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 $\Phi$ be an infinite set of FO sentences and suppose that every finite subset of $\Phi$ is satisfiable. Then also $\Phi$ 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 $\psi$, s.t. for every structure $\mathcal{G}$ representing a graph, the following equivalence holds:

Graph $\mathcal{G}$ is connected iff $\mathcal{G} \models \psi$. 
Proof.
Assume to the contrary that there exists an FO-formula $\psi$ 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
   
   \[ \phi_n := \neg \exists x_1 \ldots \exists x_n \ x_1 = c_1 \land x_n = c_2 \land \bigwedge_{1 \leq i \leq n-1} E(x_i, x_{i+1}). \]
   
   (“There does not exist a path of length $n - 1$ between $c_1$ and $c_2$.”)

2. Clearly, $\Phi$ is unsatisfiable.

3. Consider an arbitrary, finite subset $\Phi_0$ of $\Phi$. There exists $n_{\text{max}}$, s.t. $\phi_m \notin \Phi_0$ for all $m > n_{\text{max}}$.

4. $\Phi_0$ is satisfiable: a single path of length $n_{\text{max}} + 1$ satisfies $\Phi_0$. Hence, also every finite subset $\Phi_0 \subset \Phi$ is satisfiable.

5. By the Compactness Theorem, $\Phi$ is satisfiable, which contradicts the observation (2) above. Hence, $\psi$ 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 $\Phi$ of FO sentences with the following properties:

- every finite subset of $\Phi$ has a finite model and
- $\Phi$ 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., $d_n \iff \text{there exist at least } n \text{ pairwise distinct elements.}$

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

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

- $A, B$ denote structures (=databases),
- $|A|$ is the domain of a structure $A$,
- $E^A$ is the relation $E$ of a structure $A$. 
A game run with $k = 3$

A
\[
\begin{array}{cccc}
 a_1 & a_2 & a_3 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_1 & a_2 \\
 a_4 & a_3 & a_2 & a_1 \\
\end{array}
\]

$E^A$
\[
\begin{array}{c|c|c|c}
 a_1 & a_2 & a_3 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_2 & a_1 \\
 a_4 & a_3 & a_1 & a_2 \\
\end{array}
\]

$|A|$
\[
\begin{array}{c|c|c|c}
 a_1 & a_2 & a_3 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_2 & a_1 \\
 a_4 & a_3 & a_1 & a_2 \\
\end{array}
\]

$E^B$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

$|B|$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

A game run with $k = 3$

B
\[
\begin{array}{cccc}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

$E^B$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

$|B|$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

A game run with $k = 3$

A
\[
\begin{array}{cccc}
 a_1 & a_2 & S_1 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_2 & a_1 \\
 a_4 & a_3 & a_1 & a_2 \\
\end{array}
\]

$E^A$
\[
\begin{array}{c|c|c|c}
 a_1 & a_2 & S_1 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_2 & a_1 \\
 a_4 & a_3 & a_1 & a_2 \\
\end{array}
\]

$|A|$
\[
\begin{array}{c|c|c|c}
 a_1 & a_2 & S_1 & a_4 \\
 a_2 & a_1 & a_3 & a_4 \\
 a_3 & a_4 & a_2 & a_1 \\
 a_4 & a_3 & a_1 & a_2 \\
\end{array}
\]

$E^B$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]

$|B|$
\[
\begin{array}{c|c|c|c}
 b_1 & b_2 & b_3 & b_4 \\
 b_2 & b_1 & b_3 & b_4 \\
 b_3 & b_4 & b_1 & b_2 \\
 b_4 & b_3 & b_2 & b_1 \\
\end{array}
\]
A game run with $k = 3$

Partial isomorphisms

**Definition**

- $\mathcal{A}|_S$: Restriction of a structure $\mathcal{A}$ to the subdomain $S \subseteq |\mathcal{A}|$. Same schema; for each relation $R^A$:
  \[ R^{\mathcal{A}|_S} := \{ (a_1, \ldots, a_k) \in R^A \mid a_1, \ldots, a_k \in S \}. \]

- A partial function $\theta : |\mathcal{A}| \to |\mathcal{B}|$ is a partial isomorphism from $\mathcal{A}$ to $\mathcal{B}$ if and only if $\theta$ is an isomorphism from $\mathcal{A}|_{\text{dom}(\theta)}$ to $\mathcal{B}|_{\text{rng}(\theta)}$.

- This definition assumes that the schema of $\mathcal{A}$ does not contain any constants but is purely relational.
Partial isomorphisms

Example

<table>
<thead>
<tr>
<th>RA</th>
<th></th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4</td>
<td>2 1 4</td>
<td>1 2 3 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RB</th>
<th></th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b c</td>
<td>a b d</td>
<td>a b c</td>
</tr>
</tbody>
</table>

θ: { 
1 ↦→ a
2 ↦→ b
3 ↦→ c
}

<table>
<thead>
<tr>
<th>RA</th>
<th></th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3</td>
<td>1 2 3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RB</th>
<th></th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>a b c</td>
<td>a b c</td>
<td></td>
</tr>
</tbody>
</table>

θ is a partial isomorphism.

The partial function θ : |A| → |B| with

θ : { 
1 ↦→ b
2 ↦→ a
3 ↦→ c
}

is not a partial isomorphism: A ⊨ a2 ≠ a4, B ⊭ θ(a2) ≠ θ(a4).

The partial function θ : |A| → |B| with

θ : { 
1 ↦→ b
2 ↦→ a
3 ↦→ c
4 ↦→ b
}

is a partial isomorphism.

The partial function θ : |A| → |B| with

θ : { 
1 ↦→ b
a2 ↦→ b1
a3 ↦→ b3
a4 ↦→ b1
}

is not a partial isomorphism: A ⊨ E(a1, a3), B ⊭ E(θ(a1), θ(a3)).
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 \( A \sim_k B \): There is a winning strategy for Duplicator for \( k \)-move games.

Notation \( A \not\sim_k B \): There is a winning strategy for Spoiler for \( k \)-move games.

Game tree of depth 2; Spoiler has a winning strategy

1st winning strategy for Spoiler in two moves (\( A \sim_2 B \))

2nd winning strategy for Spoiler in two moves (\( A \not\sim_2 B \))
Game tree of depth 2; Spoiler has a winning strategy

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:

D wins

A \sim 2 B

3rd winning strategy for Spoiler in two moves (A \sim 2 B)

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 2: $A \sim_2 B$ – Spoiler has a winning strategy

Example 3: $A \sim_3 B$

Example 4: $A \sim_2 B$

Example 4: an FO sentence to distinguish $A$ and $B$
7.3. Examples

A

$D_1 : x_1 \mapsto a_1$

$A \models (\exists x_2 \ x_1 \neq x_2 \land \neg E(x_1, x_2))[a_1]$

$B \models (\forall x_2 \ x_1 = x_2 \lor E(x_1, x_2))[b_1]$

$a_2, a_3, a_4 \text{ symm.}$

B

$S_1 : x_1 \mapsto b_1$

$A \models (\exists x_2 \ x_1 \neq x_2 \land \neg E(x_1, x_2))[a_1/2/3/4]$

$B \models (\forall x_2 \ x_1 = x_2 \lor E(x_1, x_2))[b_1]$

$B \models (\exists x_1 \forall x_2 x_1 = x_2 \lor E(x_1, x_2))$

$A \models (\forall x_1 \exists x_2 x_1 \neq x_2 \land \neg E(x_1, x_2))$
Example 5: an FO sentence to distinguish $A$ and $B$

$$\phi = \exists x_1 (\exists x_2 R(x_1, x_2)) \land \neg \exists x_2 x_1 \neq x_2 \land \neg S(x_1, x_2); \ A \models \phi, B \not\models \phi.$$
7.4. EF Theorem

Proof of the theorem of Ehrenfeucht and Fraïssé

- We have provided a method for turning a winning strategy for Spoiler into an FO sentence that distinguishes \( A \) and \( B \).
- From this it follows immediately that 
  \[ A \preceq_k B \Rightarrow A \not\equiv_k B \]
  and thus 
  \[ A \equiv_k B \Rightarrow A \sim_k B. \]
- We still have to prove the other direction (\( A \not\equiv_k B \Rightarrow A \sim_k B \)).
- Proof idea: we can construct a winning strategy for Spoiler for the \( k \)-move EF game from a formula \( \phi \) of quantifier rank \( k \) with \( A \models \phi \) and \( B \models \neg \phi \).

Proof

Given a formula \( \phi \) with \( \text{free}(\phi) = \{x_1, \ldots, x_l\} \). If \( A \models \phi[a_1, \ldots, a_k] \) and \( B \models \neg(\phi)[b_1, \ldots, b_k] \) then Spoiler can win each game run over \( qr(\phi) + l \) moves which starts with \( a_i \mapsto b_j, \ldots, a_k \mapsto b_k \).

Lemma

Given a formula \( \phi \) with \( \text{free}(\phi) = \{x_1, \ldots, x_l\} \). If \( A \models \phi[a_1, \ldots, a_i] \) and \( B \models \neg(\phi)[b_1, \ldots, b_j] \) then Spoiler can win each game run over \( qr(\phi) + 1 \) moves which starts with \( a_i \mapsto b_j, \ldots, a_i \mapsto b_j \).

Proof

By induction:
- \( qr(\phi) = 0 \): see the lemma of the previous slide.
- \( \phi = \exists x_{l+1} \psi \): There exists an element \( a_{i+1} \) such that 
  \( A \models \psi[a_1, \ldots, a_{i+1}] \) but for all \( b_{i+1}, B \models \neg(\psi)[b_1, \ldots, b_{i+1}] \). If the induction hypothesis holds for \( \psi \) then it also holds for \( \phi \).
- \( \phi = \forall x_{l+1} \psi \): This is analogous to the previous case if one considers 
  \( \neg\phi = \exists x_{l+1} \psi' \) with \( \psi' = \neg\psi \) on \( B \).
- \( \phi = (\psi_1 \land \psi_2) \) and \( \phi = (\psi_1 \lor \psi_2) \) work analogously.
Proof of the theorem of Ehrenfeucht and Fraïssé

From

Lemma

Given a formula \( \phi \) with \( \text{free}(\phi) = \{x_1, \ldots, x_t\} \). If \( A \models \phi[a_0, \ldots, a_t] \) and \( B \models (\neg \phi)[b_0, \ldots, b_t] \) then Spoiler can win each game run over \( \phi \) + \( l \) moves which starts with \( a_i \mapsto b_i \), \( i = 0, \ldots, t \).

it immediately follows in the case \( l = 0 \) that

Lemma

If \( A \not\equiv_k B \) then \( A \not\sim_k B \).

Inexpressibility proofs

- Expressibility of a query in \( \text{FO} \) means that there is an \( \text{FO} \) 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 \( \text{FO} \) sentence that expresses \( Q \) if and only if there are, for each \( k \), structures \( A_k, B_k \) such that

- \( A_k \models Q \),
- \( B_k \not\models Q \) and
- \( A_k \sim_k B_k \).

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

- construct suitable structures \( A_k \) and \( B_k \) and
- prove that \( A_k \sim_k B_k \). (This is usually the difficult part.)

Example: Inexpressibility of the parity query

Definition (parity query)

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

- Construction of the structures \( A_n \) and \( B_n \) for arbitrary \( n \):
  \[ |A_n| := \{a_1, \ldots, a_n\} \quad |B_n| := \{b_1, \ldots, b_{n+1}\} \]

Lemma

\( A_n \sim_k B_n \) for all \( k \leq n \).

(This is shown on the next slide.)

- On the other hand, \( A_n \models \text{Parity} \) if and only if \( B_n \not\models \text{Parity} \).
- It thus follows from the methodology theorem that parity is not expressible in \( \text{FO} \).
Example: Inexpressibility of the parity query

Lemma

\[ A_n \sim_k B_n \text{ for all } k \leq n. \]

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 \( A_n \). The other direction is analogous. If \( S_i \rightarrow a \) then

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

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

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

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 \( d(b_{-1}, b_0) \geq 2^{k-0} \).
- \( i \rightarrow i + 1 \). Suppose the spoiler makes the \((i+1)\)st move in \( L_1 \). (the case of \( L_2 \) is symmetric.)
  - Case 1. \( a_{i+1} = a_i \) 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 \( j, l \).

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 \( \bar{a} = (a_{-1}, a_0, a_1, \ldots, a_i) \) in \( L_1 \) and \( \bar{b} = (b_{-1}, b_0, b_1, \ldots, b_i) \) 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.

Case 2.1. \( a_{i+1} \) is “close to” \( a_j \), i.e., \( d(a_j, a_{i+1}) < 2^{k-i} \).

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” \( a_i \), i.e., \( d(a_{i+1}, a_i) < 2^{k-i} \).

Then the Duplicator chooses \( b_{i+1} \) in the interval \( b_j \) and \( b_l \) with \( d(b_{i+1}, b_i) = d(a_{i+1}, a_i) \).

Case 2.3. \( a_{i+1} \) is “far away from” both \( a_j \) and \( a_i \), i.e., \( d(a_j, a_{i+1}) \geq 2^{k-i} \) and \( d(a_{i+1}, a_i) \geq 2^{k-i} \).

Then the Duplicator chooses \( b_{i+1} \) in the middle between \( b_j \) and \( b_l \).
Cycles

- (Isolated) directed 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.

$$C_{10} \quad S/D_1 \quad A_1 \quad S_2 \quad A_1 \quad S/D_1$$
$$C_{11} \quad S/D_1 \quad A_1 \quad D_2 \quad S_3 \quad A_1 \quad S/D_1$$

Theorem. If $n \geq 2k$, 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 $k$,
- $A_k$: $C_{2^k}$, the cycle of length $2^k$.
- $B_k$: $C_{2^k+1}$, the cycle of length $2^k + 1$.
- $A_k \sim_k B_k$.

However, a cycle $C_n$ of length $n$ is 2-colorable iff $n$ is even. Inexpressibility follows from the EF methodology theorem.

Acyclicity

From now on, “very long/large” means simply $2^k$.

**Theorem**

Acyclicity is not expressible in FO.

**Proof Sketch.**

- $A_k$: a very long path.
- $B_k$: a very long path plus (disconnected from it) a very large cycle.
- $A_k \sim_k B_k$.

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.**

- $A_k$: a very large cycle in which the nodes $a$ and $b$ are maximally distant.
- $B_k$: two very large cycles; $a$ is a node of the first cycle and $b$ a node of the second.
- $A_k \sim_k B_k$.

Remark. The same structures $A_k$, $B_k$ 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)