorem (Nešetřil, Zhu 96)

$W(n) = \Omega\left(\frac{\log\log n}{\log\log\log n}\right)$.

Dynamic width of the 3-colorability problem

Definition

$W(n) = \{ W_{\exists,+}(G, K_3) : v(G) = n,\ \chi(G) > 3 \}$

Remark

- If $W(n) \leq k(n)$, then 3COL is solvable in time $n^{O(k(n))}$.
- NP$\neq$P $\Rightarrow W(n)$ is unbounded.

Theorem (Nešetřil, Zhu 96)

$W(n) = \Omega\left(\frac{\log\log n}{\log\log\log n}\right)$.

Remark

- Exponential Time Hypothesis $\implies W(n) = \Omega(n/\log n)$.

Theorem (Atserias, Dawar, V. 14)

$W(n) = \Omega(n)$.

Dynamic width of 3COL over planar graphs

3COL of planar graphs is NP-complete, solvable in time $2^{O(\sqrt{n})}$ but, under Exponential Time Hypothesis, not in time $2^{o(\sqrt{n})}$ (Marx 13).

Definition

$W_{\mathrm{planar}}(n) = \max \{ W_{\exists,+}(G, K_3) : G$ planar, $v(G) = n$, $\chi(G) > 3 \}$.

Remark

- $W_{\mathrm{planar}}(n) \leq 5\sqrt{n}$ because $tw(G) \leq 5\sqrt{n} - 1$ for planar G, which allows Spoiler to use a divide-and-conquer strategy like for the wheel graphs.
- Exponential Time Hypothesis $\Rightarrow W_{\mathrm{planar}}(n) = \Omega(\sqrt{n}/\log n)$.

Theorem (Atserias, Dawar, V. 14)

$W_{\mathrm{planar}}(n) = \Omega(\sqrt{n})$.

Conclusion

Our lower bounds show that Consistency Checking is not an optimal approach to get an exact exponential algorithm for 3-COLORABILITY, also when only planar inputs are considered.

Conclusion

Our lower bounds show that Consistency Checking is not an optimal approach to get an exact exponential algorithm for 3-COLORABILITY, also when only planar inputs are considered.

Open problem

Can Consistency Checking be competitive for

- 3-COLORABILITY for other classes of graphs,
- other constraint satisfaction problem with unbounded dynamic width?

Conclusion

Our lower bounds show that Consistency Checking is not an optimal approach to get an exact exponential algorithm for 3-COLORABILITY, also when only planar inputs are considered.

Open problem

Can Consistency Checking be competitive for

- 3-COLORABILITY for other classes of graphs,
- other constraint satisfaction problem with unbounded dynamic width?

Thank you for your attention!