A logical approach to Isomorphism Testing and Constraint Satisfaction

Oleg Verbitsky

Humboldt University of Berlin, Germany

ESSLLI 2016, 15–19 August

Course outline

1. Logical complexity of graphs:
   Basic definitions and examples
2. Isomorphism Testing by Color Refinement and $\mathrm{FO}^2_\#$
   (first-order logic with 2 variables and counting quantifiers)
3. $\mathrm{FO}^2_\#$ and linear programming methods
4. $\mathrm{FO}^2_\#$ and Distributed Computing
5. Existential-positive FO^2 and Constraint Satisfaction
6. Alternation hierarchy of FO^k
7. $\mathrm{FO}^k_\#$ and the Weisfeiler-Leman algorithm

Part 1: Logical complexity of graphs: Basic definitions and examples

Outline

Outline

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 and y lie at distance no more than n:

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

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

Outline

Succinctness measures of a formula Φ: Width

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:

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

Succinctness measures of a formula Φ: Depth

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 $W(\Delta''_n) = 3$.

Succinctness measures of a formula Φ: Length

Definition

The length $L(\Phi)$ is the total number of symbols in Φ (each variable symbol contributes 1).

Example: $L(\Delta_n) = O(n)$ and $L(\Delta''_n) = O(n)$ but we can economize

$$\Delta'''_{2n+1}(x,y) \stackrel{\text{def}}{=} \exists z \left(\Delta_1(x,z) \wedge \Delta'''_{2n}(z,y)\right)$$

$$\Delta'''_{2n}(x,y) \stackrel{\text{def}}{=} \exists z \forall u \big(u = x \vee u = y \rightarrow \Delta'''_n(u,z)\big),$$

getting $L(\Delta'''_n) = O(\log n)$ and still keeping $D(\Delta'''_n) \leq 2\log n$ and $W(\Delta'''_n) = 4$.

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

$$\forall x \forall y \Delta_{n-1}(x,y) \land \neg \forall x \forall y \Delta_{n-2}(x,y)$$

`% diameter = n-1`

$$\land \forall x \forall y_1 \forall y_2 \forall y_3 (x \sim y_1 \land x \sim y_2 \land x \sim y_3$$
$$\rightarrow y_1 = y_2 \lor y_2 = y_3 \lor y_3 = y_1)$$

`% max degree < 3`

$$\land \exists x \exists y \forall z \big(x \sim y \land (z \sim x \rightarrow z = y)\big)$$

`% min degree = 1`

Logical depth, width, and length of a graph: Definitions

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

Example

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

Logical depth, width, and length of a graph: Relations

$W(G) \leq D(G) < L(G)$

Exercise

Prove that for any sentence Φ there is an equivalent Φ' such that $W(\Phi') \leq D(\Phi)$.

Logical depth, width, and length of a graph: Relations

$W(G) \leq D(G) < L(G)$

Exercise

Prove that for any sentence Φ there is an equivalent Φ' such that $W(\Phi') \leq D(\Phi)$.

Theorem (Pikhurko, Spencer, V. 2006)

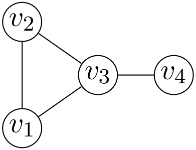
$L(G) < Tower(D(G) + \log^* D(G) + 2)$. *This bound is tight in the sense that* $L(G) \geq Tower(D(G) - 7)$ *for infinitely many* G*.*

† $Tower(1) = 2, \quad Tower(i+1) = 2^{Tower(i)}$
‡ $\log^* n = \min\{ i : Tower(i) \geq n \}$, the inverse of $Tower(i)$

Logical depth, width, and length of a graph: Upper bounds

- Every finite graph G is definable.
- If G has n vertices, then
  - $D(G) \leq n+1$,
  - $L(G) = O(n^2)$.

Proof by example:



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

Outline

Toolkit

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

- $D(G) = \max_{H \ncong G} D(G, H)$, where $D(G, H)$ is the minimum quantifier depth needed to distinguish between G and H. Similarly for $W(G)$.

Toolkit

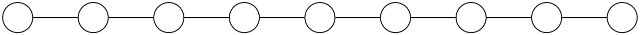
How to determine $W(G)$ or $D(G)$?

- $D(G) = \max_{H \ncong G} D(G,H)$, 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:

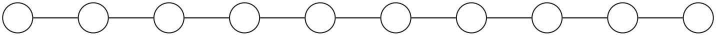
> *G and H are distinguishable with k variables and quantifier depth r iff Spoiler wins the k-pebble Ehrenfeucht game on G and H in r rounds.*

The k-pebble Ehrenfeucht game

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



$G = P_9$



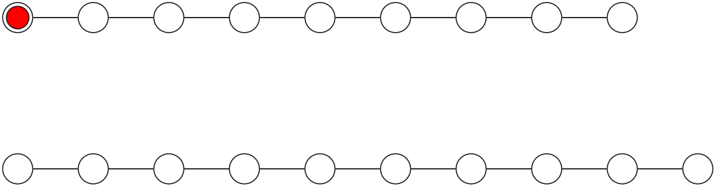
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



$G = P_9$

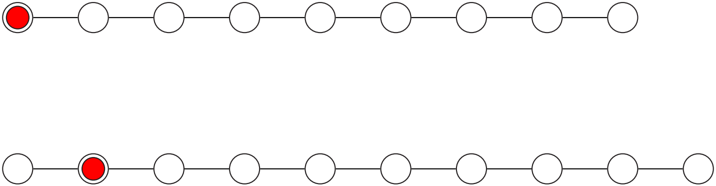
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



$G = P_9$

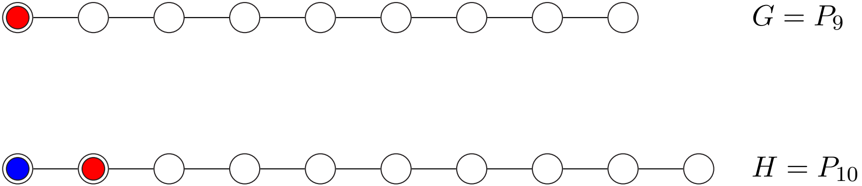
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

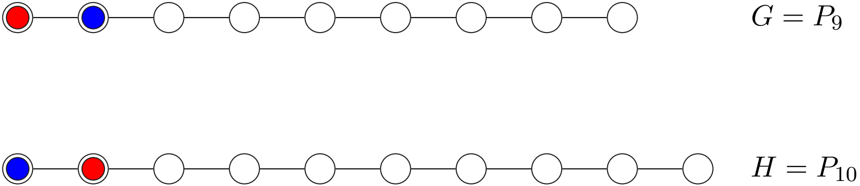
Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

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

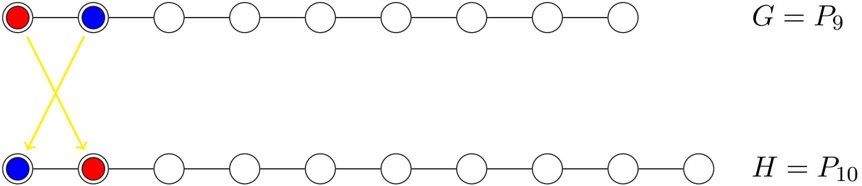


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

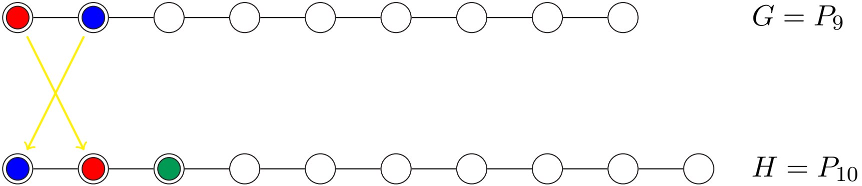
Example 1: $W(P_n, P_{n+1}) \leq 3$, $D(P_n, P_{n+1}) \leq \log_2 n + 3$



Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

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

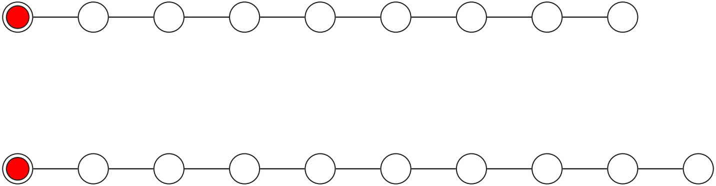


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



$G = P_9$

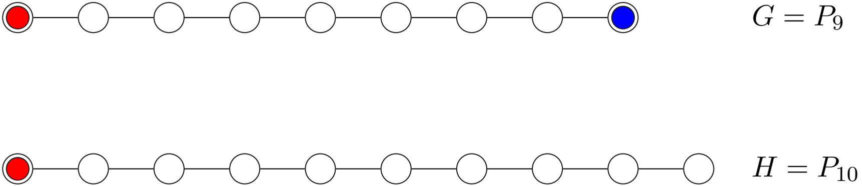
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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

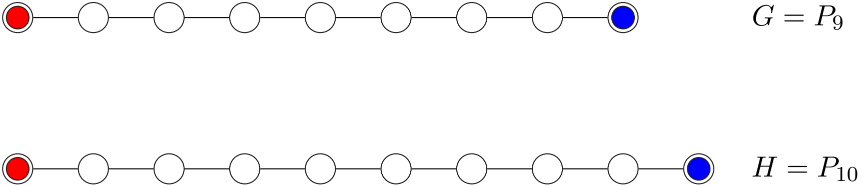


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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

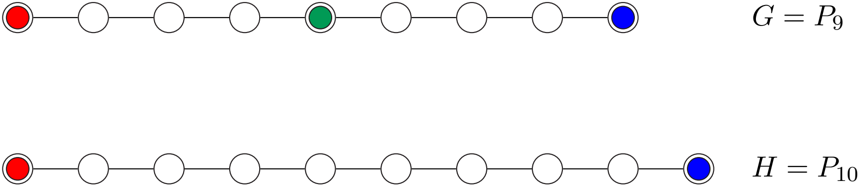


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



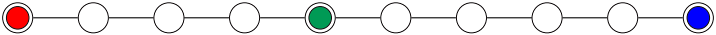
Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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

$G = P_9$



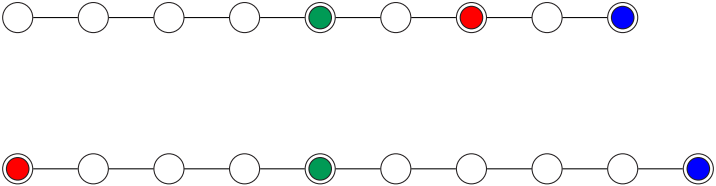
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



$G = P_9$

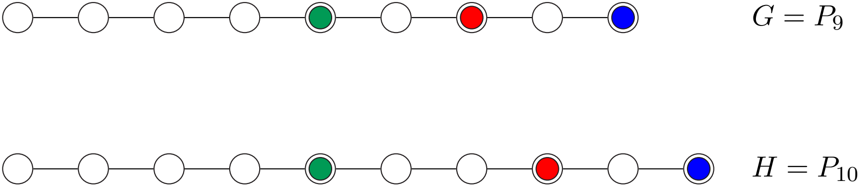
$H = P_{10}$

Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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

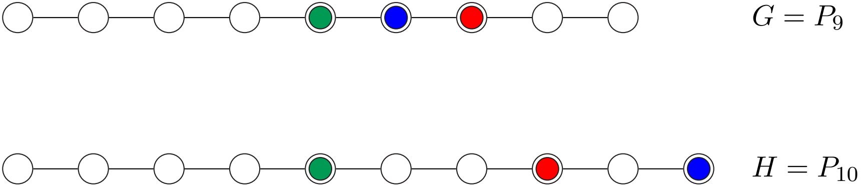


Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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



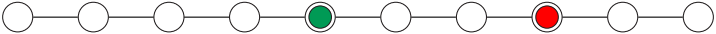
Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

The k-pebble Ehrenfeucht game

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

$G = P_9$

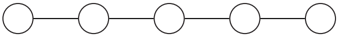


$H = P_{10}$

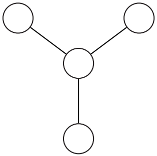
Two players: Spoiler and Duplicator

Duplicator's objective: to keep a partial isomorphism

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

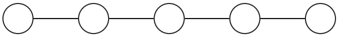


$G = P_5$

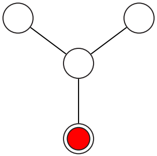


$K_{1,3}$ in H

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



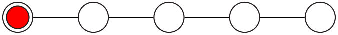
$G = P_5$



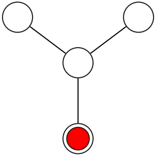
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

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

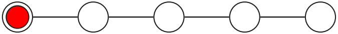


$G = P_5$

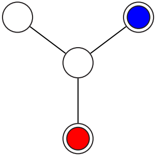


$K_{1,3}$ in H

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



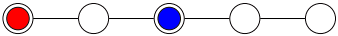
$G = P_5$



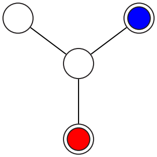
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

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

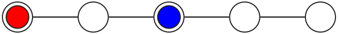


$G = P_5$

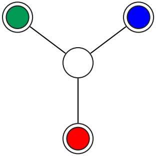


$K_{1,3}$ in H

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

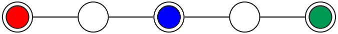


$G = P_5$

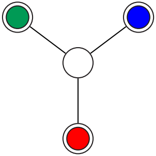


$K_{1,3}$ in H

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



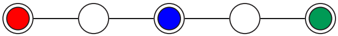
$G = P_5$



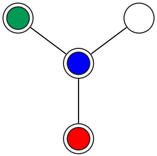
$K_{1,3}$ in H

The k-pebble Ehrenfeucht game

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

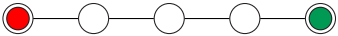


$G = P_5$

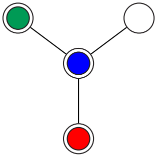


$K_{1,3}$ in H

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



$G = P_5$



$K_{1,3}$ in H

Exercises

Exercise 1

Prove that $W(P_n) = 3$ if $n \geq 2$.

Exercises

Exercise 1

Prove that $W(P_n) = 3$ if $n \geq 2$.

Exercise 2

Let $\overline{G}$ denote the complement graph of G.
Prove that $W(\overline{G}) = W(G)$ and $D(\overline{G}) = D(G)$.

Exercise 3

Let $G + H$ denote the vertex-disjoint union of G and H.
Suppose that both G and H are connected. Prove that

$$W(G) \leq W(G + H) \leq W(G) + W(H).$$

Outline

1. First-order logic (FO)
2. The logical width/depth/length of a graph
3. Ehrenfeucht game
4. Finite-variable logics and counting quantifiers
5. References

k-variable logic (FO^k)

$D^k(G)$ denotes the logical depth of G in FO^k
(assuming $W(G) \le k$).

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

k-variable logic (FO^k)

$D^k(G)$ denotes the logical depth of G in FO^k (assuming $W(G) \le k$).

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

Theorem

1. $D^k(G) \le n^{k-1}$ *for any graph G on n vertices.*
2. *[Kiefer, Schweitzer 16]* $D^3(G) = O(n^2/\log n)$.

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

For example,

$W(K_n) = n + 1$ because $W(K_n, K_{n+1}) = n + 1$.
(hence, $W(G) \leq D(G) \leq n + 1$ cannot be better)

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

Logic with counting quantifiers ($\mathrm{FO}_{\#}$, $\mathrm{FO}^{k}_{\#}$)

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

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

Example

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

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

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

Exercise

1. Define $K_{1,n}$ in $\mathrm{FO}^2_{\#}$.
2. Define P_n in $\mathrm{FO}^2_{\#}$.

Counting move in the Ehrenfeucht game

- Spoiler exhibits a set A of "good" vertices in G or H.
- Duplicator responds with B in the other graph such that $|B| = |A|$.
- Spoiler selects $b \in B$ and puts a pebble on it.
- Duplicator selects $a \in A$ and puts the other pebble on it.

Counting move in the Ehrenfeucht game

- Spoiler exhibits a set A of "good" vertices in G or H.
- Duplicator responds with B in the other graph such that $|B| = |A|$.
- Spoiler selects $b \in B$ and puts a pebble on it.
- Duplicator selects $a \in A$ and puts the other pebble on it.

Exercise

Let $\Delta(G)$ denote the maximum degree of a vertex in G. Assume that $\Delta(G) \neq \Delta(H)$. Prove that $D^2_{\#}(G, H) \leq 2$.

Outline

References

- Neil Immerman. *Descriptive Complexity*. Springer, 1999.
- Oleg Pikhurko and Oleg Verbitsky. Logical complexity of graphs: a survey. In: *Model Theoretic Methods in Finite Combinatorics*, J. Makowsky and M. Grohe Eds. Contemporary Mathematics, vol. 558, Amer. Math. Soc., Providence, RI, pp. 129–179, 2011.
- Sandra Kiefer and Pascal Schweitzer. Upper bounds on the quantifier depth for graph differentiation in first order logic. LICS'16.