

Logic and Automata

Lecture 3

Expressivness of MSO Graph Properties

Sebastian Maneth
NICTA & UNSW

Logic Summer School - Canberra, December 2006

Outline

How to prove that something can NOT be expressed

1. **Pumping Lemma** for regular languages
2. **Ehrenfeucht-Fraïssé games**

Recap

MSO = first-order logic (FO) + quantification over **sets**

FO theory of finite graphs: $\{ \phi \in \text{FO} \mid g \models \phi \text{ for every finite graph } g \}$
is undecidable.

MSO: If **C** contains infinitely many square grids,
then MSO theory of **C** is undecidable.

→ **Restrict class C**, so that MSO theory becomes decidable!

(1) **C = STRINGS**

Theorem (Büchi) *Set of strings is **MSO definable** if and only if it is accepted by a **finite-state automaton**.*

Corollary ***MSO theory** of **strings** is decidable.*

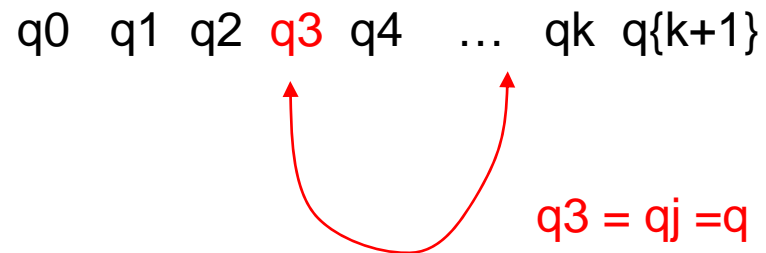
1. Pumping Lemma

A = deterministic FS automaton with k states

$L(A)$ infinite.

If $\text{length}(w) > k$, then there is **repetition of states**

q : first state that repeats



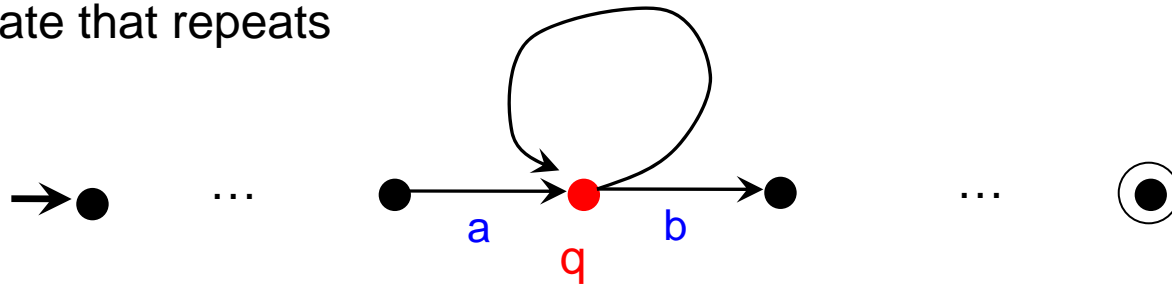
1. Pumping Lemma

A = deterministic FS automaton with k states

$L(A)$ infinite.

If $\text{length}(w) > k$, then there is **repetition of states**

q : first state that repeats



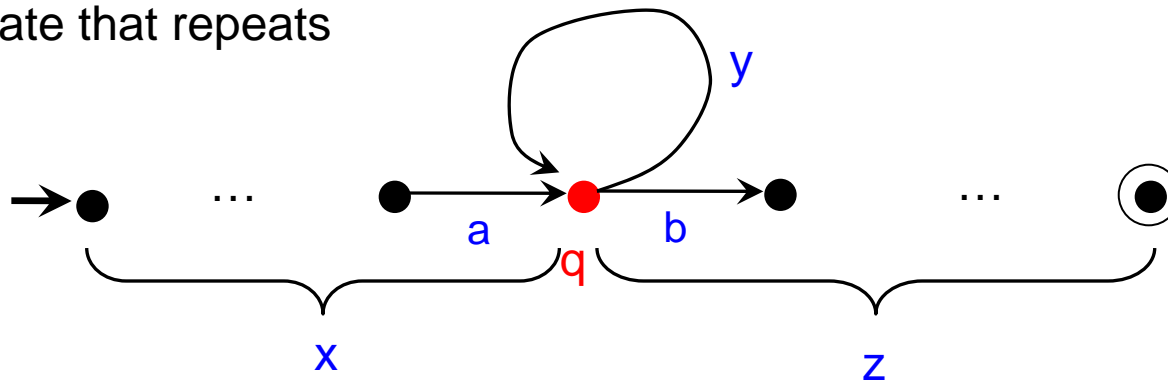
1. Pumping Lemma

A = deterministic FS automaton with k states

L(A) infinite.

If $\text{length}(w) > k$, then there is **repetition of states**

q : first state that repeats



$$w = xyz$$

Observations

1. $|xy| \leq k$

2. $|y| \geq 1$

3. $xz \in L(A), xyyz \in L(A), xyyyz \in L(A) \dots$

for all $m \geq 0$: $xy^m z \in L(A)$

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$

we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$
we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

Use PL to prove that $C = \{a^n b^n \mid n \geq 0\}$ is **NOT** regular.

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$
we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

Use PL to prove that $C = \{a^n b^n \mid n \geq 0\}$ is **NOT** regular.

Assume C is regular, and let k be as in lemma.

$$w = xyz = a^n b^n \quad \text{and} \quad |y| \geq 1$$

case 1 y contains only a 's. Then $xz = a^s b^n$ with $s < n$, i.e., w NOT in L .

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$
we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

Use PL to prove that $C = \{a^n b^n \mid n \geq 0\}$ is **NOT** regular.

Assume C is regular, and let k be as in lemma.

$$w = xyz = a^n b^n \quad \text{and} \quad |y| \geq 1$$

case 1 y contains only a 's. Then $xz = a^s b^n$ with $s < n$, i.e., w NOT in L .

case 2 $y = a^s b^v$ with $s, v \geq 1$. Then $xy^2z = a^n b^v a^s b^n$

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$
we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

Use PL to prove that $C = \{a^n b^n \mid n \geq 0\}$ is **NOT** regular.

Assume C is regular, and let k be as in lemma.

$$w = xyz = a^n b^n \quad \text{and} \quad |y| \geq 1$$

case 1 y contains only a 's. Then $xz = a^s b^n$ with $s < n$, i.e., w NOT in L .

case 2 $y = a^s b^v$ with $s, v \geq 1$. Then $xy^2z = a^n b^v a^s b^n$

case 3 $y = b^v$ with $v \geq 1$. Then $xz = a^n b^{n-v}$

1. Pumping Lemma

Let L be infinite regular language (i.e. accepted by some automaton)

There exists a k such that for every $w \in L$ with $|w| > k$
we can write $w = xyz$ with $|xy| < k$, $|y| > 0$ such that

$$xy^mz \in L, \text{ for } m = 0, 1, 2, \dots$$

Use PL to prove that $C = \{a^n b^n \mid n \geq 0\}$ is **NOT** regular.

Assume C is regular, and let k be as in lemma.

$$w = xyz = a^n b^n \quad \text{and} \quad |y| \geq 1$$

case 1 y contains only a 's. Then $xz = a^s b^n$ with $s < n$, i.e., w NOT in L .

case 2 $y = a^s b^v$ with $s, v \geq 1$. Then $xy^2z = a^n b^v a^s b^n$

case 3 $y = b^v$ with $v \geq 1$. Then $xz = a^n b^{n-v}$

→ By PL, C is not regular.

1. Pumping Lemma

$C = \{ a^n b^n \mid n \geq 0 \}$ is not regular.

By Büchi's Theorem: C is not MSO definable.

We can use this to prove that certain *graph properties* are not MSO definable!

1. Pumping Lemma

$C = \{ a^n b^n \mid n \geq 0 \}$ is not regular.

By Büchi's Theorem: C is not MSO definable.

We can use this to prove that certain *graph properties* are not MSO definable!

Note By same proof as for C ,

$$E = \{ w \in \{a,b\}^* \mid \#_a(w) = \#_b(w) \}$$

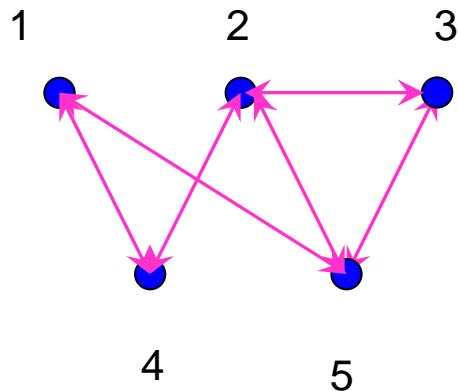
is NOT regular.

1. Pumping Lemma

$$E = \{ w \in \{a,b\}^* \mid \#_a(w) = \#_b(w) \}$$

is NOT definable in MSO.

Example existence of **directed Hamiltonian cycle**

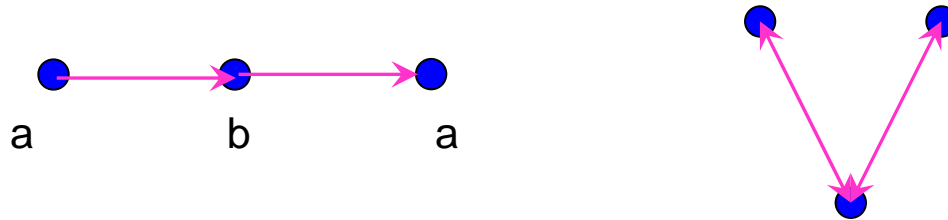


1, 4, 2, 3, 5, 1

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

Proof.

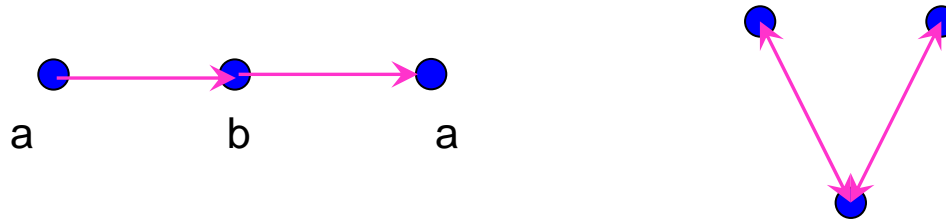


with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w ,
and edge from n to m iff n is labeled a and m is labeled b , or vice versa

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

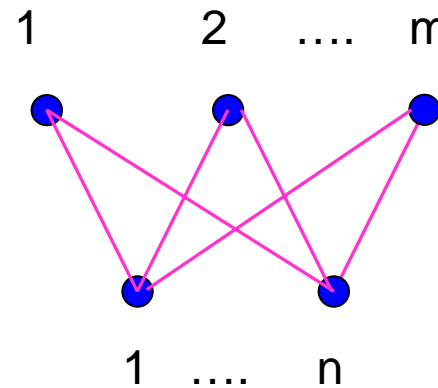
Proof.



with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w , and edge from n to m iff n is labeled a and m is labeled b , or vice versa.

$\rightarrow K_w$ is complete bipartite graph

K_w is Hamiltonian **iff** $w \in E$

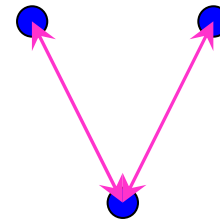
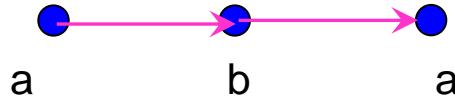


FullBip
But not **hc!**

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

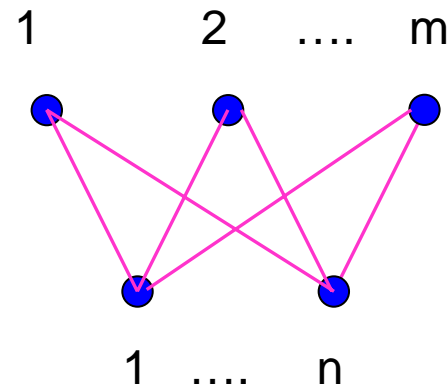
Proof.



with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w , and edge from n to m iff n is labeled a and m is labeled b , or vice versa.

$\rightarrow K_w$ is complete bipartite graph

K_w is Hamiltonian **iff** $w \in E$



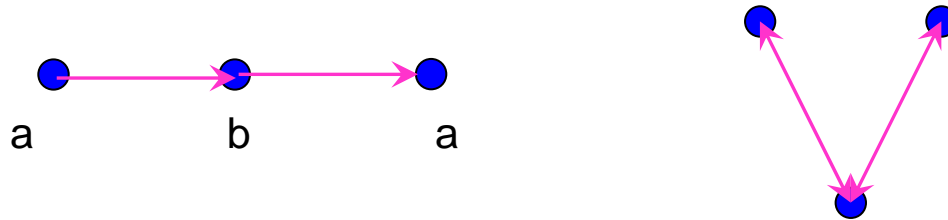
FullBip
But not **hc!**

Assume there is MSO formula **hc**.

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

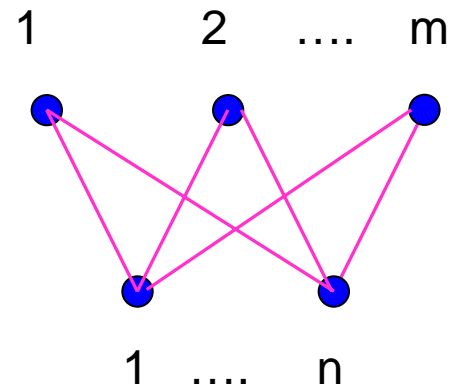
Proof.



with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w , and edge from n to m iff n is labeled a and m is labeled b , or vice versa.

$\rightarrow K_w$ is complete bipartite graph

K_w is Hamiltonian **iff** $w \in E$



FullBip
But not **hc!**

Assume there is MSO formula **hc**.

Let

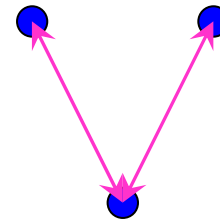
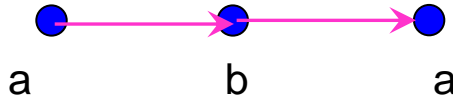
$$\eta(x, y) = (\text{lab}(a,x) \wedge \text{lab}(b,y)) \vee (\text{lab}(a,y) \wedge \text{lab}(b,x))$$

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

Proof.

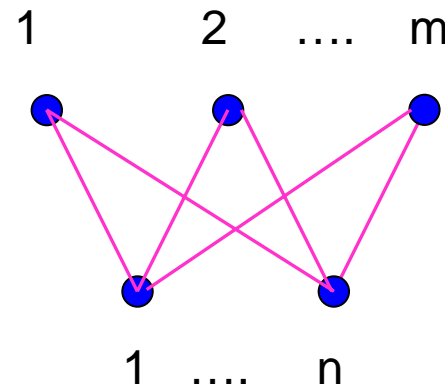
STR(aba) =



with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w , and edge from n to m iff n is labeled a and m is labeled b , or vice versa.

$\rightarrow K_w$ is complete bipartite graph

K_w is Hamiltonian **iff** $w \in E$



FullBip
But not **hc!**

Assume there is MSO formula **hc**.

Let

$$\eta(x, y) = (\text{lab}(a,x) \wedge \text{lab}(b,y)) \vee (\text{lab}(a,y) \wedge \text{lab}(b,x))$$

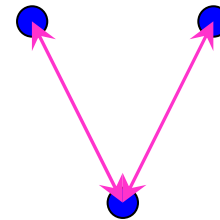
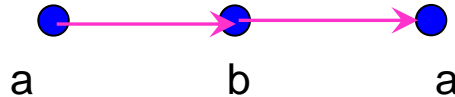
$$w \in E \text{ iff } K_w \text{ is Hamiltonian iff } \text{STR}(w) \models \text{hc}[\text{edge}(x,y) \leftarrow \eta(x, y)]$$

1. Pumping Lemma

Lemma Whether a graph has a **directed Hamiltonian Cycle** is *not* MSO-definable.

Proof.

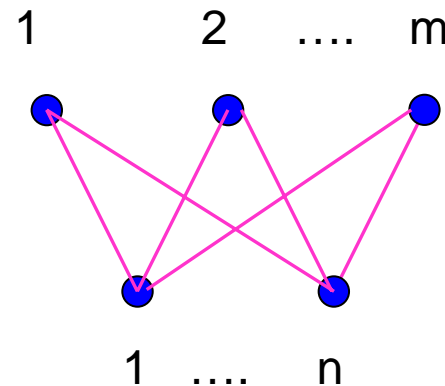
STR(aba) =



with $w \in \{a, b\}^*$ associate **graph** K_w with same set of nodes as w , and edge from n to m iff n is labeled a and m is labeled b , or vice versa.

$\rightarrow K_w$ is complete bipartite graph

K_w is Hamiltonian **iff** $w \in E$



FullBip
But not **hc!**

Assume there is MSO formula **hc**.

Let

$$\eta(x, y) = (\text{lab}(a,x) \wedge \text{lab}(b,y)) \vee (\text{lab}(a,y) \wedge \text{lab}(b,x))$$

$$w \in E \text{ iff } K_w \text{ is Hamiltonian iff } \text{STR}(w) \models \text{hc}[\text{edge}(x,y) \leftarrow \eta(x, y)]$$

By Büchi this would imply that E is regular. **CONTRADITON!**

1. Pumping Lemma

Examples of graph properties **NOT definable in MSO**

- (1) two sets X and Y have equal cardinality
- (2) in a directed graph, X is the set of nodes of a path from x to y
- (3) a directed graph is Hamiltonian

For (1) use language E and similar argument. If $\psi(X,Y)$ expresses $|X|=|Y|$, then replace $u \in X$ by $\text{lab}(a,u)$ and $u \in Y$ by $\text{lab}(b,u)$. (gives ϕ)
Now $\text{STR}(w) \models \phi$ iff $w \in E$.

1. Pumping Lemma

Examples of graph properties **NOT definable in MSO**

- (1) two sets X and Y have equal cardinality
- (2) in a directed graph, X is the set of nodes of a path from x to y
- (3) a directed graph is Hamiltonian

For (1) use language E and similar argument. If $\psi(X,Y)$ expresses $|X|=|Y|$, then replace $u \in X$ by $\text{lab}(a,u)$ and $u \in Y$ by $\text{lab}(b,u)$. (gives ϕ)
Now $\text{STR}(w) \models \phi$ iff $w \in E$.

For (2), assume we have $\phi(x, y, X)$ expressing that X is the set of nodes of a directed path from x to y . Then

$$(\exists X) [(\forall u) (u \in X) \wedge (\exists x)(\exists y) (\phi(x,y,X) \wedge x \neq y \wedge \text{edge}(y,x))]$$

1. Pumping Lemma

Examples of graph properties **NOT definable in MSO**

- (1) two sets X and Y have equal cardinality
 - (2) in a directed graph, X is the set of nodes of a path from x to y
 - (3) a directed graph is Hamiltonian
-

For (1) use language E and similar argument. If $\psi(X,Y)$ expresses $|X|=|Y|$, then replace $u \in X$ by $\text{lab}(a,u)$ and $u \in Y$ by $\text{lab}(b,u)$. (gives ϕ)
Now $\text{STR}(w) \models \phi$ iff $w \in E$.

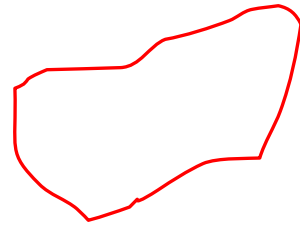
For (2), assume we have $\phi(x, y, X)$ expressing that X is the set of nodes of a directed path from x to y . Then

$$(\exists X) [(\forall u) (u \in X) \wedge (\exists x)(\exists y) (\phi(x,y,X) \wedge x \neq y \wedge \text{edge}(y,x))]$$

expresses that graph has ≥ 2 nodes and is Hamiltonian.

By (3), CONTRADICTION.

Use of Pumping Lemma



graph **g**
with property **P**

Assume existence of
MSO formula ϕ for **P**

$$g \models \phi \Leftrightarrow g \text{ fulfils } P$$



Easy substitution
on ϕ

ψ

$$\text{string } w \models \psi \Leftrightarrow w \in L$$

s.t. we *know* (\rightarrow use pumping lemma)
that **L is not regular**

m-equivalent structures

(finite) *relational structures*

- signature **Sig** (= finite set of relational symbols, each with an arity)
- (finite) domain D
- interpretation, maps relational symbol p of arity n to a function of type $D^n \rightarrow \{ \text{true}, \text{false} \}$

Notation

Relational structure \mathbf{g} , $\mathbf{v}=(v_1, \dots, v_n)$ n -tuple of elements in domain D

$\phi(\mathbf{x})$ = FO formula; at most variables $\mathbf{x}=(x_1, \dots, x_n)$ occur free

$(\mathbf{g}, \mathbf{v}) \models \phi(\mathbf{x})$ IFF $\phi(\mathbf{x})$ holds in \mathbf{g} where x_j is interpreted by v_j
--

m-equivalent structures

Relational structure \mathbf{g} , $\mathbf{v}=(v_1, \dots, v_n)$ n-tuple of elements of the domain

$\phi(\mathbf{x})$ = FO formula; at most variables $\mathbf{x}=(x_1, \dots, x_n)$ occur free

$(\mathbf{g}, \mathbf{v}) \models \phi(\mathbf{x})$ IFF $\phi(\mathbf{x})$ holds in \mathbf{g} where x_j is interpreted by v_j

$\text{qd}(\phi)$ = quantifier-depth of ϕ
(maximal # of nested quantifiers)

$\text{qd}(\phi) = 0$ if ϕ is atomic

$\text{qd}(\phi_1 \wedge \phi_2) = \text{qd}(\phi_1 \vee \phi_2) = \max(\text{qd}(\phi_1), \text{qd}(\phi_2))$

$\text{qd}(\neg\phi) = \text{qd}(\phi)$

$\text{qd}((\forall x)\phi) = \text{qd}((\exists x)\phi) = 1 + \text{qd}(\phi)$

e.g. quantifier-depth of

$(\exists x)[(\forall y) y \neq x \rightarrow (\exists x) \text{edge}(y, x)]$ equals 2

m-equivalent structures

Relational structure \mathbf{g} , $\mathbf{v}=(v_1,\dots,v_n)$ n-tuple of elements of the domain

$\phi(\mathbf{x})$ = FO formula; at most variables $\mathbf{x}=(x_1,\dots,x_n)$ occur free

$(\mathbf{g}, \mathbf{v}) \models \phi(\mathbf{x})$ IFF $\phi(\mathbf{x})$ holds in \mathbf{g} where x_j is interpreted by v_j

$\text{qd}(\phi)$ = quantifier-depth of ϕ
(maximal # of nested quantifiers)

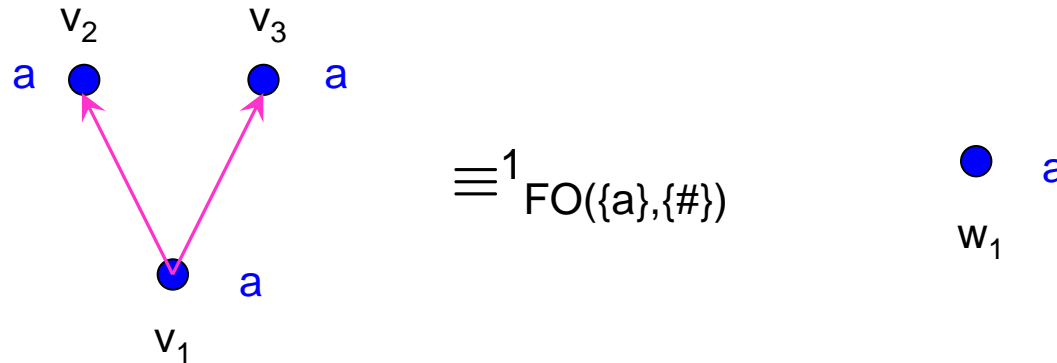
Definition Let $m \geq 0$, \mathbf{g} and \mathbf{h} relational structures over Sig .

(\mathbf{g}, \mathbf{u}) is **m-equivalent** to (\mathbf{h}, \mathbf{v}) if

$(\mathbf{g}, \mathbf{u}) \models \phi(\mathbf{x}) \Leftrightarrow (\mathbf{h}, \mathbf{v}) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

Notation $\equiv_{\text{FO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\Sigma,\Delta)}^m$

m-equivalent structures



Definition Let $m \geq 0$, g and h relational structures over Sig .

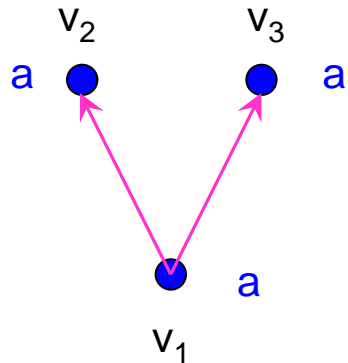
(g, u) is **m-equivalent** to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

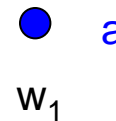
Notation $\equiv^m_{\text{FO}(\text{Sig})}$ $\equiv^m_{\text{MSO}(\text{Sig})}$ $\equiv^m_{\text{MSO}(\Sigma, \Delta)}$

m-equivalent structures

Note $(g, (v_1, v_2))$ **not** $\equiv_{\text{FO}(\{a, \{\#\})}^0 (h, (w_1, w_1))$
 $\rightarrow x_1 \neq x_2$



$\equiv_{\text{FO}(\{a, \{\#\})}^1$



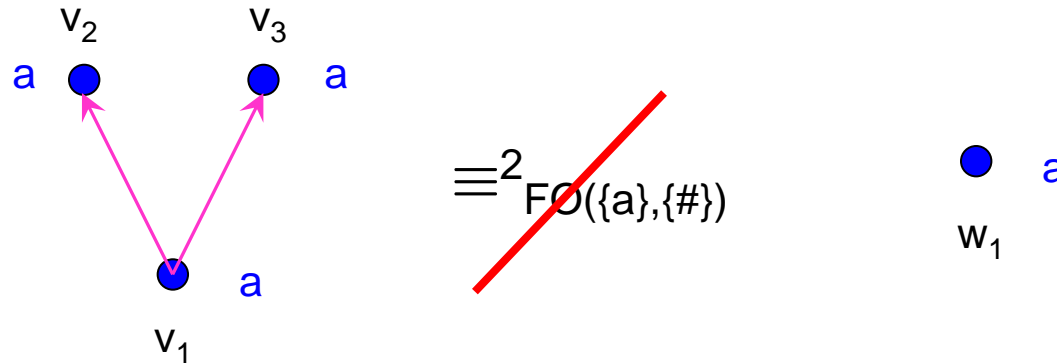
Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is **m-equivalent** to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

Notation $\equiv_{\text{FO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\Sigma, \Delta)}^m$

m-equivalent structures



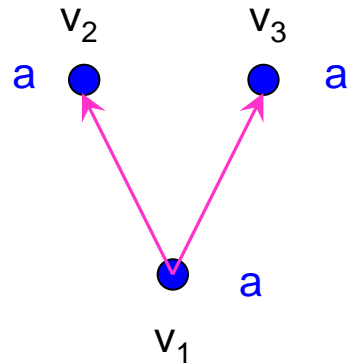
Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is **m-equivalent** to (h, v) if

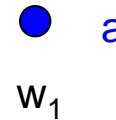
$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

Notation $\equiv^m_{FO(\text{Sig})}$ $\equiv^m_{MSO(\text{Sig})}$ $\equiv^m_{MSO(\Sigma, \Delta)}$

m-equivalent structures



~~$\equiv^2_{FO(\{a\},\{\#\})}$~~



$$qd((\exists x)(\exists y) \text{ edge}(x,y)) = 2$$

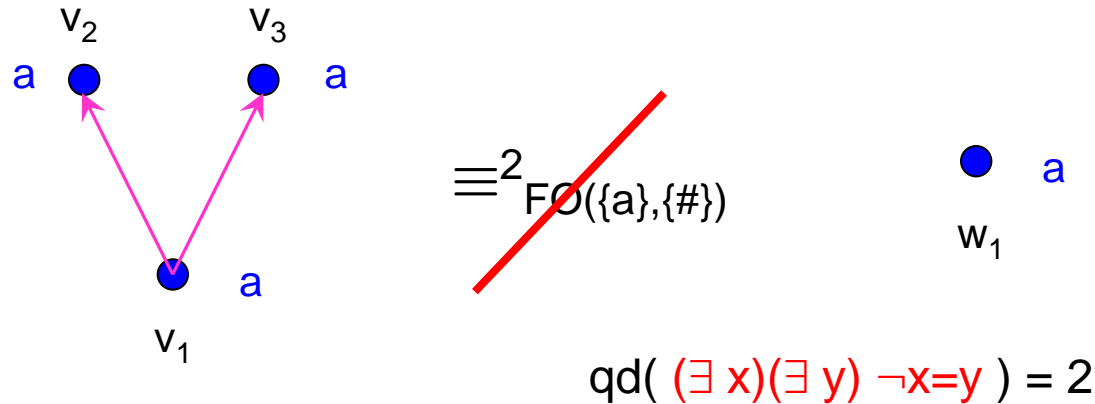
Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is m -equivalent to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $qd(\phi) \leq m$.

Notation $\equiv^m_{FO(\text{Sig})}$ $\equiv^m_{MSO(\text{Sig})}$ $\equiv^m_{MSO(\Sigma, \Delta)}$

m-equivalent structures



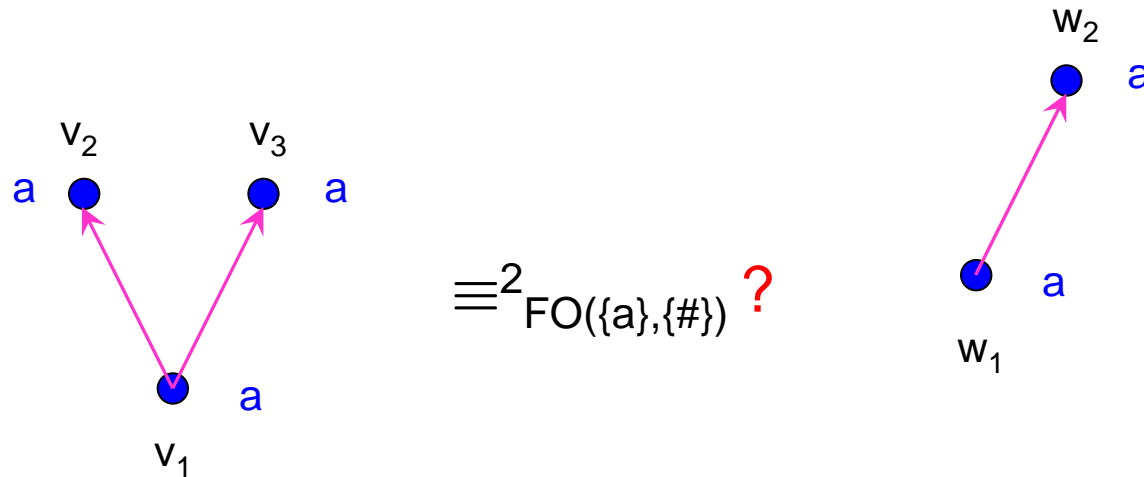
Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is m -equivalent to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $qd(\phi) \leq m$.

Notation $\equiv^m_{FO(\text{Sig})}$ $\equiv^m_{MSO(\text{Sig})}$ $\equiv^m_{MSO(\Sigma, \Delta)}$

m-equivalent structures



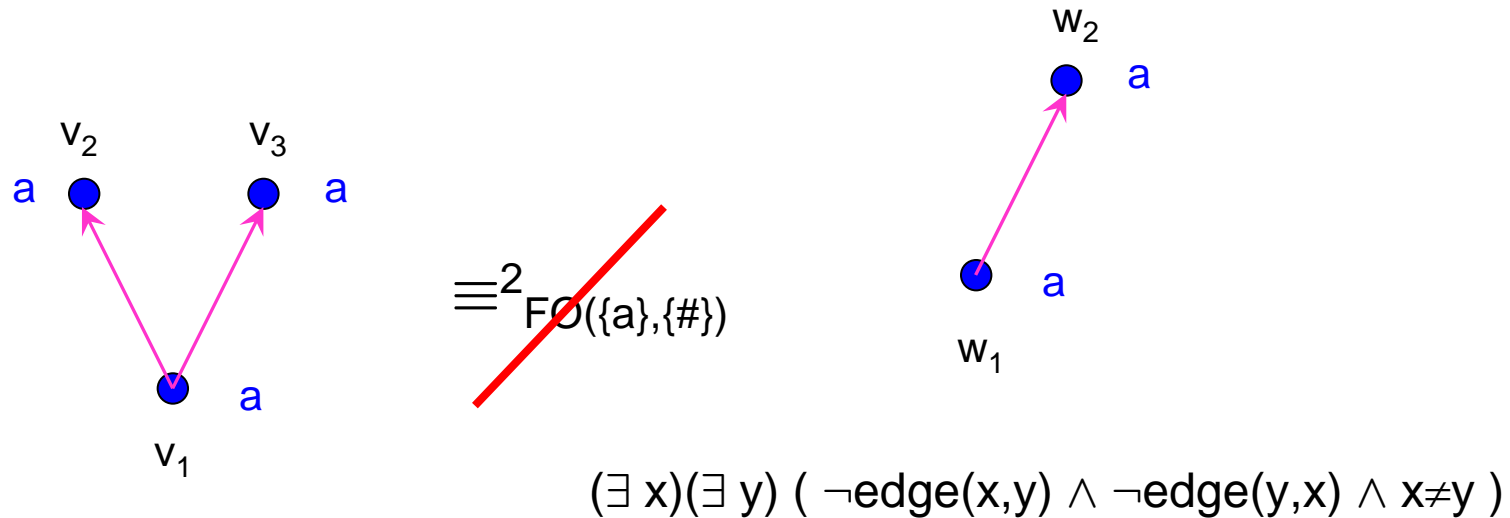
Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is **m-equivalent** to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

Notation $\equiv_{\text{FO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\text{Sig})}^m$ $\equiv_{\text{MSO}(\Sigma, \Delta)}^m$

m-equivalent structures



Definition Let $m \geq 0$, g and h relational structures over Sig .

(g, u) is m -equivalent to (h, v) if

$(g, u) \models \phi(\mathbf{x}) \Leftrightarrow (h, v) \models \phi(\mathbf{x})$ for all Sig -formulas ϕ with $\text{qd}(\phi) \leq m$.

Notation $\equiv_{FO(\text{Sig})}^m$ $\equiv_{MSO(\text{Sig})}^m$ $\equiv_{MSO(\Sigma, \Delta)}^m$

2. Ehrenfeucht-Fraïssé Games

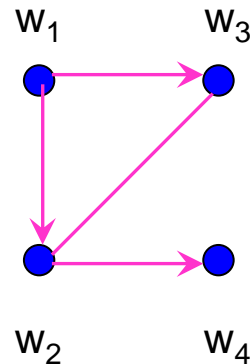
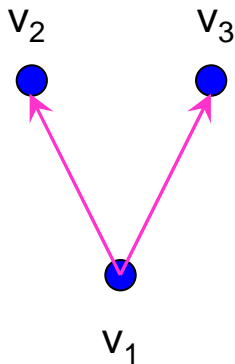
Graphs $g=(V,E)$, $h=(W,F)$

A partial function p from V to W is a **partial isomorphism** if

(1) $p(v_1) = p(v_2)$ implies $v_1 = v_2$ (injective)

(2) $\text{lab-}\sigma(p(v))$ implies $\text{lab-}\sigma(v)$

(3) $\text{edge-}\gamma(p(v_1), p(v_2))$ implies $\text{edge-}\gamma(v_1, v_2)$



2. Ehrenfeucht-Fraïssé Games

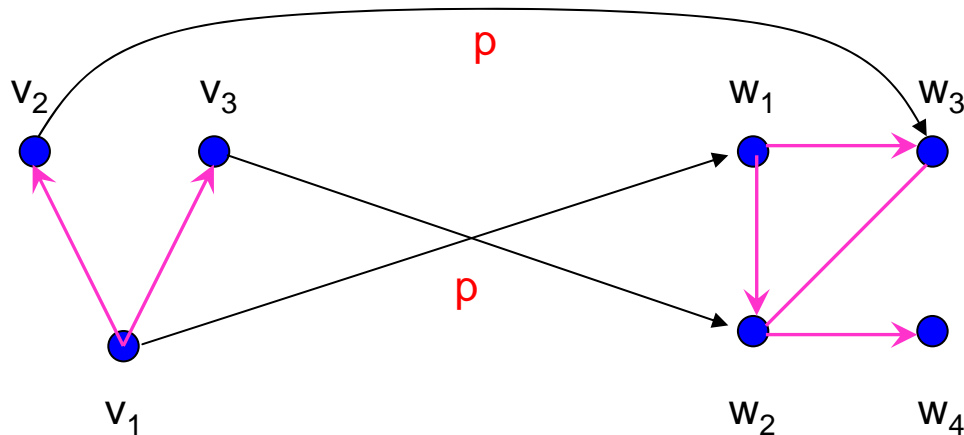
Graphs $g=(V,E)$, $h=(W,F)$

A partial function p from V to W is a **partial isomorphism** if

(1) $p(v_1) = p(v_2)$ implies $v_1 = v_2$ (injective)

(2) $\text{lab-}\sigma(p(v))$ implies $\text{lab-}\sigma(v)$

(3) $\text{edge-}\gamma(p(v_1), p(v_2))$ implies $\text{edge-}\gamma(v_1, v_2)$



$$p(v_1) = w_1, p(v_2) = w_3, p(v_3) = w_2$$

2. Ehrenfeucht-Fraïssé Games

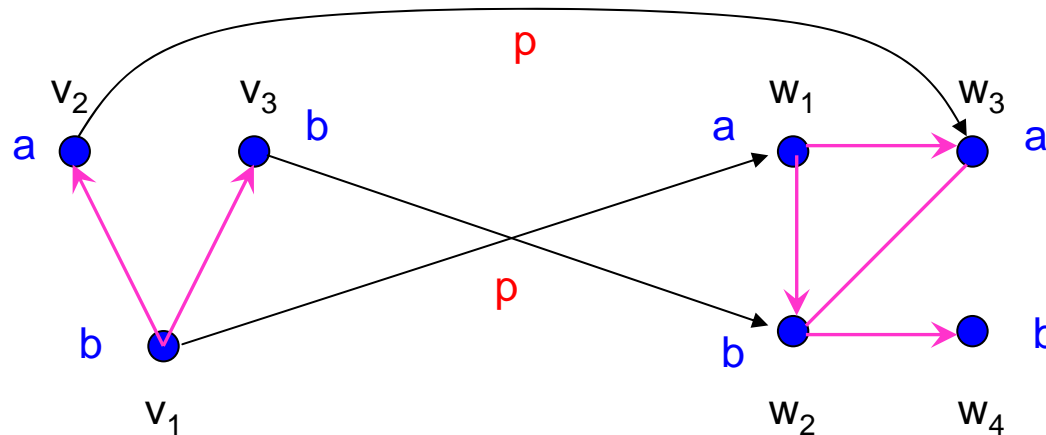
Graphs $g=(V,E)$, $h=(W,F)$

A partial function p from V to W is a **partial isomorphism** if

(1) $p(v_1) = p(v_2)$ implies $v_1 = v_2$ (injective)

(2) $\text{lab-}\sigma(p(v))$ implies $\text{lab-}\sigma(v)$

(3) $\text{edge-}\gamma(p(v_1), p(v_2))$ implies $\text{edge-}\gamma(v_1, v_2)$



$$p(v_1) = w_1, \quad p(v_2) = w_3, \quad p(v_3) = w_2$$

2. Ehrenfeucht-Fraïssé Games

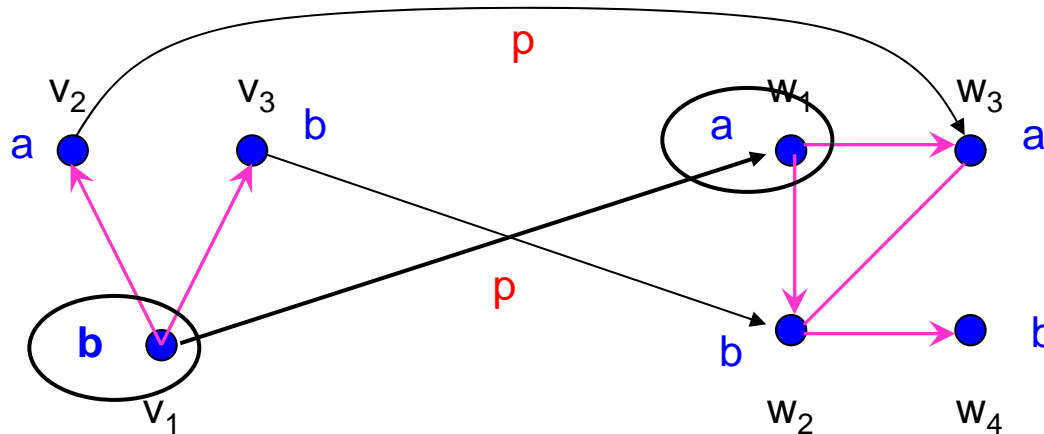
Graphs $g=(V,E)$, $h=(W,F)$

A partial function p from V to W is a **partial isomorphism** if

(1) $p(v_1) = p(v_2)$ implies $v_1 = v_2$ (injective)

(2) $\text{lab-}\sigma(p(v))$ implies $\text{lab-}\sigma(v)$

(3) $\text{edge-}\gamma(p(v_1), p(v_2))$ implies $\text{edge-}\gamma(v_1, v_2)$



$$p(v_1) = w_1, \quad p(v_2) = w_3, \quad p(v_3) = w_2$$

NOT a partial isomorphism!

2. Ehrenfeucht-Fraïssé Games

Graphs $g=(V,E)$, $h=(W,F)$

A partial function p from V to W is a **partial isomorphism** if

(1) $p(v_1) = p(v_2)$ implies $v_1 = v_2$ (injective)

(2) $\text{lab-}\sigma(p(v))$ implies $\text{lab-}\sigma(v)$

(3) $\text{edge-}\gamma(p(v_1), p(v_2))$ implies $\text{edge-}\gamma(v_1, v_2)$

g, h : **relational structures** over signature Sig

(2') $p(a)=c$ implies $a=c$ (for all constants c in Sig)

(3') $r(p(a_1), \dots, p(a_k))$ implies $f(a_1, \dots, a_k)$
(for every r in Sig of arity k)

“ p preserves all constants & relations”

2. Ehrenfeucht-Fraïssé Games

EF-game allows to verify the claim $(g, u) \equiv_m (h, v)$.

Two players: “**Spoiler**” and “**Duplicator**”

Players pick nodes from **g** and **h**, respectively.

→ **Spoiler** tries to show that **g** is NOT equivalent to **h**

→ **Duplicator** tries to prevent this (i.e., tries to show that **g** is equivalent to **h**)

If **Duplicator** wins (an m -round game), then $(g, u) \equiv_m (h, v)$

(otherwise, g and h are not m -equivalent and the game constitutes a counter-example formula which holds in one graph but not in the other)

2. Ehrenfeucht-Fraïssé Games

EF-game allows to verify the claim $(g, u) \equiv_m (h, v)$.

Game $G_m((g, u), (h, v))$

Initial configuration: $u \mapsto v$

A round: given configuration r .

first: **Spoiler** picks an element s of g or a t of h

next: **Duplicator** picks a t of h or an s of g , respectively

new configuration: $r \cup \{ (s,t) \}$

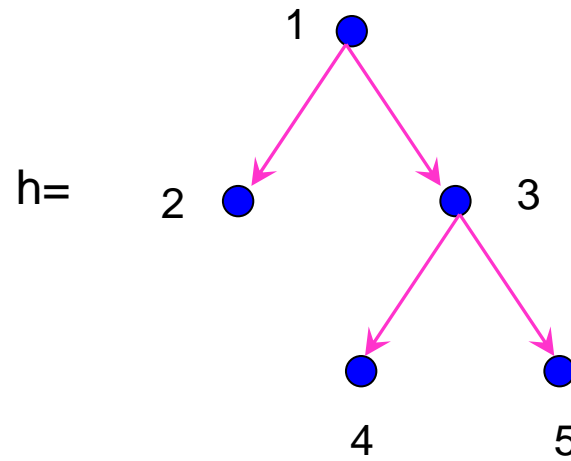
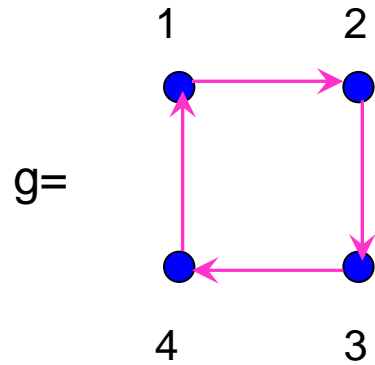
After m rounds,

Duplicator wins if final configuration is a *partial isomorphism*.

Otherwise Spoiler wins.

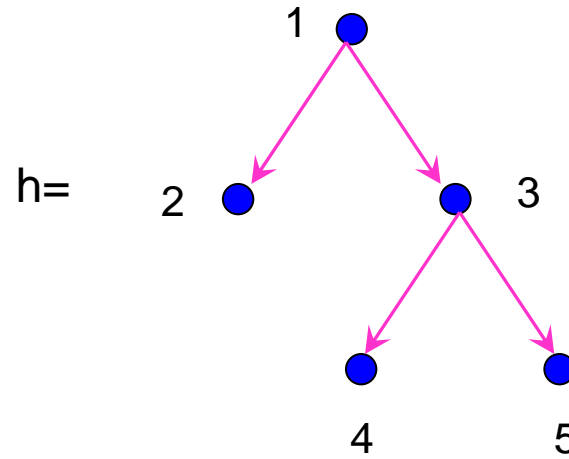
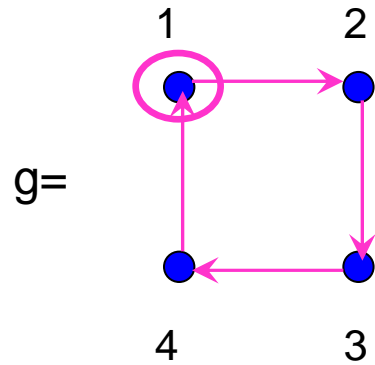
“Duplicator wins $G_m((g, u), (h, v))$ ” if it has a strategy to win each play.

2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

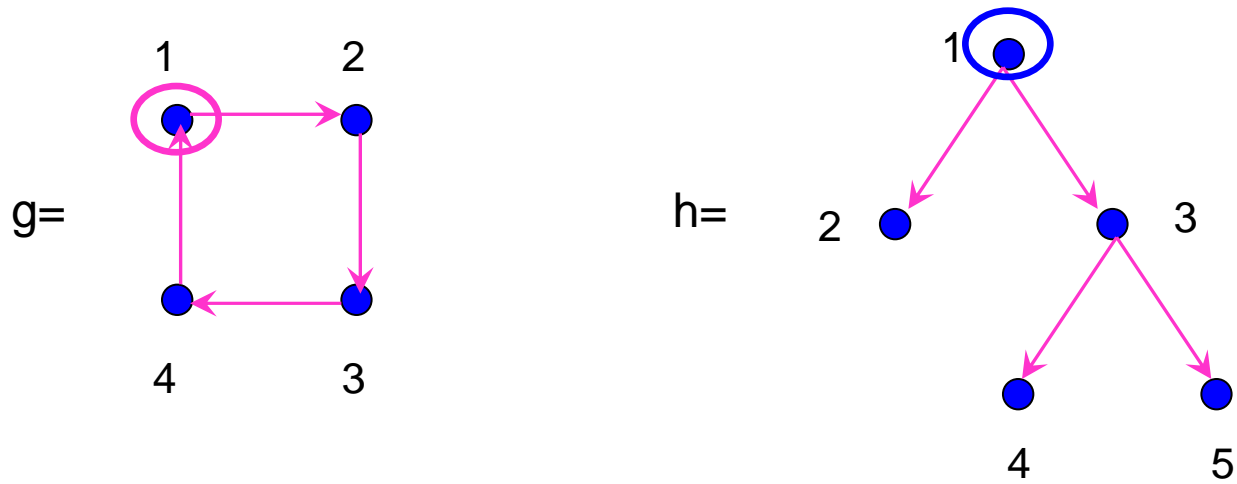
2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g.

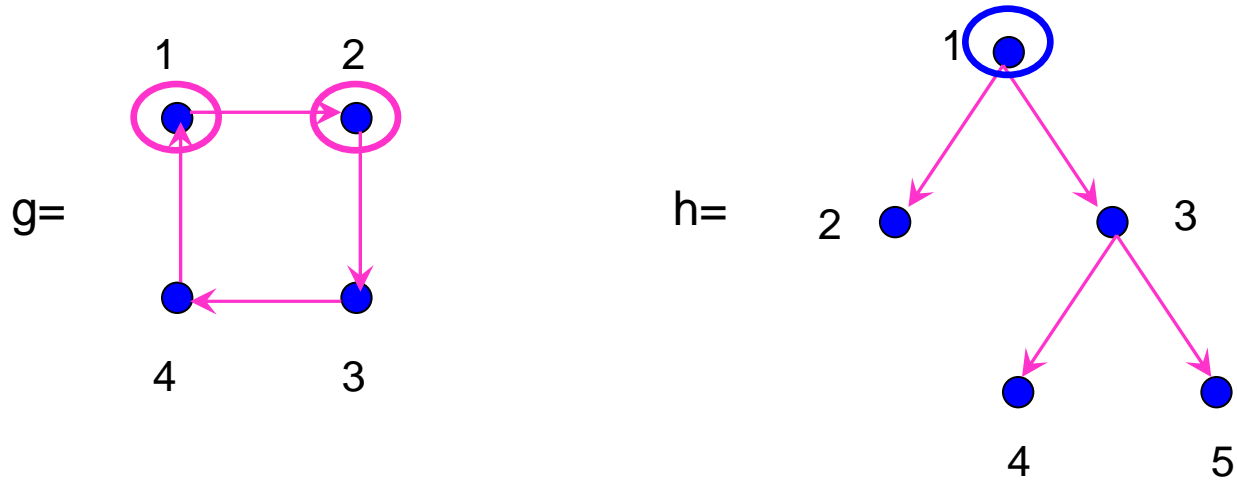
2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. **Duplicator** picks 1 in h. $r = \{(1,1)\}$

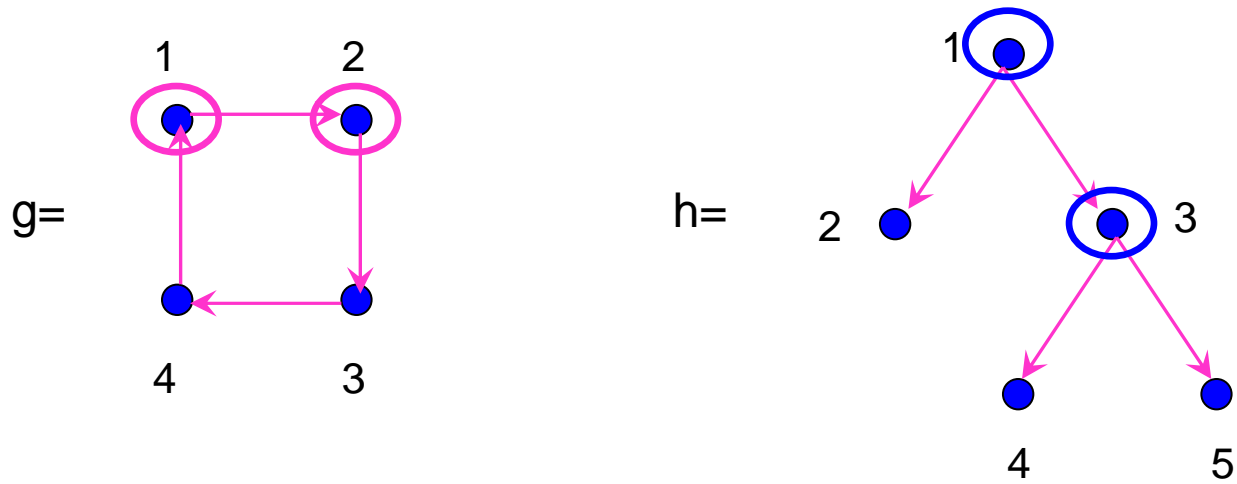
2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. Duplicator picks 1 in h. $r = \{(1,1)\}$
Spoiler picks 2 in g.

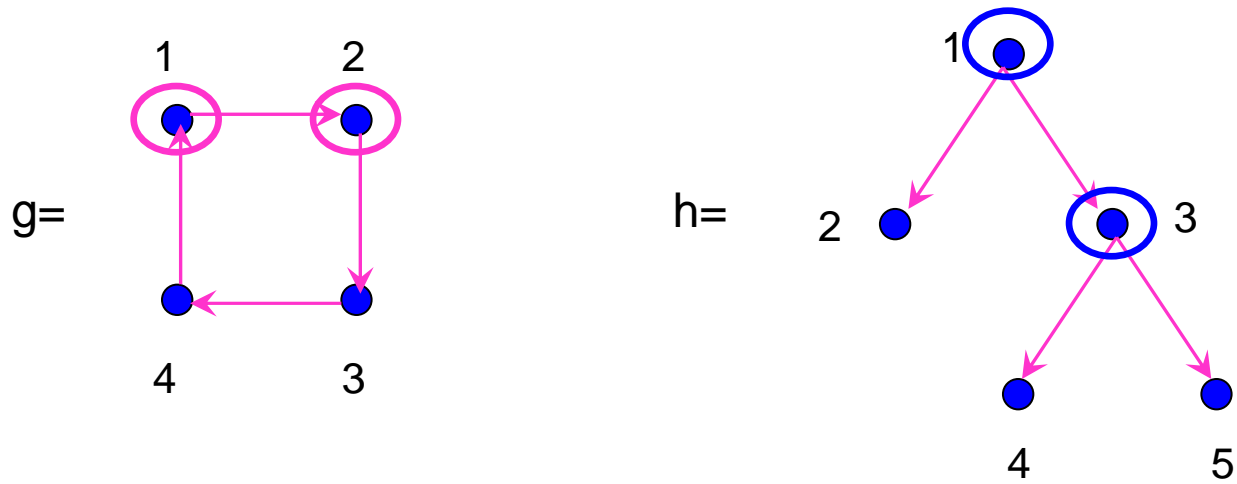
2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g.	Duplicator picks 1 in h.	$r = \{(1,1)\}$
Spoiler picks 2 in g.	Duplicator picks 3 in h.	$r = \{(1,1), (2,3)\}$

2. Ehrenfeucht-Fraïssé Games



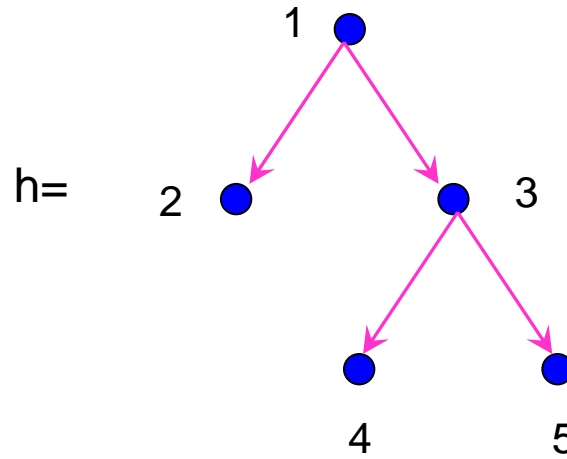
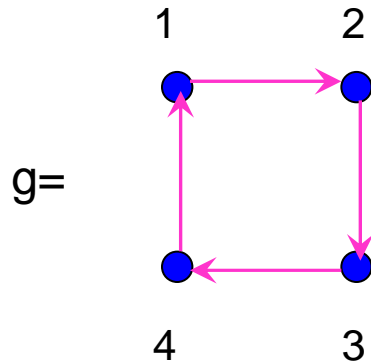
Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. Duplicator picks 1 in h. $r = \{(1,1)\}$
Spoiler picks 2 in g. Duplicator picks 3 in h. $r = \{(1,1), (2,3)\}$

Duplicator has won the 2-round game,
because r is partial isomorphism.

... but, he was lucky... 😊

2. Ehrenfeucht-Fraïssé Games



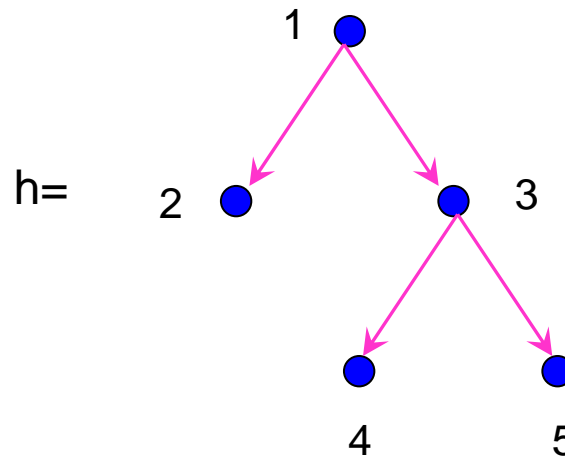
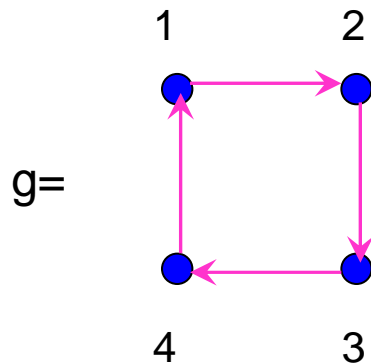
Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. **Duplicator** picks 1 in h. $r = \{(1,1)\}$

Spoiler picks 2 in g. **Duplicator** picks 3 in h. $r = \{(1,1), (2,3)\}$

Is there a *winning strategy* for **Duplicator** to win each 2-round game??

2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. **Duplicator** picks 1 in h. $r = \{(1,1)\}$

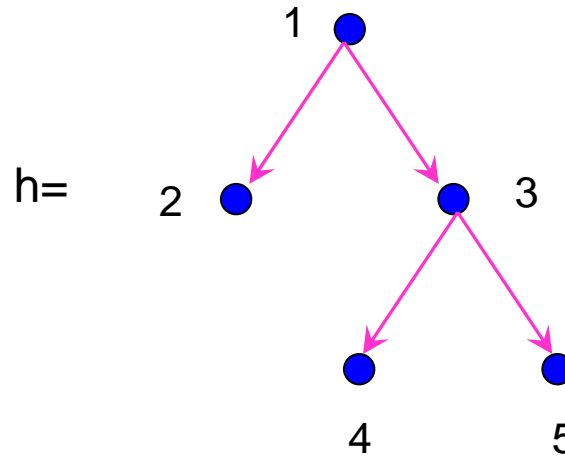
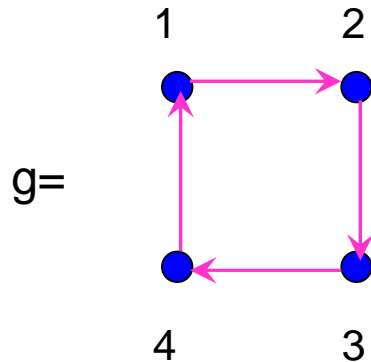
Spoiler picks 2 in g. **Duplicator** picks 3 in h. $r = \{(1,1), (2,3)\}$

Is there a *winning strategy* for **Duplicator** to win each 2-round game??

Yes! if spoiler starts in h, then any node in g will do.

if spoiler starts in g, then pick node 3!!

2. Ehrenfeucht-Fraïssé Games



Initial Configuration $r = \emptyset$

Spoiler picks 1 in g. Duplicator picks 1 in h. $r = \{(1,1)\}$

Spoiler picks 2 in g. Duplicator picks 3 in h. $r = \{(1,1), (2,3)\}$

Is there a *winning strategy* for Duplicator to win each 2-round game??

Yes! Thus, g is 2-equivalent to h!! $g \equiv_{\text{FO}(\text{gr})}^2 h$

EF Theorem

How to verify that Duplicator wins $G_m((g, u), (h, v))$?

For each $k = 0, \dots, m$ specify a set I_k of partial isomorphisms which allow Duplicator to win, with k rounds ahead.

(back property) $\forall p \in I_k \forall t \in h \exists s \in g$ such that $p \cup \{(s,t)\} \in I_{k-1}$
(forth property) $\forall p \in I_k \forall s \in g \exists t \in h$ such that $p \cup \{(s,t)\} \in I_{k-1}$

If such sequence I_m, \dots, I_0 exists, write $(g,s) \cong_m (h,t)$

Theorem For $m \geq 0$:

$(g,s) \equiv_m (h,t)$ iff
 $(g,s) \cong_m (h,t)$ iff
Duplicator wins $G_m((g,s),(h,t))$.

[Fraïssé1954]

[Ehrenfeucht1961]

2. Ehrenfeucht-Fraïssé Games

Example $G_2(\underline{u}, \underline{v})$

$u = a a b a a c a a$

$v = a a c a a b a a$

Initial configuration: \emptyset

Spoiler to move.

u, v : structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and $j+1$)

2. Ehrenfeucht-Fraïssé Games

Example $G_2(\underline{u}, \underline{v})$

$u = a a \boxed{b} a a c a a$

$v = a a c a a b a a$

Initial configuration: \emptyset

Spoiler to move.

u, v : structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and $j+1$)

2. Ehrenfeucht-Fraïssé Games

Example $G_2(\underline{u}, \underline{v})$

$u = a a \boxed{b} a a c a a$

$v = a a c a a \boxed{b} a a$

Initial configuration: \emptyset

Spoiler to move.

Duplicator must pick p in v with $b^v(p)$. \rightarrow configuration is $\{ (3, 6) \}$

u, v : structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and $j+1$)

2. Ehrenfeucht-Fraïssé Games

Example $G_2(\underline{u}, \underline{v})$

$u = a \ a \ \boxed{b} \ a \ a \ \boxed{c} \ a \ a$

$v = a \ a \ c \ a \ a \ \boxed{b} \ a \ a$

Initial configuration: \emptyset

Spoiler to move.

Duplicator must pick p in v with $b^v(p)$. \rightarrow configuration is $\{ (3, 6) \}$

Spoiler picks the c in u .

u, v : structures with domain $\{ 1, 2, 3, 4, 5, 6, 7, 8 \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and $j+1$)

Ehrenfeucht-Fraisse Games

Example

$i < j$
u = a a b a a c a a
v = a a c a a b a a

Initial configuration: \emptyset

Spoiler to move.

u, v: structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and j+1)

Duplicator must pick p in v with $b^v(p)$. \rightarrow configuration is $\{ (3, 6) \}$

Spoiler picks the c in u.

Duplicator loses!

(canNOT pick p' with $c^v(p')$ and $p < p'$)

Ehrenfeucht-Fraïssé Games

Example

$i < j$
 $u = a \ a \ \boxed{b} \ a \ a \ \boxed{c} \ a \ a$
 $v = a \ a \ c \ a \ a \ \boxed{b} \ a \ a$

Initial configuration: \emptyset

Spoiler to move.

u, v : structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Binary relation symbol: $<$

(less than on positions)

And binary relation symbol: next

(holds for j and $j+1$)

Duplicator must pick p in v with $b^v(p)$. \rightarrow configuration is $\{ (3, 6) \}$

Spoiler picks the c in u .

Duplicator loses!

(canNOT pick p' with $c^v(p')$ and $p < p'$)

Thus, u is *not* 2-equivalent to v .

$(\exists x)(\exists y)(x < y \wedge b(x) \wedge c(y))$ is true in u but false in v .

EF-Games in different Word Models

Example2, again $G_2(\underline{u}, \underline{v})$

$u = a a b a a c a a$

$v = a a c a a b a a$

u, v : structures with domain $\{ 1,2,3,4,5,6,7,8 \}$

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = a^v(1) = a^v(8) \dots = \text{true}$

Only binary relation symbol: succ
 $\text{succ}(j, j+1)$ for $j \in \{1, \dots, 7\}$

NOW, **Duplicator** has **winning strategy** for $G_2(u,v)$:

If **Spoiler** picks b- or c-position or adjacent position, then **Duplicator** picks corresponding position in the other string.

Otherwise (i.e., first or last position), **Duplicator** picks same position.

EF-Games in different Word Models

Example3

u = a a

v = a a a

u, v: structures with domain { 1,2, ... }

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = \text{true}$

Only < (less than on positions)

Spoiler wins $G_2(u, v)$

EF-Games in different Word Models

Example3

u = a a

v = a a a

u, v: structures with domain { 1,2, ... }

Unary relation symbols a,b,c

$a^u(1) = a^u(2) = \text{true}$

Only < (less than on positions)

Spoiler wins $G_2(u, v)$

(pick second a in v, and then react opposite to Duplicator's first move)

EF-Games in different Word Models

Example3

$u = a a$

$v = a a a$

u, v : structures with domain $\{ 1, 2, \dots \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = \text{true}$

Only $<$ (less than on positions)

Spoiler wins $G_2(u, v)$

(pick second a in v , and then react opposite to Duplicator's first move)

$u = \{ 1, 2, 3, 4, 5, 6, 7, 8 \}$ with $<$

$v = \{ 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$

EF-Games in different Word Models

Example3

$u = a a$

$v = a a a$

u, v : structures with domain $\{ 1, 2, \dots \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = \text{true}$

Only $<$ (less than on positions)

Spoiler wins $G_2(u, v)$

(pick second a in v , and then react opposite to **Duplicator**'s first move)

$u = \{ 1, 2, 3, 4, 5, 6, 7, 8 \}$ with $<$

$v = \{ 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$

→ **Spoiler** wins $G_4(u, v)$ / **Duplicator** wins $G_3(u, v)$!!

EF-Games in different Word Models

Example3

$u = a a$

$v = a a a$

u, v : structures with domain $\{ 1, 2, \dots \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = \text{true}$

Only $<$ (less than on positions)

Spoiler wins $G_2(u, v)$

(pick second a in v , and then react opposite to **Duplicator**'s first move)

$u = \{ 1, 2, 3, 4, 5, 6, 7, 8 \}$ with $<$

$v = \{ 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$

→ **Spoiler** wins $G_4(u, v)$ / **Duplicator** wins $G_3(u, v)$!!

$u = \{ 1, 2, \dots, 16 \}$

$v = \{ 1, 2, \dots, 17 \}$ → **Duplicator** wins $G_4(u, v)$.

EF-Games in different Word Models

Example3

$u = a a$

$v = a a a$

u, v : structures with domain $\{ 1, 2, \dots \}$

Unary relation symbols a, b, c

$a^u(1) = a^u(2) = \text{true}$

Only $<$ (less than on positions)

Spoiler wins $G_2(u, v)$

(pick second a in v , and then react opposite to **Duplicator's** first move)

$u = \{ 1, 2, 3, 4, 5, 6, 7, 8 \}$ with $<$

$v = \{ 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$

→ **Spoiler** wins $G_4(u, v)$!

$u = \{ 1, 2, \dots, 16 \}$

$v = \{ 1, 2, \dots, 17 \}$ → **Spoiler** wins $G_5(u, v)$.

$\forall m$

Duplicator wins

$G_m(u, v)$

where

$u = \{ 1, \dots, 2^m \}$

$v = \{ 1, \dots, 2^{m+1} \}$

EF Theorem

Duplicator wins $G_m(u, v)$ iff
u and v can NOT be distinguished by sentences of quantifier-depth m

u = a a b a a c a a

v = a a c a a b a a

$\exists x \exists y (b(x) \wedge x < y \wedge c(y))$

Theorem The language $\{a^n \mid n \text{ is even}\}$ is not FO definable.

Proof.

EF Theorem

Duplicator wins $G_m(u, v)$ iff
u and v can NOT be distinguished by sentences of quantifier-depth m

u = a a b a a c a a

v = a a c a a b a a

$\exists x \exists y (b(x) \wedge x < y \wedge c(y))$

Theorem The language $\{a^n \mid n \text{ is even}\}$ is not FO definable.

Proof. Assume there is ϕ of qd m, which only uses $<$.

Then ϕ is satisfied by a^{2^m}

EF Theorem

Duplicator wins $G_m(u, v)$ iff
 u and v can NOT be distinguished by sentences of quantifier-depth m

$u = a a \boxed{b} a a \boxed{c} a a$

$v = a a c a a \boxed{b} a a$

$\exists x \exists y (b(x) \wedge x < y \wedge c(y))$

Theorem The language $\{a^n \mid n \text{ is even}\}$ is not FO definable.

Proof. Assume there is ϕ of qd m , which only uses $<$.

Then ϕ is satisfied by a^{2^m}

But, $a^{2^m} \equiv_m a^{2^m+1}$

Hence, ϕ is also satisfied in the word a^{2^m+1} of ODD length. \rightarrow contradiction.

EF Theorem

Duplicator wins $G_m(u, v)$ iff
u and v can NOT be distinguished by sentences of quantifier-depth m

u = a a b a a c a a

v = a a c a a b a a

$\exists x \exists y (b(x) \wedge x < y \wedge c(y))$

Theorem The language $\{a^n \mid n \text{ is even}\}$ is not FO(<) definable.

Corollary $L = \{a^n \mid n \text{ is even}\}$ is not FO(succ) definable.

EF Theorem

Duplicator wins $G_m(u, v)$ iff
u and v can NOT be distinguished by sentences of quantifier-depth m

u = a a b a a c a a

v = a a c a a b a a

$\exists x \exists y (b(x) \wedge x < y \wedge c(y))$

Theorem The language $\{a^n \mid n \text{ is even}\}$ is not FO(<) definable.

Corollary $L = \{a^n \mid n \text{ is even}\}$ is not FO(succ) definable.

Proof. Assume there is FO(succ) formula ϕ for L.

In ϕ , replace each $\text{succ}(x, y)$ by $(x < y) \wedge \neg(\exists z) x < z < y$

Gives equivalent FO(<) formula. \rightarrow CONTRADICTION

EF Theorem

This was the EF game for **First-Order logic**.

How can we extend the game to **MSO**?

What will be an admissible move of a player?

→ What is the relationship of the **MSO**-game, to the pumping lemma for regular languages?

EF Theorem

This was the EF game for **First-Order logic**.

How can we extend the game to **MSO**?

What will be an admissible move of a player?

- Add “set”-moves (player selects a subset of nodes)
- Now, also \in and \subset relations must be preserved!

→ What is the relationship of the **MSO**-game, to the pumping lemma for regular languages?

NON definability Tools

In general, you can define **games a la EF**, for all kinds of logics, even much more expressive than MSO.

Similarly, you can define a **pumping lemma** for any kind of inductively defined generator system. (cf languages, indexed languages, etc)

Often “inside-conditions” become convoluted... ☹

→ Use reduction tools

For instance, “bridge theorem”

If $f(L) \in \text{BigClass}$, then $L \in \text{SmallClass}$.