Database Theory

VU 181.140, SS 2018

7. Ehrenfeucht-Fraïssé Games

Reinhard Pichler

Institut für Informationssysteme
Arbeitsbereich DBAI
Technische Universität Wien

15 May, 2018

Outline

Core of the slides by Christoph Koch, with kind permission

Using logic to express properties of structures

Definition

Let $\mathcal{L}$ be some logic (e.g., FO logic, (Monadic) SO logic, etc.).
We say that some property $\mathcal{P}$ of structures is expressible in $\mathcal{L}$ if there exists a sentence ϕ in $\mathcal{L}$, s.t. for all structures $\mathcal{A}$, the following equivalence holds:

$$\mathcal{A} \text{ has property } \mathcal{P} \text{ iff } \mathcal{A} \models \phi$$

Example

Property: "graph is closed w.r.t. transitivity":

This property is expressible in First-Order logic:

$$\phi = \forall x \forall y \forall z \big(e(x, y) \land e(y, z) \rightarrow e(x, z)\big)$$

Using logic to express properties of structures

Example

Property: "3-Colorability of a graph"

This property is expressible in Monadic Second-Order logic (MSO):

$$\exists X \exists Y \exists Z \big(partition(X, Y, Z) \wedge legal(X, Y, Z) \big) \text{ with}$$

$$partition(X, Y, Z) \equiv \forall v \big((v \in X \vee v \in Y \vee v \in Z) \wedge \neg(v \in X \wedge v \in Y) \wedge \neg(v \in X \wedge v \in Z) \wedge \neg(v \in Y \wedge v \in Z) \big)$$

$$legal(X, Y, Z) \equiv \forall u \forall v \big(e(u, v) \rightarrow (\neg(u \in X \wedge v \in X) \wedge \neg(u \in Y \wedge v \in Y) \wedge \neg(u \in Z \wedge v \in Z)) \big)$$

Remark. We shall provide tools to prove that 3-Colorability (of finite graphs) is not expressible in FO.

Motivation

- Goal: Inexpressibility proofs for FO queries.
- A standard technique for inexpressibility proofs from logic (model theory): Compactness theorem.
    - Discussed in logic lectures.
    - Fails if we are only interested in finite structures (=databases). The compactness theorem does not hold in the finite!
- We need a different technique to prove that certain queries are not expressible in FO.
- EF games are such a technique.

Inexpressibility via Compactness Theorem

Theorem (Compactness)

Let Φ be an infinite set of FO sentences and suppose that every finite subset of Φ is satisfiable. Then also Φ is satisfiable.

Definition

Property CONNECTED: Does there exists a (finite) path between any two nodes u, v in a given (possibly infinite) graph?

Theorem

CONNECTED is not expressible in FO, i.e., there does not exist an FO sentence ψ, s.t. for every structure $\mathcal{G}$ representing a graph, the following equivalence holds:

$$\text{Graph } \mathcal{G} \text{ is connected iff } \mathcal{G} \models \psi.$$

Proof.

Assume to the contrary that there exists an FO-formula ψ which expresses CONNECTED. We derive a contradiction as follows.

1. Extend the vocabulary of graphs by two constants c_1 and c_2 and consider the set of formulae $\Phi = \{\psi\} \cup \{\phi_n \mid n \geq 1\}$ with

$$\phi_n := \neg \exists x_1 \ldots \exists x_n \; x_1 = c_1 \wedge x_n = c_2 \wedge \bigwedge_{1 \leq i \leq n-1} E(x_i, x_{i+1}).$$

   ("There does not exist a path of length $n-1$ between c_1 and c_2".)
2. Clearly, Φ is unsatisfiable.
3. Consider an arbitrary, finite subset Φ_0 of Φ. There exists $n_{\max}$, s.t. $\phi_m \notin \Phi_0$ for all $m > n_{\max}$.
4. Φ_0 is satisfiable: indeed, a single path of length $n_{\max} + 1$ (where we interpret c_1 and c_2 as the endpoints of this path) satisfies Φ_0.
5. By the Compactness Theorem, Φ is satisfiable, which contradicts the observation (2) above. Hence, ψ cannot exist. □

Compactness over Finite Models

Question. Does the theorem also establish that connectedness of finite graphs is FO inexpressible? The answer is "no"!

Proposition

Compactness fails over finite models, i.e., there exists a set Φ of FO sentences with the following properties:

- every finite subset of Φ has a finite model and
- Φ has no finite model.

Proof.

Consider the set $\Phi = \{d_n \mid n \geq 2\}$ with $d_n := \exists x_1 \ldots \exists x_n \bigwedge_{i \neq j} x_i \neq x_j$,
i.e., $d_n \Leftrightarrow$ there exist at least n pairwise distinct elements.

Clearly, every finite subset $\Phi_0 = \{d_{i_1}, \ldots, d_{i_k}\}$ of Φ has a finite model: just take a set whose cardinality exceeds $\max(\{i_1, \ldots, i_k\})$.
However, Φ does not have a finite model. $\square$

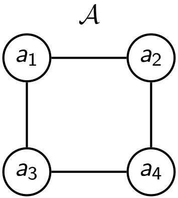
Rules of the EF game

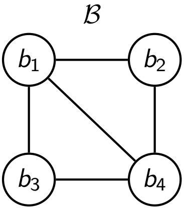
- Two players: Spoiler S, Duplicator D.
- "Game board": Two structures of the same schema.
- Players move alternatingly; Spoiler starts (like in chess).
- The number of moves k to be played is fixed in advance (differently from chess).
- Tokens $S_1, \ldots, S_k, D_1, \ldots, D_k$.
- In the i-th move, Spoiler first selects a structure and places token S_i on a domain element of that structure. Next, Duplicator places token D_i on an arbitrary domain element of the other structure. (That's one move, not two.)
- Spoiler may choose its structure anew in each move. Duplicator always has to answer in the other structure.
- A token, once placed, cannot be (re)moved.
- The winning condition follows a bit later.

Notation from Finite Model Theory

- $\mathcal{A}, \mathcal{B}$ denote structures (=databases),
- $|\mathcal{A}|$ is the domain of a structure $\mathcal{A}$,
- $E^{\mathcal{A}}$ is the relation E of a structure $\mathcal{A}$.

A game run with $k = 3$





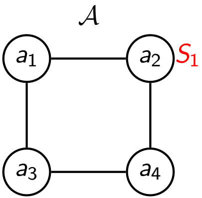
$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

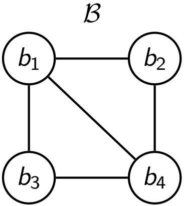
$\|\mathcal{A}\|$	
	a_1
	a_2
	a_3
	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\|\mathcal{B}\|$	
	b_1
	b_2
	b_3
	b_4

A game run with $k = 3$





$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
	a_3
	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
	b_1
	b_2
	b_3
	b_4

A game run with $k = 3$

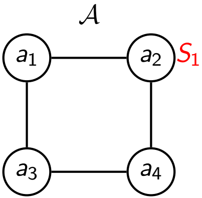
$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

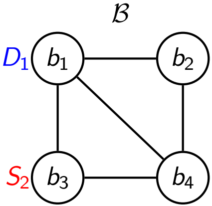
$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
	a_3
	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
D_1	b_1
	b_2
	b_3
	b_4

A game run with $k = 3$





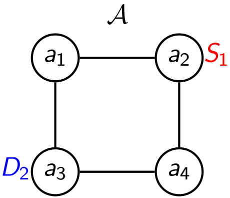
$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

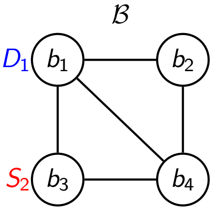
$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
	a_3
	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
D_1	b_1
	b_2
S_2	b_3
	b_4

A game run with $k = 3$





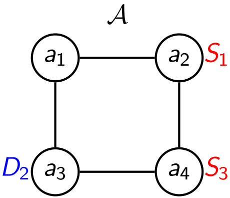
$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

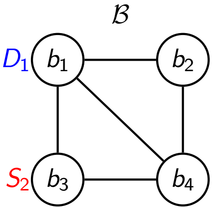
$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
D_2	a_3
	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
D_1	b_1
	b_2
S_2	b_3
	b_4

A game run with $k = 3$





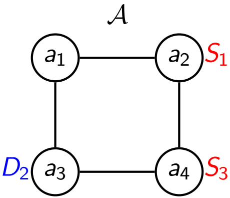
$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

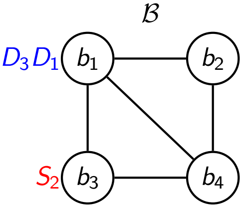
$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
D_2	a_3
S_3	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
D_1	b_1
	b_2
S_2	b_3
	b_4

A game run with $k = 3$





$E^{\mathcal{A}}$		
	a_1	a_2
	a_2	a_1
	$\vdots$	$\vdots$
	a_4	a_3

$\lvert\mathcal{A}\rvert$	
	a_1
S_1	a_2
D_2	a_3
S_3	a_4

$E^{\mathcal{B}}$		
	b_1	b_2
	b_2	b_1
	$\vdots$	$\vdots$
	b_4	b_3
	b_1	b_4
	b_4	b_1

$\lvert\mathcal{B}\rvert$	
$D_3\,D_1$	b_1
	b_2
S_2	b_3
	b_4

Partial isomorphisms

Definition

- $\mathcal{A}|_S$: Restriction of a structure $\mathcal{A}$ to the subdomain $S \subseteq |\mathcal{A}|$. Same schema; for each relation $R^{\mathcal{A}}$:

$$R^{\mathcal{A}|_S} := \{\langle a_1, \ldots, a_k \rangle \in R^{\mathcal{A}} \mid a_1, \ldots, a_k \in S\}.$$

- A partial function $\theta : |\mathcal{A}| \to |\mathcal{B}|$ is a partial isomorphism from $\mathcal{A}$ to $\mathcal{B}$ if and only if θ is an isomorphism from $\mathcal{A}|_{\mathrm{dom}(\theta)}$ to $\mathcal{B}|_{\mathrm{rng}(\theta)}$.
- This definition assumes that the schema of $\mathcal{A}$ does not contain any constants but is purely relational.

Partial isomorphisms

Example

$R^{\mathcal{A}}$			
	1	2	3
	2	1	4

$\lvert\mathcal{A}\rvert$	
	1
	2
	3
	4

$R^{\mathcal{B}}$			
	a	b	c
	a	b	d

$\lvert\mathcal{B}\rvert$	
	a
	b
	c
	d

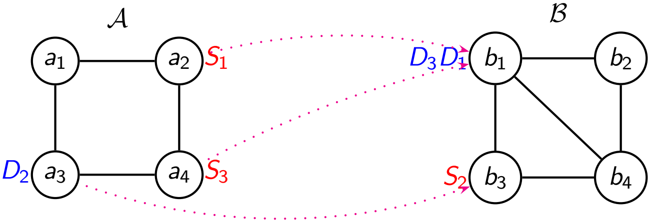
$$\theta : \begin{cases} 1 \mapsto a \\ 2 \mapsto b \\ 3 \mapsto c \end{cases}$$

$R^{\mathcal{A}}\vert_{\{1,2,3\}}$			
	1	2	3

$R^{\mathcal{B}}\vert_{\{a,b,c\}}$			
	a	b	c

θ is a partial isomorphism.

Partial isomorphisms

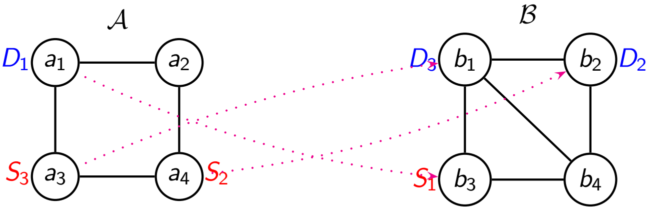


The partial function $\theta : |\mathcal{A}| \to |\mathcal{B}|$ with

$$\theta : \begin{cases} a_2 \mapsto b_1 \\ a_3 \mapsto b_3 \\ a_4 \mapsto b_1 \end{cases}$$

is **not** a partial isomorphism: $\mathcal{A} \models a_2 \neq a_4$, $\mathcal{B} \not\models \theta(a_2) \neq \theta(a_4)$.

Partial isomorphisms

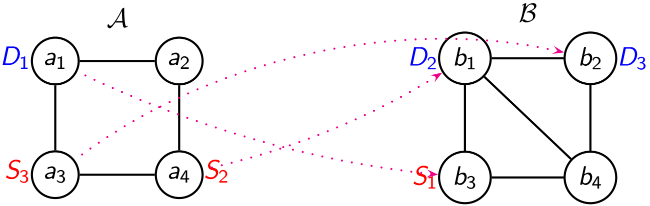


The partial function $\theta : |\mathcal{A}| \rightarrow |\mathcal{B}|$ with

$$\theta : \begin{cases} a_1 \mapsto b_3 \\ a_4 \mapsto b_2 \\ a_3 \mapsto b_1 \end{cases}$$

is a partial isomorphism.

Partial isomorphisms



The partial function $\theta : |\mathcal{A}| \to |\mathcal{B}|$ with

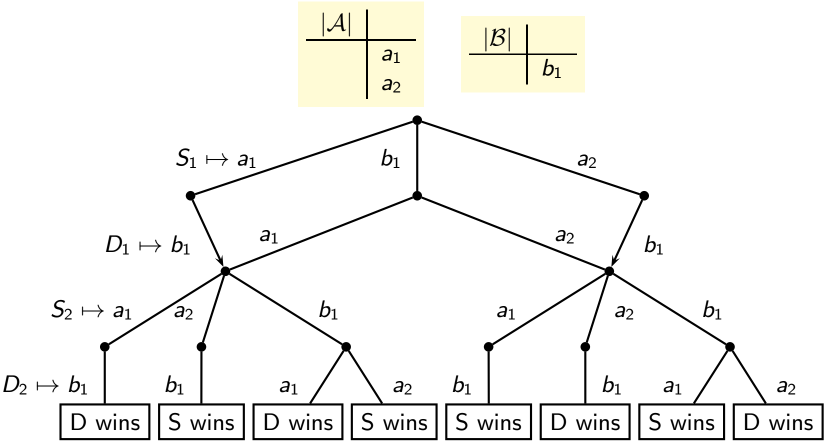
$$\theta : \begin{cases} a_1 \mapsto b_3 \\ a_4 \mapsto b_1 \\ a_3 \mapsto b_2 \end{cases}$$

is not a partial isomorphism: $\mathcal{A} \models E(a_1, a_3)$, $\mathcal{B} \not\models E(\theta(a_1), \theta(a_3))$

Winning Condition

- Duplicator wins a run of the game if the mapping between elements of the two structures defined by the game run is a partial isomorphism.
- Otherwise, Spoiler wins.
- A player has a winning strategy for k moves if s/he can win the k-move game no matter how the other player plays.
- Winning strategies can be fully described by finite game trees.
- There is always either a winning strategy for Spoiler or for Duplicator.
- Notation $\mathcal{A} \sim_k \mathcal{B}$: There is a winning strategy for Duplicator for k-move games.
- Notation $\mathcal{A} \nsim_k \mathcal{B}$: There is a winning strategy for Spoiler for k-move games.

Game tree of depth 2

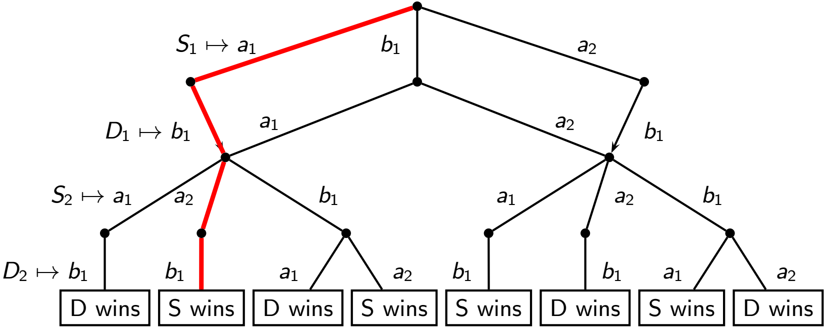


(Here, subtrees are used multiple times to save space – the game tree really is a tree, not a DAG.)

Game tree of depth 2; Spoiler has a winning strategy

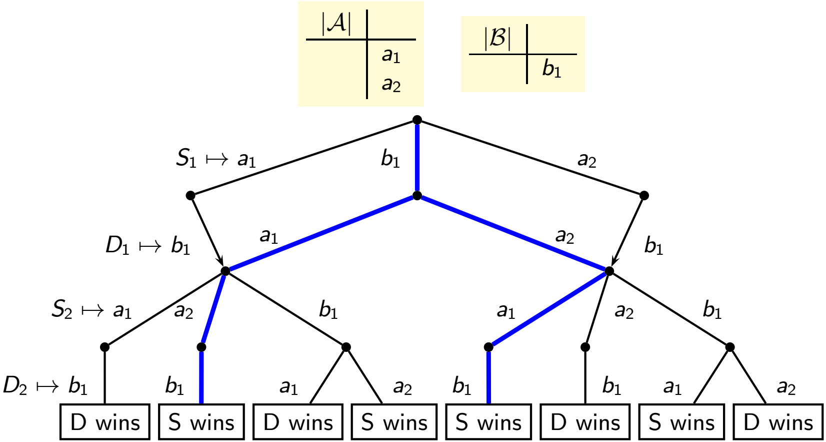
$\lvert\mathcal{A}\rvert$	
	a_1
	a_2

$\lvert\mathcal{B}\rvert$	
	b_1



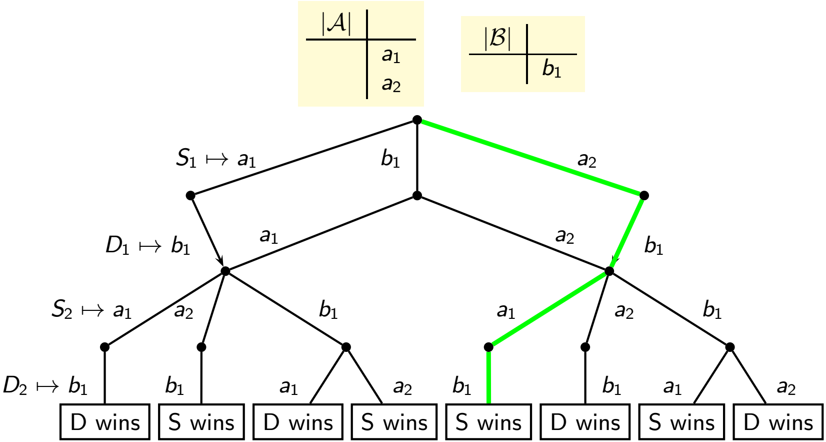
1st winning strategy for Spoiler in two moves ($\mathcal{A} \not\sim_2 \mathcal{B}$)

Game tree of depth 2; Spoiler has a winning strategy



2nd winning strategy for Spoiler in two moves ($\mathcal{A} \not\sim_2 \mathcal{B}$)

Game tree of depth 2; Spoiler has a winning strategy



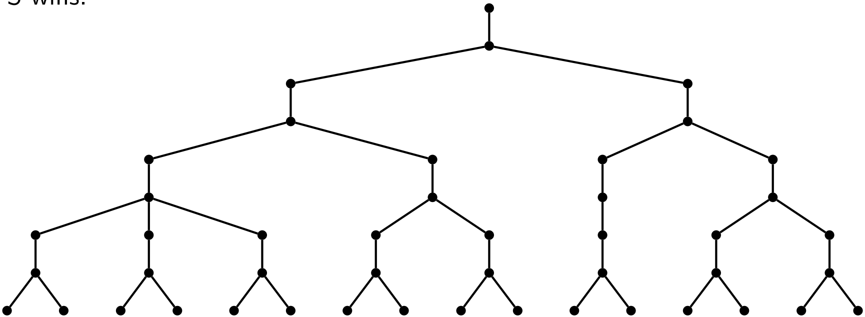
3rd winning strategy for Spoiler in two moves ($\mathcal{A} \not\sim_2 \mathcal{B}$)

Schema of a winning strategy for Spoiler

There is a possible move for S such that
for all possible answer moves of D
there is a possible move for S such that
for all possible answer moves of D

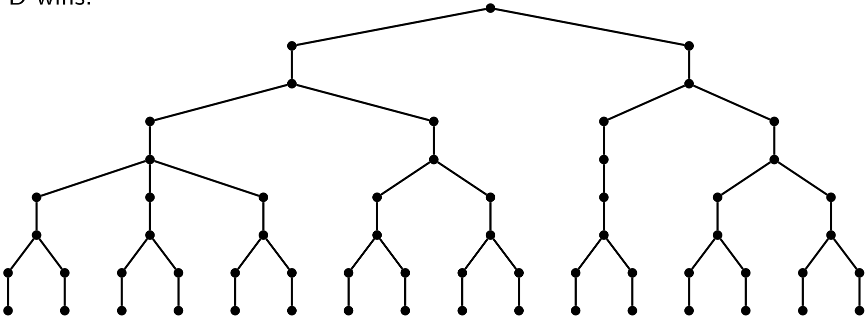
⋮

S wins.

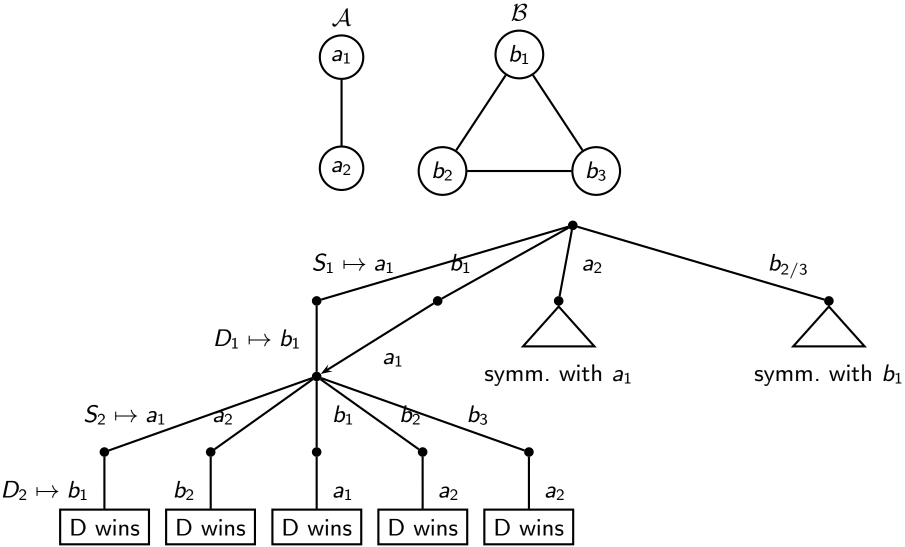


Schema of a winning strategy for Duplicator

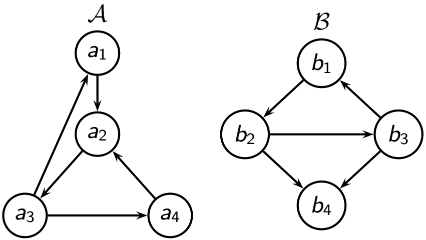
For all possible moves of S
there is a possible answer move for D such that
for all possible moves of S
there is a possible answer move for D such that
⋮
D wins.



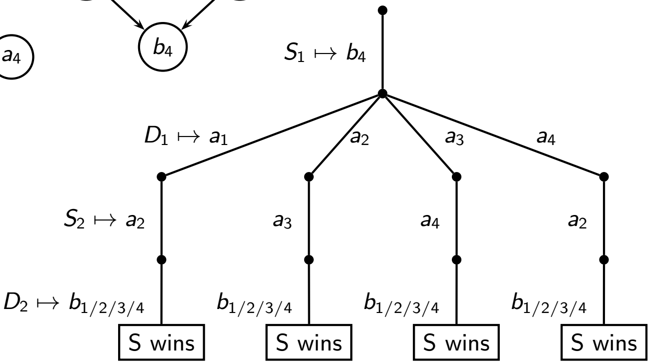
Example 1: $\mathcal{A} \sim_2 \mathcal{B}$ – Duplicator has a winning strategy



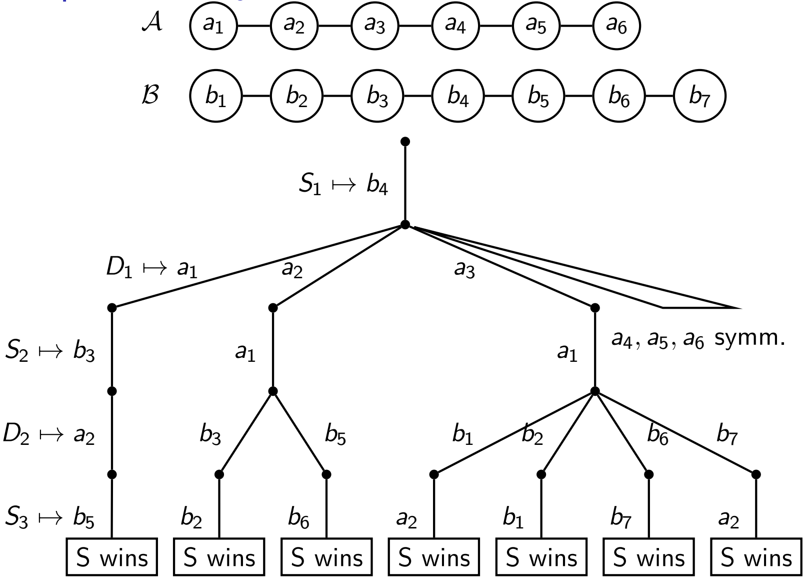
Example 2: $\mathcal{A} \not\approx_2 \mathcal{B}$ – Spoiler has a winning strategy



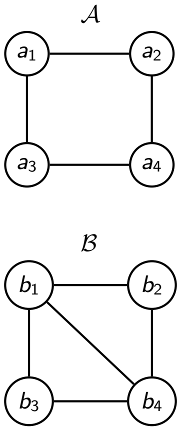
$\mathcal{B} \models \exists x_1 \forall x_2\ \neg E(x_1, x_2)$
$\mathcal{A} \not\models \exists x_1 \forall x_2\ \neg E(x_1, x_2)$

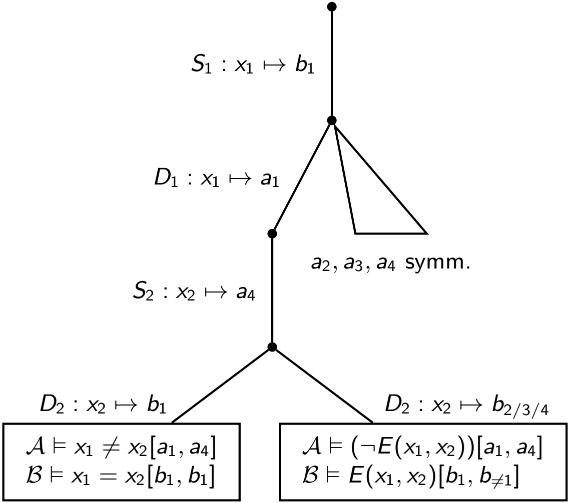


Example 3: $\mathcal{A} \not\approx_3 \mathcal{B}$

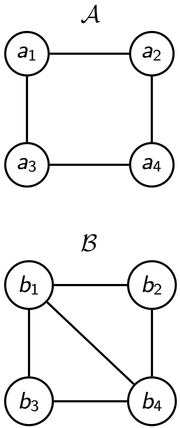


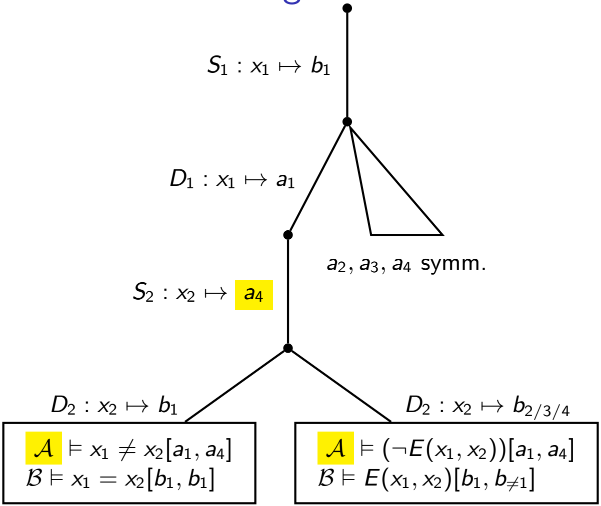
Example 4: $\mathcal{A} \not\sim_2 \mathcal{B}$



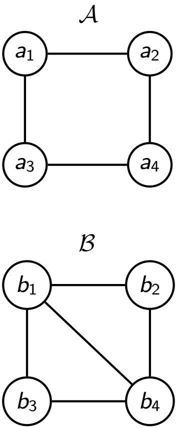


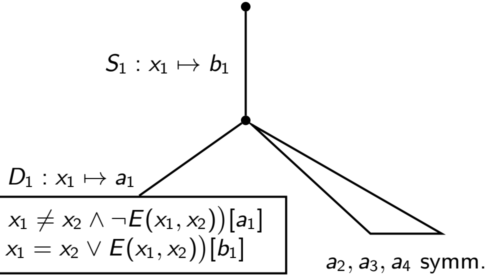
Example 4: an FO sentence to distinguish $\mathcal{A}$ and $\mathcal{B}$





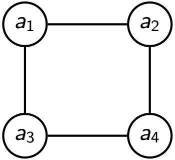
If $x_1 \mapsto a_1$ in $\mathcal{A}$ and $x_1 \mapsto b_1$ in $\mathcal{B}$ then there exists an x_2 (that is, a_4) in $\mathcal{A}$ such that $x_1 \neq x_2$ and $\neg E(x_1, x_2)$. In $\mathcal{B}$ this is not the case.

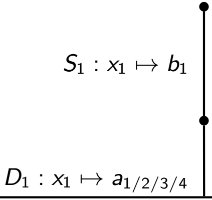




$$\boxed{\begin{array}{l} \mathcal{A} \models \big(\exists x_2\ x_1 \neq x_2 \wedge \neg E(x_1, x_2)\big)[a_1] \\ \mathcal{B} \models \big(\forall x_2\ x_1 = x_2 \vee E(x_1, x_2)\big)[b_1] \end{array}}$$

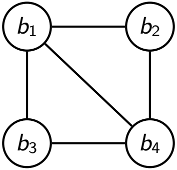
$\mathcal{A}$

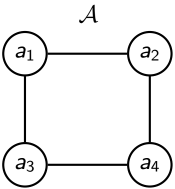


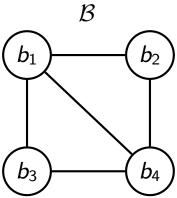


$$\boxed{\begin{array}{l} \mathcal{A} \models \big(\exists x_2\ x_1 \neq x_2 \wedge \neg E(x_1, x_2)\big)[a_{1/2/3/4}] \\ \mathcal{B} \models \big(\forall x_2\ x_1 = x_2 \vee E(x_1, x_2)\big)[b_1] \end{array}}$$

$\mathcal{B}$





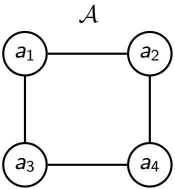


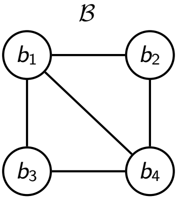
$S_1 : x_1 \mapsto \boxed{b_1}$

$D_1 : x_1 \mapsto a_{1/2/3/4}$

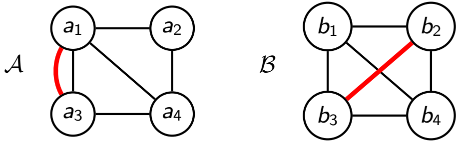
$$\mathcal{A} \models \big(\exists x_2\ x_1 \neq x_2 \wedge \neg E(x_1, x_2)\big)[a_{1/2/3/4}]$$
$$\boxed{\mathcal{B}} \models \big(\forall x_2\ x_1 = x_2 \vee E(x_1, x_2)\big)[b_1]$$

$$\boxed{\begin{array}{l}\mathcal{B} \models \exists x_1 \forall x_2\ x_1 = x_2 \lor E(x_1, x_2) \\ \mathcal{A} \models \forall x_1 \exists x_2\ x_1 \neq x_2 \land \neg E(x_1, x_2)\end{array}}$$



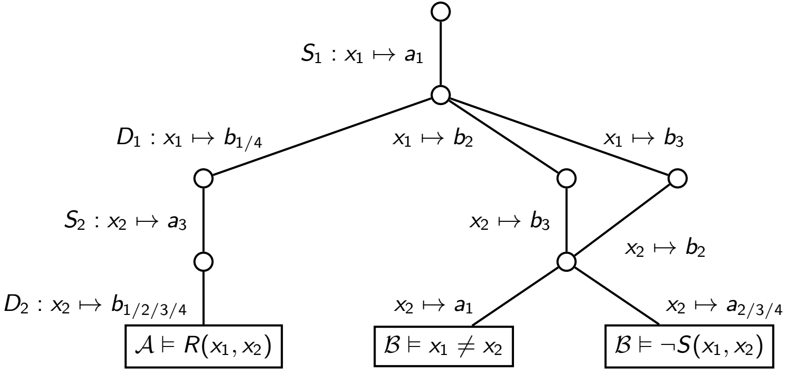


Example 5: an FO sentence to distinguish $\mathcal{A}$ and $\mathcal{B}$

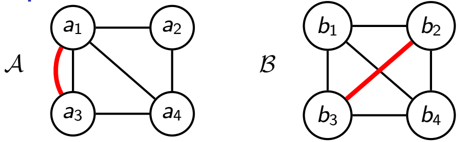


two symmetric binary relations R (red) and S (black).

$\mathcal{A} \not\sim_2 \mathcal{B}$

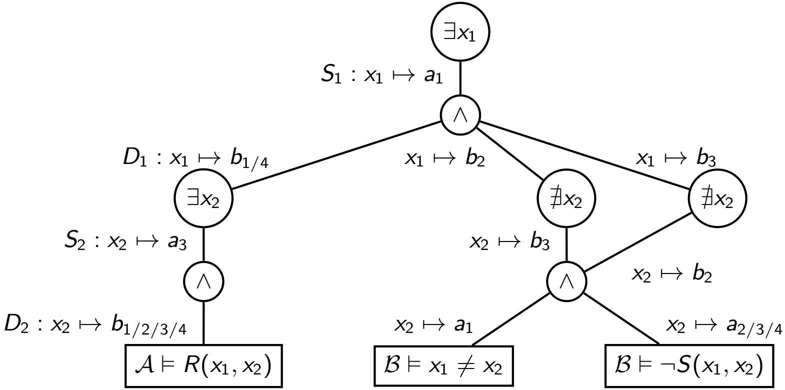


Example 5: an FO sentence to distinguish $\mathcal{A}$ and $\mathcal{B}$



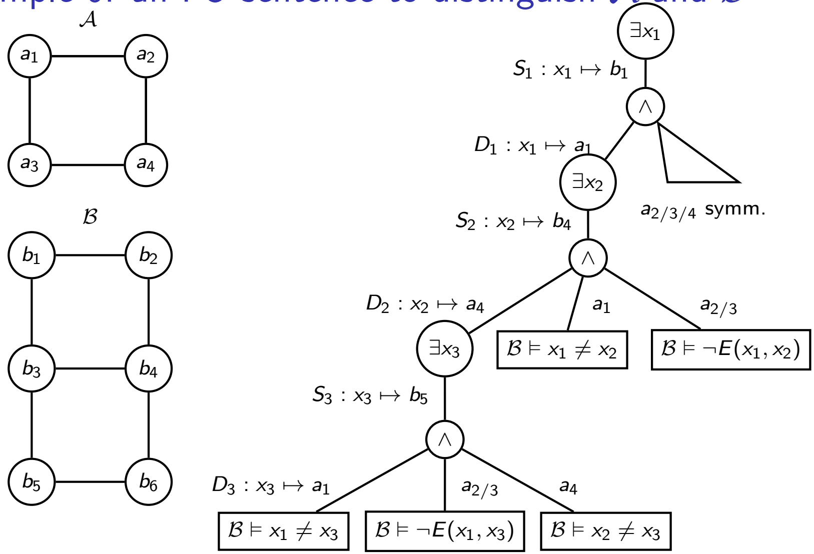
two symmetric binary relations R (red) and S (black).

$\mathcal{A} \not\sim_2 \mathcal{B}$



$\phi = \exists x_1(\exists x_2\ R(x_1, x_2)) \wedge \nexists x_2\ x_1 \neq x_2 \wedge \neg S(x_1, x_2)$; $\mathcal{A} \models \phi, \mathcal{B} \not\models \phi$.

Example 6: an FO sentence to distinguish $\mathcal{A}$ and $\mathcal{B}$



$\phi = \exists x_1 \exists x_2 \, (\exists x_3 \; x_1 \neq x_3 \wedge \neg E(x_1, x_3) \wedge x_2 \neq x_3) \wedge x_1 \neq x_2 \wedge \neg E(x_1, x_2)$

$\mathcal{B} \models \phi$, $\mathcal{A} \not\models \phi$.

An FO sentence that distinguishes between $\mathcal{A}$ and $\mathcal{B}$

- Input: a winning strategy for Spoiler.
- We construct a sentence ϕ which is true on the structure on which Spoiler puts the first token (this structure is initially the "current structure") and is false on the other structure.
- Spoiler's choice of structure in move i decides the i-th quantifier:
  - $\exists x_i$ if $i = 1$ or if Spoiler chooses the same structure that she has chosen in move $i - 1$ and
  - $\neg\exists x_i$ if Spoiler does not choose the same structure as in the previous move. We switch the current structure.
- The alternative answers of Duplicator are combined using conjunctions.
- Each leaf of the strategy tree corresponds to a literal (=a possibly negated atomic formula) that is true on the current structure and false on the other structure. Such a literal exists because Spoiler wins on the leaf, i.e., a mapping is forced that is not a partial isomorphism.

Main theorem

Definition

We write $\mathcal{A} \equiv_k \mathcal{B}$ for two structures $\mathcal{A}$ and $\mathcal{B}$ if and only if the following is true for all FO sentences ϕ of quantifier rank k:

$$\mathcal{A} \models \phi \quad \Leftrightarrow \quad \mathcal{B} \models \phi.$$

Theorem (Ehrenfeucht, Fraïssé)

Given two structures $\mathcal{A}$ and $\mathcal{B}$ and an integer k. Then the following statements are equivalent:

1. $\mathcal{A} \equiv_k \mathcal{B}$, *i.e., $\mathcal{A}$ and $\mathcal{B}$ cannot be distinguished by FO sentences of quantifier rank k.*
2. $\mathcal{A} \sim_k \mathcal{B}$, *i.e., Duplicator has a winning strategy for the k-move EF game.*

Proof of the theorem of Ehrenfeucht and Fraïssé

Proof.

- We have provided a method for turning a winning strategy for Spoiler into an FO sentence that distinguishes $\mathcal{A}$ and $\mathcal{B}$.
- From this it follows immediately that

$$\mathcal{A} \not\sim_k \mathcal{B} \;\Rightarrow\; \mathcal{A} \not\equiv_k \mathcal{B}$$

  and thus

$$\mathcal{A} \equiv_k \mathcal{B} \;\Rightarrow\; \mathcal{A} \sim_k \mathcal{B}.$$

- We still have to prove the other direction ($\mathcal{A} \not\equiv_k \mathcal{B} \;\Rightarrow\; \mathcal{A} \not\sim_k \mathcal{B}$).
- Proof idea: we can construct a winning strategy for Spoiler for the k-move EF game from a formula ϕ of quantifier rank k with $\mathcal{A} \models \phi$ and $\mathcal{B} \models \neg\phi$.

□

Proof of the theorem of Ehrenfeucht and Fraïssé

Lemma (quantifier-free case)

Given a formula ϕ with $qr(\phi) = 0$ and $free(\phi) = \{x_1, \ldots, x_l\}$. If $\mathcal{A} \models \phi[a_{i_1}, \ldots, a_{i_l}]$ and $\mathcal{B} \models (\neg\phi)[b_{j_1}, \ldots, b_{j_l}]$ then

$$\{a_{i_1} \mapsto b_{j_1}, \ldots, a_{i_l} \mapsto b_{j_l}\}$$

is not a partial isomorphism.

Proof.

W.l.o.g., only atomic formulae may occur in negated form.
By structural induction:

- If ϕ is an atomic formula, then the lemma holds.
- If $\phi = \psi_1 \wedge \psi_2$ then $\neg\phi = (\neg\psi_1) \vee (\neg\psi_2)$; the lemma holds again.
- If $\phi = \psi_1 \vee \psi_2$ then $\neg\phi = (\neg\psi_1) \wedge (\neg\psi_2)$; as above. $\square$

Proof of the theorem of Ehrenfeucht and Fraïssé

Lemma

Given a formula ϕ with $k = qr(\phi)$ and $free(\phi) = \{x_1, \dots, x_l\}$ for $l \geq 0$. If $\mathcal{A} \vDash \phi[a_{i_1}, \dots, a_{i_l}]$ and $\mathcal{B} \vDash (\neg\phi)[b_{j_1}, \dots, b_{j_l}]$ then Spoiler can win each game run over $k + l$ moves which starts with $a_{i_1} \mapsto b_{j_1}, \dots, a_{i_l} \mapsto b_{j_l}$.

Proof

By induction on k:

- $\mathrm{qr}(\phi) = 0$: see the lemma of the previous slide.
- $\phi = \exists x_{l+1}\ \psi$: There exists an element $a_{i_{l+1}}$ such that $\mathcal{A} \vDash \psi[a_{i_1}, \dots, a_{i_{l+1}}]$ but for all $b_{j_{l+1}}$, $\mathcal{B} \vDash (\neg\psi)[b_{j_1}, \dots, b_{j_{l+1}}]$. If the induction hypothesis holds for ψ then it also holds for ϕ.
- $\phi = \forall x_{l+1}\ \psi$: This is analogous to the previous case if one considers $\neg\phi = \exists x_{l+1}\ \psi'$ with $\psi' = \neg\psi$ on $\mathcal{B}$.
- $\phi = (\psi_1 \wedge \psi_2)$ and $\phi = (\psi_1 \vee \psi_2)$ work analogously. $\square$

Proof of the theorem of Ehrenfeucht and Fraïssé

From

Lemma

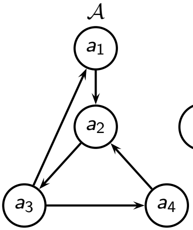
Given a formula ϕ *with* $free(\phi) = \{x_1, \ldots, x_l\}$*. If* $\mathcal{A} \models \phi[a_{i_1}, \ldots, a_{i_l}]$ *and* $\mathcal{B} \models (\neg\phi)[b_{j_1}, \ldots, b_{j_l}]$ *then Spoiler can win each game run over* $qr(\phi) + l$ *moves which starts with* $a_{i_1} \mapsto b_{j_1}, \ldots, a_{i_l} \mapsto b_{j_l}$*.*

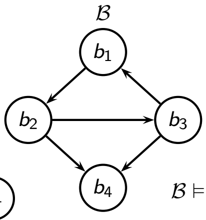
it immediately follows in the case $l = 0$ that

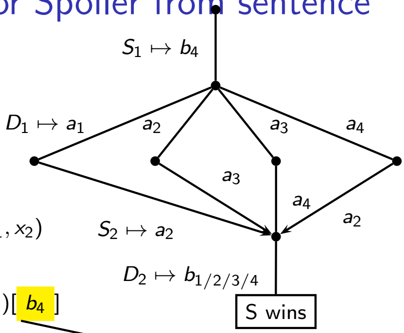
Lemma

If $\mathcal{A} \not\equiv_k \mathcal{B}$ *then* $\mathcal{A} \nsim_k \mathcal{B}$*.*

Construction: Winning strategy for Spoiler from sentence







$\mathcal{B} \models \exists x_1 \forall x_2\ \neg E(x_1, x_2)$

$\mathcal{B} \models (\forall x_2\ \neg E(x_1, x_2))[\,b_4\,]$

$\mathcal{B} \models (\neg E(x_1, x_2))[b_4, b_1]$ $\mathcal{B} \models (\neg E(x_1, x_2))[b_4, b_2]$ $\mathcal{B} \models (\neg E(x_1, x_2))[b_4, b_3]$ $\mathcal{B} \models (\neg E(x_1, x_2))[b_4, b_4]$

$\mathcal{A} \models \forall x_1 \exists x_2\ E(x_1, x_2)$

$\mathcal{A} \models (\exists x_2\ E(x_1, x_2))[a_1]$ $\mathcal{A} \models (\exists x_2\ E(x_1, x_2))[a_2]$ $\mathcal{A} \models (\exists x_2\ E(x_1, x_2))[a_3]$ $\mathcal{A} \models (\exists x_2\ E(x_1, x_2))[a_4]$

$\mathcal{A} \models E(x_1, x_2)[\,a_1, a_2\,]$ $\mathcal{A} \models E(x_1, x_2)[\,a_2, a_3\,]$ $\mathcal{A} \models E(x_1, x_2)[\,a_3, a_4\,]$ $\mathcal{A} \models E(x_1, x_2)[\,a_4, a_2\,]$

Inexpressibility proofs

- Expressibility of a query in FO means that there is an FO formula equivalent to that query;
- if there is such a formula, it must have some quantifier rank.
- We thus get the following methodology for proving inexpressibility:

Theorem (Methodology theorem)

Given a Boolean query Q. There is **no** *FO sentence that expresses Q if and only if there are, for each k, structures $\mathcal{A}_k$, $\mathcal{B}_k$ such that*

- $\mathcal{A}_k \models Q$,
- $\mathcal{B}_k \not\models Q$ *and*
- $\mathcal{A}_k \sim_k \mathcal{B}_k$.

Thus, EF games provide a complete methodology for constructing inexpressibility proofs. To prove inexpressibility, we only have to

- construct suitable structures $\mathcal{A}_k$ and $\mathcal{B}_k$ and
- prove that $\mathcal{A}_k \sim_k \mathcal{B}_k$. (This is usually the difficult part.)

Example: Inexpressibility of the parity query

Definition (parity query)

Given a structure $\mathcal{A}$ with empty schema (i.e., only $|\mathcal{A}|$ is given).
Question: Does $|\mathcal{A}|$ have an even number of elements?

- Construction of the structures $\mathcal{A}_n$ and $\mathcal{B}_n$ for arbitrary n:

$$|\mathcal{A}_n| := \{a_1, \ldots, a_n\} \qquad |\mathcal{B}_n| := \{b_1, \ldots, b_{n+1}\}$$

Lemma

$\mathcal{A}_n \sim_k \mathcal{B}_n$ for all $k \leq n$.

(This is shown on the next slide.)

- On the other hand, $\mathcal{A}_n \vDash$ Parity if and only if $\mathcal{B}_n \nvDash$ Parity.
- It thus follows from the methodology theorem that parity is not expressible in FO.

Example: Inexpressibility of the parity query

Lemma

$\mathcal{A}_n \sim_k \mathcal{B}_n$ for all $k \leq n$.

Proof.

We construct a winning strategy for Duplicator. This time no strategy trees are explicitly shown, but a general construction is given.
We handle the case in which Spoiler plays on $\mathcal{A}_n$. The other direction is analogous. If $S_i \mapsto a$ then

- $D_i \mapsto b$ where b is a new element of $|\mathcal{B}_n|$ if a has not been played on yet (=no token was put on it);
- If, for some $j < i$, $S_j \mapsto a, D_j \mapsto b'$ or $S_j \mapsto b', D_j \mapsto a$ was played then $D_i \mapsto b'$.

Over k moves, we only construct partial isomorphisms in this way and obtain a winning strategy for Duplicator. □

Undirected Paths

Theorem

Let L_1, L_2 be undirected paths of length $\geq 2^k$. Then $L_1 \sim_k L_2$ holds.

Proof idea.

- Consider the nodes in L_1 and L_2 arranged from left to right, s.t. we have a linear order on the nodes.
- Add nodes "min" on the left and "max" on the right of each path.
- For every $i \in \{0, \ldots, k\}$, consider the i-round EF-game and assume that before the actual game, the additional nodes "min" and "max" are played in the two graphs.
- Hence, after i moves, the players have chosen vectors $\vec{a} = (a_{-1}, a_0, a_1, \ldots, a_i)$ in L_1 and $\vec{b} = (b_{-1}, b_0, b_1, \ldots, b_i)$ in L_2 with $a_{-1} = b_{-1} =$ "min" and $a_0 = b_0 =$ "max".
- As usual, we define the distance $d(u, v)$ between two nodes u and v as the length of the shortest path between u and v.

Proof continued.

A winning strategy for the Duplicator can be obtained as follows:
The Duplicator can play in such a way that for every $j, l \in \{-1, \ldots, i\}$, the following conditions hold:

1. if $d(a_j, a_l) < 2^{k-i}$, then $d(a_j, a_l) = d(b_j, b_l)$;
2. if $d(a_j, a_l) \geq 2^{k-i}$, then $d(b_j, b_l) \geq 2^{k-i}$;
3. $a_j \leq a_l$ if and only if $b_j \leq b_l$

The claim is proved by induction on i:
$i = 0$. Clear. In particular, we have $d(a_{-1}, a_0) \geq 2^{k-0}$ and $d(b_{-1}, b_0) \geq 2^{k-0}$.

$i \to i + 1$. Suppose the spoiler makes the $(i + 1)$st move in L_1.
(the case of L_2 is symmetric.)
Case 1. $a_{i+1} = a_j$ for some j. Then the Duplicator chooses $b_{i+1} = b_j$.
Case 2. a_{i+1} is in the interval a_j and a_l for some j, l.

Proof continued.

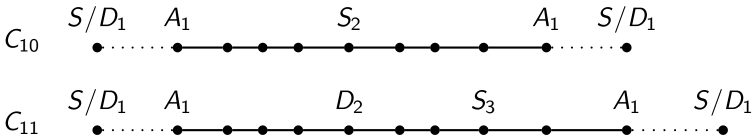
Case 2.1. a_{i+1} is "close to" a_j, i.e., $d(a_j, a_{i+1}) < 2^{k-i-1}$.
Then the Duplicator chooses b_{i+1} in the interval b_j and b_l with $d(b_j, b_{i+1}) = d(a_j, a_{i+1})$.

Case 2.2. a_{i+1} is "close to" a_l, i.e., $d(a_{i+1}, a_l) < 2^{k-i-1}$.
Then the Duplicator chooses b_{i+1} in the interval b_j and b_l with $d(b_{i+1}, b_l) = d(a_{i+1}, a_l)$.

Case 2.3. a_{i+1} is "far away from" both a_j and a_l, i.e., $d(a_j, a_{i+1}) \geq 2^{k-i-1}$ and $d(a_{i+1}, a_l) \geq 2^{k-i-1}$.
Then the Duplicator chooses b_{i+1} in the middle between b_j and b_l. $\square$

Cycles

- (Isolated) undirected cycles C_n: Graphs with nodes $\{v_1, \ldots, v_n\}$ and edges $\{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)\}$.
- After the first move, there is one distinguished node in the cycle, the one with token S_1 or D_1 on it.
- We can treat this cycle like a path obtained by cutting the cycle at the distinguished node.



- Theorem. If $n \geq 2^k$, then $C_n \sim_k C_{n+1}$.

2-colorability

Definition

2-colorability: Given a graph, is there a function that maps each node to either "red" or "green" such that no two adjacent nodes have the same color?

Theorem

2-colorability is not expressible in FO.

Proof Sketch.

For each k,

- $\mathcal{A}_k$: C_{2^k}, the cycle of length 2^k.
- $\mathcal{B}_k$: C_{2^k+1}, the cycle of length $2^k + 1$.
- $\mathcal{A}_k \sim_k \mathcal{B}_k$.
- However, a cycle C_n of length n is 2-colorable iff n is even.

Inexpressibility follows from the EF methodology theorem. $\square$

Acyclicity

From now on, "very long/large" means simply 2^k.

Theorem

Acyclicity is not expressible in FO.

Proof Sketch.

- $\mathcal{A}_k$: a very long path.
- $\mathcal{B}_k$: a very long path plus (disconnected from it) a very large cycle.
- $\mathcal{A}_k \sim_k \mathcal{B}_k$.

□

Graph reachability

Theorem

Graph reachability from a to b is not expressible in FO.

a, b are constants or are given by an additional unary relation with two entries.

Proof Sketch.

- $\mathcal{A}_k$: a very large cycle in which the nodes a and b are maximally distant.
- $\mathcal{B}_k$: two very large cycles; a is a node of the first cycle and b a node of the second.
- $\mathcal{A}_k \sim_k \mathcal{B}_k$.

□

Remark. The same structures $\mathcal{A}_k$, $\mathcal{B}_k$ can be used to show that connectedness of a graph is not expressible in FO.

Further Examples

Theorem

The following Boolean queries are not expressible in FO:

- *Hamiltonicity (does the graph have a Hamilton cycle);*
- *Eulerian Graph (does the graph have a Eulerian cycle, i.e., a round trip that visits each edge of the graph exactly once);*
- *k-Colorability for arbitrary $k \geq 2$;*
- *Existence of a clique of size $\geq n/2$ (with $n =$ number of vertices).*

Learning Objectives

- Rules of EF game
- Winning condition and winning strategies of EF games
- EF Theorem and its proof
- Inexpressibility proofs using the Methodology theorem

Literature

- Phokion Kolaitis, "Combinatorial Games in Finite Model Theory": http://www.cse.ucsc.edu/∼kolaitis/talks/essllif.ps (Slides 1–40)
- Abiteboul, Hull, Vianu, "Foundations of Databases", Addison-Wesley 1994. Chapter 17.2.
- Libkin, "Elements of Finite Model Theory", Springer 2004. Chapter 3.
- Ebbinghaus, Flum, "Finite Model Theory", Springer 1999. Chapter 2.1–2.3.