The Cai-Fürer-Immerman construction

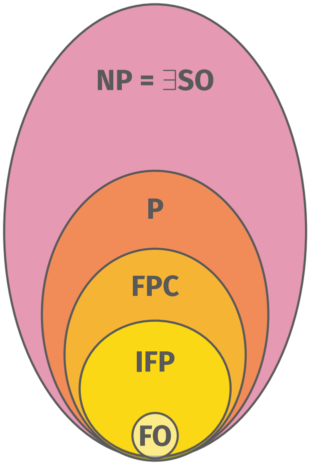
Benedikt Pago [1]

ESSLLI 2025, Bochum

[1]University of Cambridge

UNIVERSITY OF CAMBRIDGE
Department of Computer Science and Technology

Reminder



≈ Weisfeiler-Leman algorithm for Graph Isomorphism

≈ k-consistency algorithm for CSP

Infinitary Counting Logic

Just as IFP can be seen as a fragment of $\mathcal{L}^{\omega}_{\infty\omega}$, FPC is a fragment of $\mathcal{C}^{\omega}_{\infty\omega}$.

For every $k \in \mathbb{N}$, $\mathcal{C}^{k}_{\infty\omega}$ is the extension of $\mathcal{L}^{k}_{\infty\omega}$ with counting quantifiers $\exists^{\geq m}x$ for all $m \in \mathbb{N}$.

Just as IFP can be seen as a fragment of $\mathcal{L}^{\omega}_{\infty\omega}$, FPC is a fragment of $\mathcal{C}^{\omega}_{\infty\omega}$.

For every $k \in \mathbb{N}$, $\mathcal{C}^{k}_{\infty\omega}$ is the extension of $\mathcal{L}^{k}_{\infty\omega}$ with counting quantifiers $\exists^{\geq m}x$ for all $m \in \mathbb{N}$.

Theorem (Grädel and Otto, 1993)

For every sentence $\psi \in \text{FPC}$, there exists a $k \in \mathbb{N}$ and a $\varphi \in \mathcal{C}^{k}_{\infty\omega}$ such that ψ and φ are equivalent on all finite structures.

Separating FPC from PTIME

Goal: Construct a family of pairs of graphs $(G_n, H_n)_{n\in\mathbb{N}}$ such that

- For every $k \in \mathbb{N}$, for large enough $n \in \mathbb{N}$, it holds $G_n \equiv_{\mathcal{C}^k} H_n$.
- For all $n \in \mathbb{N}$, $G_n \not\cong H_n$.
- There is a PTIME-algorithm that distinguishes all G_n and H_n.

Goal: Construct a family of pairs of graphs $(G_n, H_n)_{n\in\mathbb{N}}$ such that

- For every $k \in \mathbb{N}$, for large enough $n \in \mathbb{N}$, it holds $G_n \equiv_{\mathcal{C}^k} H_n$.
- For all $n \in \mathbb{N}$, $G_n \not\cong H_n$.
- There is a PTIME-algorithm that distinguishes all G_n and H_n.

Consequences:

- There is no fixed k such that $\mathcal{C}^k$**-equivalence** is the same as isomorphism.
- There is no fixed k such that the k**-dimensional Weisfeiler-Leman** algorithm decides isomorphism.
- For each $k \in \mathbb{N}$, $\mathcal{C}^k \neq$ PTIME.
- $\implies$ Since every FPC-sentence is equivalent to a $\mathcal{C}^k$-sentence for a fixed k, FPC $\neq$ PTIME.

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f\colon A \to B$.

The bijective k-pebble game for $\mathcal{C}^k_{\infty\omega}$

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f : A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.

The bijective k-pebble game for $\mathcal{C}^k_{\infty\omega}$

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f \colon A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

The bijective k-pebble game for $\mathcal{C}^k_{\infty\omega}$

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f : A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f : A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

Duplicator wins if the play continues forever without Spoiler winning.

The bijective k-pebble game for $\mathcal{C}^k_{\infty\omega}$

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

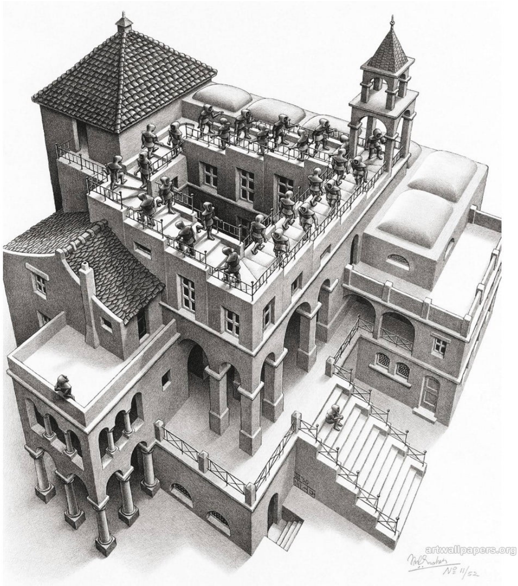
- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f : A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

Duplicator wins if the play continues forever without Spoiler winning.

Theorem (Hella, 1996)

Duplicator has a winning strategy in the k-pebble game on $(\mathfrak{A}, \mathfrak{B})$ if and only if $\mathfrak{A} \equiv_{\mathcal{C}^k} \mathfrak{B}$.

Motto: Construct locally consistent globally inconsistent instances.



$$0 + 1 + \boxed{1 + 0 + 1} + 0 = 0 \bmod 2?$$

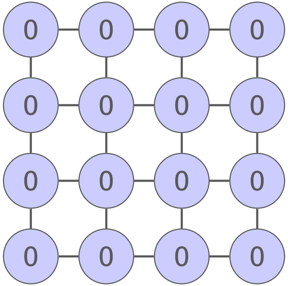
$$\boxed{0 + 1 + 1} + 0 + 1 + 0 = 0 \bmod 2?$$

Plan

Starting point: A **CSP instance** that is hard for *k-consistency*.

Second step: Lifting to graph isomorphism instances hard for *k-Weisfeiler-Leman*.

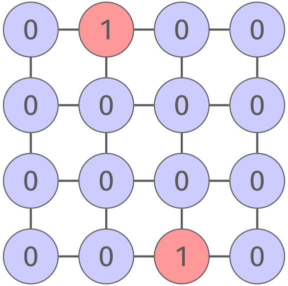
Tseitin systems



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

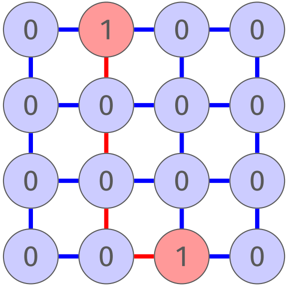


Given a graph $G = (V, E)$ with node labels $\lambda\colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Tseitin systems

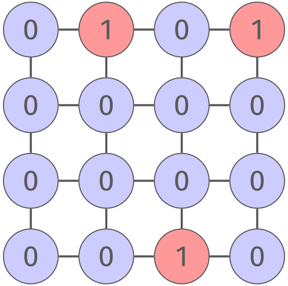


Given a graph $G = (V, E)$ with node labels $\lambda\colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Tseitin systems

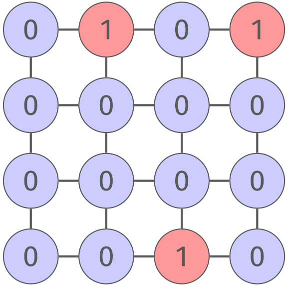


Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Tseitin systems



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

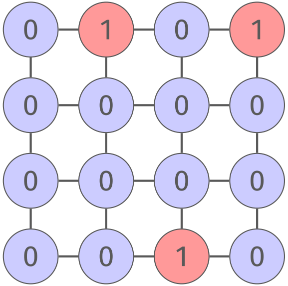
- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma

$\mathcal{T}(G, \lambda)$ is satisfiable over $\mathbb{Z}_2$ if and only if $\sum_{v \in V} \lambda(v) = 0 \mod 2$.

Tseitin systems



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

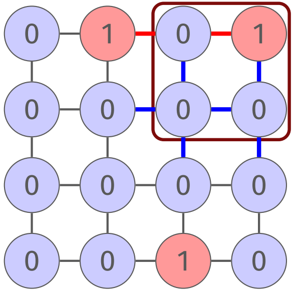
- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.

Tseitin systems



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

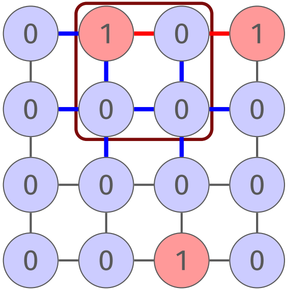
- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.

Tseitin systems



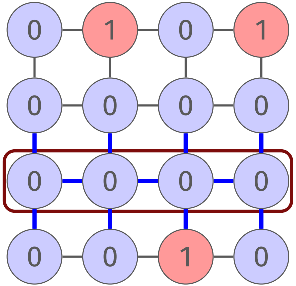
Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.



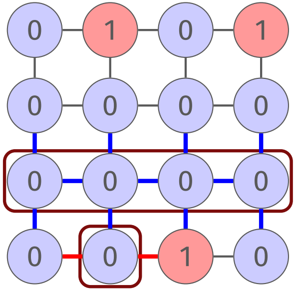
Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.



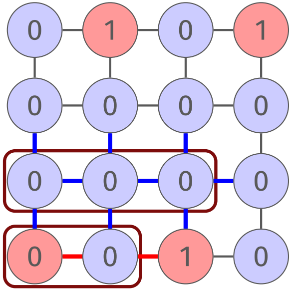
Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.



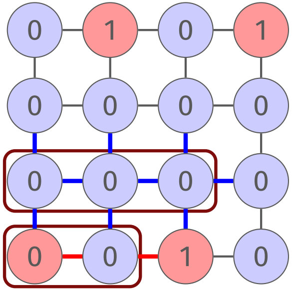
Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

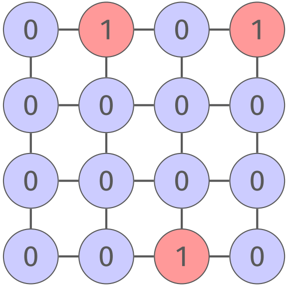
$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

Lemma (Atserias, Bulatov, Dalmau, 2007)

If k is smaller than the dimensions of the grid, the k-consistency algorithm does not detect unsatisfiability of $\mathcal{T}(G, \lambda)$.

Proof: Duplicator keeps the violated equation outside of the local window.

Tseitin systems



Given a graph $G = (V, E)$ with node labels $\lambda \colon V \to \mathbb{Z}_2$, define the *Tseitin system* $\mathcal{T}(G, \lambda)$:

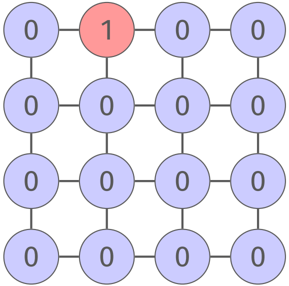
- **Variables:** For each $e \in E$, we have a variable x_e.
- **Equations:** For each $v \in V$, we have an equation

$$\sum_{e \in E(v)} x_e = \lambda(v) \mod 2.$$

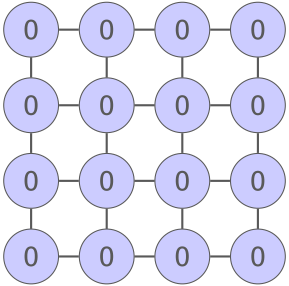
Problem: In the logic $\mathcal{C}^k$, we can decide satisfiability by expressing how many equations are odd.

Goal: Construct a family of pairs of graphs $(G_n, H_n)_{n \in \mathbb{N}}$ such that

- For every $k \in \mathbb{N}$, for large enough $n \in \mathbb{N}$, it holds $G_n \equiv_{\mathcal{C}^k} H_n$.
- For all $n \in \mathbb{N}$, $G_n \not\cong H_n$.
- There is a PTIME-algorithm that distinguishes all G_n and H_n.

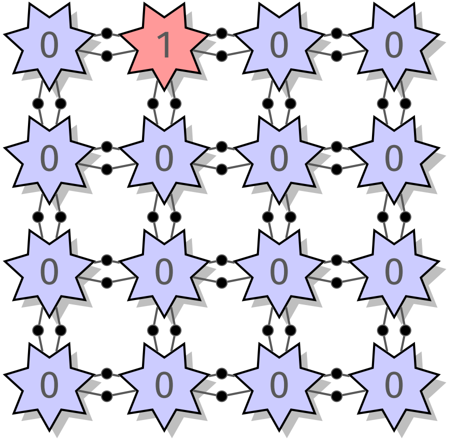


G

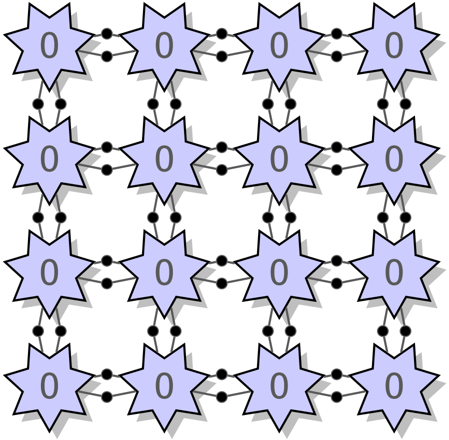


H

From Tseitin systems to Cai-Fürer-Immerman graphs

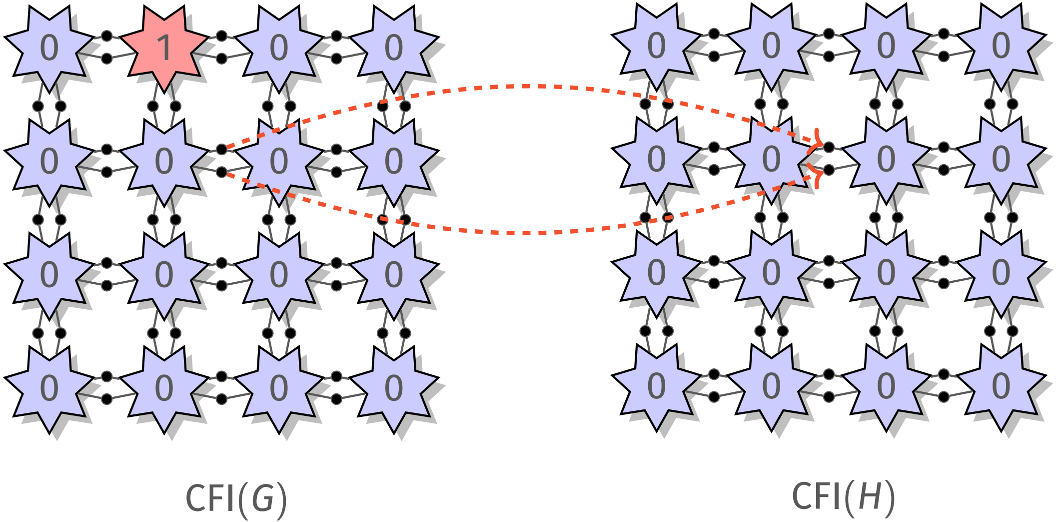


CFI(G)

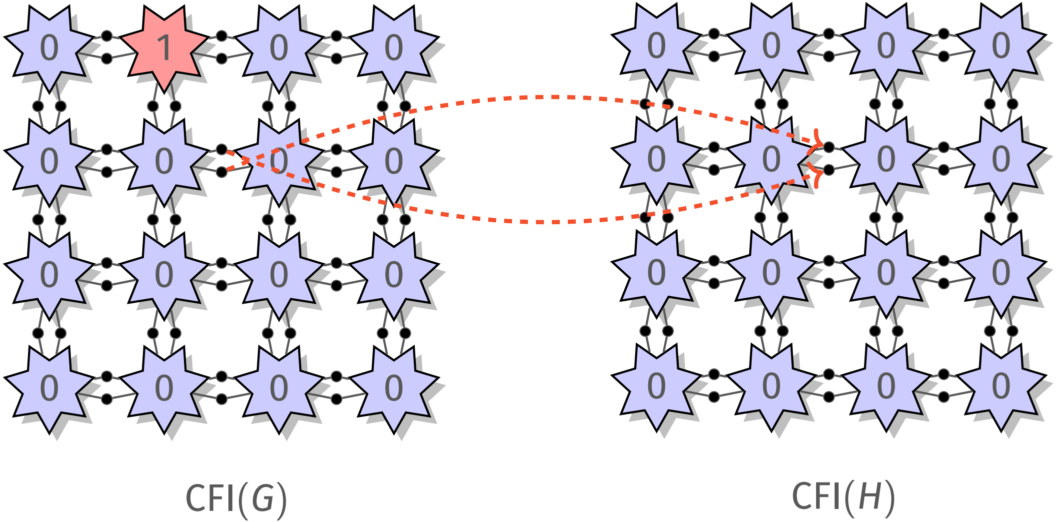


CFI(H)

From Tseitin systems to Cai-Fürer-Immerman graphs



From Tseitin systems to Cai-Fürer-Immerman graphs



Properties of CFI graphs

Lemma (Cai, Fürer, Immerman, 1992)

Let G be a connected graph, and $\lambda_0, \lambda_1 \colon V \to \mathbb{Z}_2$ two node labellings.

$$\mathrm{CFI}(G, \lambda_0) \cong \mathrm{CFI}(G, \lambda_1) \iff \sum_{v \in V} \lambda_0(v) = \sum_{v \in V} \lambda_1(v) \mod 2.$$

Lemma (Cai, Fürer, Immerman, 1992)

Let G be a connected graph, and $\lambda_0, \lambda_1 \colon V \to \mathbb{Z}_2$ two node labellings.

$$\mathrm{CFI}(G, \lambda_0) \cong \mathrm{CFI}(G, \lambda_1) \iff \sum_{v \in V} \lambda_0(v) = \sum_{v \in V} \lambda_1(v) \mod 2.$$

To show:

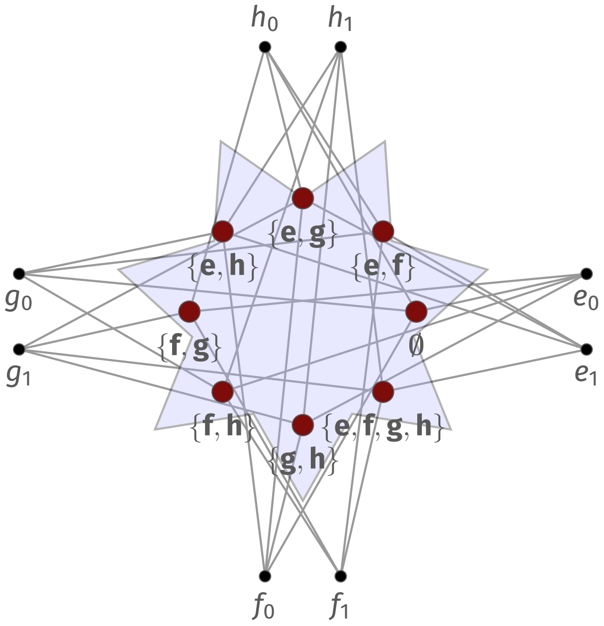
Lemma (Cai, Fürer, Immerman, 1992)

If $k \in \mathbb{N}$ is smaller than any separator of G, then

$$\mathrm{CFI}(G, \lambda_0) \equiv_{\mathcal{C}^k} \mathrm{CFI}(G, \lambda_1),$$

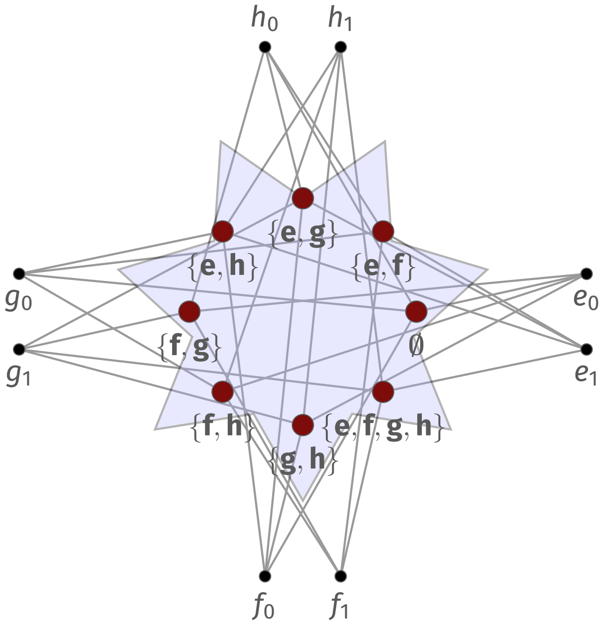
for any choice of λ_0, λ_1.

The CFI gadget



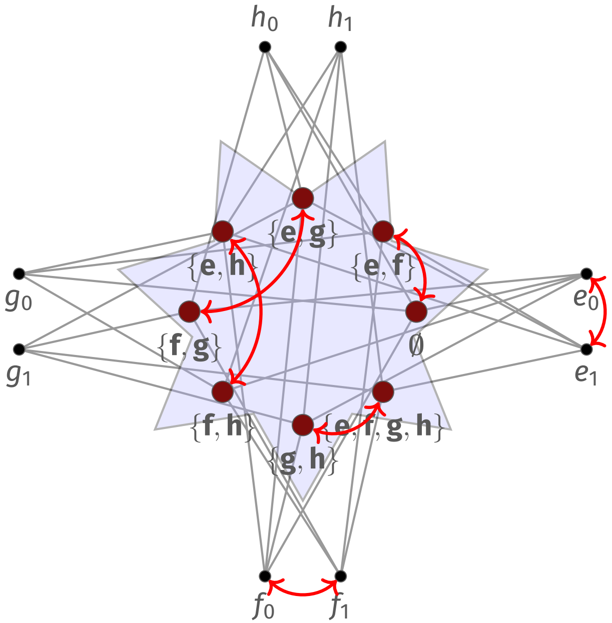
- Each inner vertex is labelled with an even set S of incident edges.

The CFI gadget



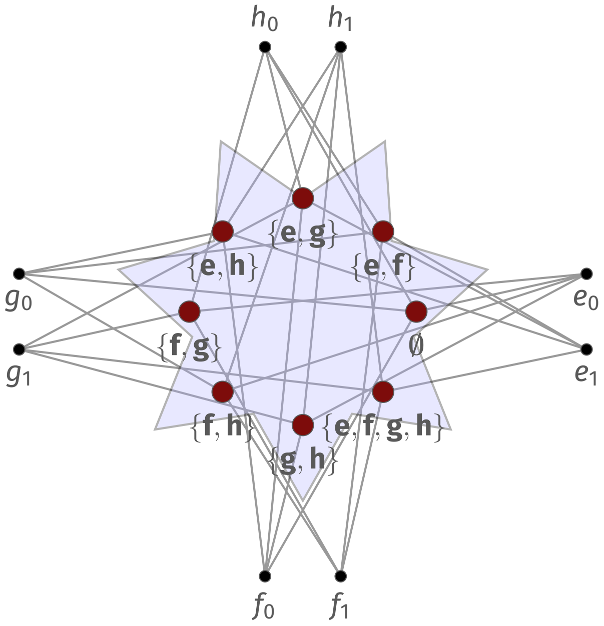
- Each inner vertex is labelled with an even set S of incident edges.
- The vertex with label S is connected with $\begin{cases} e_1 & \text{if } e \in S \\ e_0 & e \notin S \end{cases}$.

The CFI gadget



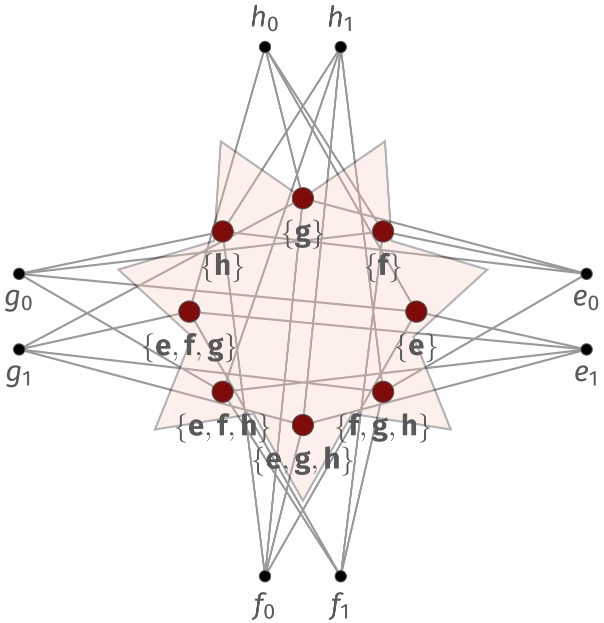
- Each inner vertex is labelled with an even set S of incident edges.
- The vertex with label S is connected with $\begin{cases} e_1 & \text{if } e \in S \\ e_0 & e \notin S \end{cases}$.
- Every "*flip*" of an even number of edges induces an automorphism of the gadget.

The CFI gadget



- Each inner vertex is labelled with an even set S of incident edges.
- The vertex with label S is connected with $\begin{cases} e_1 & \text{if } e \in S \\ e_0 & e \notin S \end{cases}$.
- Every "*flip*" of an even number of edges induces an automorphism of the gadget.
- Every flip of an odd number of edges is an isomorphism into the *odd gadget*.

The CFI gadget



- Each inner vertex is labelled with an odd set S of incident edges.
- The vertex with label S is connected with $\begin{cases} e_1 & \text{if } e \in S \\ e_0 & e \notin S \end{cases}$.
- Every "*flip*" of an even number of edges induces an automorphism of the gadget.
- Every flip of an odd number of edges is an isomorphism into the even gadget.

To show:

Lemma (Cai, Fürer, Immerman, 1992)

If $k \in \mathbb{N}$ is smaller than any separator of G, then

$$\mathrm{CFI}(G, \lambda_0) \equiv_{\mathcal{C}^k} \mathrm{CFI}(G, \lambda_1),$$

for any choice of λ_0, λ_1.

To show:

Lemma (Cai, Fürer, Immerman, 1992)

If $k \in \mathbb{N}$ is smaller than any separator of G, then

$$\mathsf{CFI}(G, \lambda_0) \equiv_{\mathcal{C}^k} \mathsf{CFI}(G, \lambda_1),$$

for any choice of λ_0, λ_1.

We have to show: Duplicator wins the bijective k-pebble game on $\mathsf{CFI}(G, \lambda_0)$ and $\mathsf{CFI}(G, \lambda_1)$.

Reminder: the bijective pebble game

Definition

Let $\mathfrak{A}, \mathfrak{B}$ two structures, $k \in \mathbb{N}$ the number of pebbles.
The *position* after any round is $(\bar{a} \in A^\ell, \bar{b} \in B^\ell)$ with $\ell \leq k$. In each round,

- Spoiler may remove a pebble-pair (a_i, b_i) that is currently on the board.
- Duplicator announces a bijection $f : A \to B$.
- Spoiler places a pebble a_i on an element of A, and b_i on $f(a_i)$.
- If $\bar{a} \to \bar{b}$ does *not* define a *local isomorphism* $\mathfrak{A}[\bar{a}] \to \mathfrak{B}[\bar{b}]$, then Spoiler wins.

Duplicator wins if the play continues forever without Spoiler winning.

A winning strategy for Duplicator

- If $\mathsf{CFI}(G, \lambda_0) \not\cong \mathsf{CFI}(G, \lambda_1)$, then there is a bijection $f\colon \mathsf{CFI}(G, \lambda_0) \to \mathsf{CFI}(G, \lambda_1)$ which is an isomorphism except at the gadget of one vertex $u \in V(G)$. Call such an f good bar u.

A winning strategy for Duplicator

- If $\mathsf{CFI}(G, \lambda_0) \not\cong \mathsf{CFI}(G, \lambda_1)$, then there is a bijection $f\colon \mathsf{CFI}(G, \lambda_0) \to \mathsf{CFI}(G, \lambda_1)$ which is an isomorphism except at the gadget of one vertex $u \in V(G)$. Call such an f good bar u.
- Duplicator maintains the **invariant** that the current bijection f is good bar u, for a vertex u whose gadget is pebble-free and in a component of G of size $\geq |V(G)|/2$.

A winning strategy for Duplicator

- If $\mathrm{CFI}(G, \lambda_0) \not\cong \mathrm{CFI}(G, \lambda_1)$, then there is a bijection $f\colon \mathrm{CFI}(G, \lambda_0) \to \mathrm{CFI}(G, \lambda_1)$ which is an isomorphism except at the gadget of one vertex $u \in V(G)$. Call such an f good bar u.
- Duplicator maintains the **invariant** that the current bijection f is good bar u, for a vertex u whose gadget is pebble-free and in a component of G of size $\geq |V(G)|/2$.
- Suppose the current bijection f satisfies the **invariant** for $u \in V(G)$, and Spoiler places a pebble on some $x \in V(\mathrm{CFI}(G, \lambda_0))$ such that $(x, y) \in E(\mathrm{CFI}(G, \lambda_0))$ but $(f(x), f(y)) \notin E(\mathrm{CFI}(G, \lambda_1))$.

A winning strategy for Duplicator

- If $\mathsf{CFI}(G, \lambda_0) \not\cong \mathsf{CFI}(G, \lambda_1)$, then there is a bijection $f \colon \mathsf{CFI}(G, \lambda_0) \to \mathsf{CFI}(G, \lambda_1)$ which is an isomorphism except at the gadget of one vertex $u \in V(G)$. Call such an f good bar u.
- Duplicator maintains the **invariant** that the current bijection f is good bar u, for a vertex u whose gadget is pebble-free and in a component of G of size $\geq |V(G)|/2$.
- Suppose the current bijection f satisfies the **invariant** for $u \in V(G)$, and Spoiler places a pebble on some $x \in V(\mathsf{CFI}(G, \lambda_0))$ such that $(x, y) \in E(\mathsf{CFI}(G, \lambda_0))$ but $(f(x), f(y)) \notin E(\mathsf{CFI}(G, \lambda_1))$.
- Duplicator chooses an "*escape path*" P from u to some pebble-free $v \in V(G)$.

A winning strategy for Duplicator

- If $\mathrm{CFI}(G, \lambda_0) \ncong \mathrm{CFI}(G, \lambda_1)$, then there is a bijection $f\colon \mathrm{CFI}(G, \lambda_0) \to \mathrm{CFI}(G, \lambda_1)$ which is an isomorphism except at the gadget of one vertex $u \in V(G)$. Call such an f good bar u.
- Duplicator maintains the **invariant** that the current bijection f is good bar u, for a vertex u whose gadget is pebble-free and in a component of G of size $\geq |V(G)|/2$.
- Suppose the current bijection f satisfies the **invariant** for $u \in V(G)$, and Spoiler places a pebble on some $x \in V(\mathrm{CFI}(G, \lambda_0))$ such that $(x, y) \in E(\mathrm{CFI}(G, \lambda_0))$ but $(f(x), f(y)) \notin E(\mathrm{CFI}(G, \lambda_1))$.
- Duplicator chooses an "*escape path*" P from u to some pebble-free $v \in V(G)$.
- In the next round, Duplicator defines a new bijection f' like f but with all edges in P *flipped*. This f' is *good bar v*.

Indistinguishability of CFI graphs

Lemma (Cai, Fürer, Immerman, 1992)

If $k \in \mathbb{N}$ is smaller than any separator of G, then

$$\mathsf{CFI}(G, \lambda_0) \equiv_{\mathcal{C}^k} \mathsf{CFI}(G, \lambda_1),$$

for any choice of λ_0, λ_1.

Indistinguishability of CFI graphs

Lemma (Cai, Fürer, Immerman, 1992)

If $k \in \mathbb{N}$ is smaller than any separator of G, then

$$\mathrm{CFI}(G, \lambda_0) \equiv_{\mathcal{C}^k} \mathrm{CFI}(G, \lambda_1),$$

for any choice of λ_0, λ_1.

Choose a family of base graphs $(G_n)_{n \in \mathbb{N}}$ such that in each G_n, any separator is large:

- The $(n \times n)$**-grid** has separator size $\Theta(\sqrt{|V(G_n)|})$.
- 3-regular **expander graphs** have separator size $\Theta(|V(G_n)|)$.

Theorem (Cai, Fürer, Immerman, 1992)

There is a family of pairs of graphs $(G_n, H_n)_{n \in \mathbb{N}}$ such that

- *For every $k \in o(|V(G_n)|)$ it holds $G_n \equiv_{\mathcal{C}^k} H_n$ for all large enough n.*
- *For all $n \in \mathbb{N}$, $G_n \not\cong H_n$.*
- *There is a PTIME-algorithm that distinguishes all G_n and H_n.*

Wrap-up

Theorem (Cai, Fürer, Immerman, 1992)

There is a family of pairs of graphs $(G_n, H_n)_{n \in \mathbb{N}}$ *such that*

- *For every* $k \in o(|V(G_n)|)$ *it holds* $G_n \equiv_{\mathcal{C}^k} H_n$ *for all large enough* n.
- *For all* $n \in \mathbb{N}$, $G_n \not\cong H_n$.
- *There is a* PTIME*-algorithm that distinguishes all* G_n *and* H_n.

Distinguishability in PTIME: Arbitrarily assign labels e_0, e_1 to the vertices in edge gadgets. Then read off how many odd vertex gadgets there are.

Best possible pebble number

Theorem (Atserias, Bulatov, Dawar, 2009)

Let G be a connected base graph and t its treewidth. Then $\mathsf{CFI}(G, \lambda) \equiv_{\mathcal{C}^k} \mathsf{CFI}(G, \lambda')$ for all $k \leq t$.

Applications of the CFI construction

Lower bounds based on the CFI construction

- CFI graphs are hard to distinguish in the polynomial calculus proof system [Berkholz, Grohe, 2015] .
- So-called "*multipedes*" [Gurevich, Shelah, 1996] are a hard example for individualisation refinement graph isomorphism algorithms [Neuen, Schweitzer, 2017].
- A disjunction construction of CFI graphs is hard for integer programming relaxations of graph isomorphism [Berkholz, Grohe, 2017] and CSPs [Lichter, P., 2025].
- A variant of the CFI construction yields graphs that have a different number of homomorphisms from a fixed graph F [Roberson, 2022].

The **polynomial calculus** allows to derive that a given set of polynomials has no common zero.

Definition (Proof rules)

Let $\mathbb{F}$ be a field, $\mathcal{V}$ the set of variables, f, g polynomials.

Linear combination: $$\frac{f \quad g}{a \cdot f + b \cdot g} \qquad a, b \in \mathbb{F}.$$

Multiplication with variable: $$\frac{f}{Xf} \qquad X \in \mathcal{V}.$$

Lower bounds for the polynomial calculus

Theorem (Berkholz, Grohe, 2015)

Any polynomial calculus proof of non-isomorphism of CFI graphs requires at least linear degree.

Multipedes

CFI graphs have many automorphisms, which explains why $\mathcal{C}^k$ cannot define them up to isomorphism.

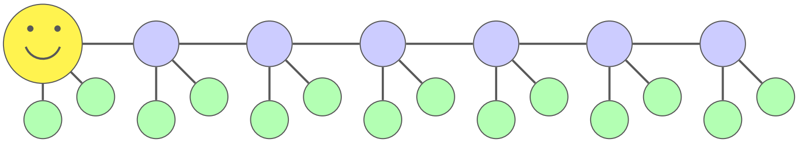
But can $\mathcal{C}^k$ define isomorphism on structures *without automorphisms*?

Multipedes

CFI graphs have many automorphisms, which explains why $\mathcal{C}^k$ cannot define them up to isomorphism.

But can $\mathcal{C}^k$ define isomorphism on structures *without automorphisms*?

No! The feet of a *multipede* are indistinguishable even though it has no automorphisms.



Multipedes

Theorem (Neuen, Schweitzer, 2017)

Graph isomorphism algorithms based on the individualisation-refinement technique require exponential running time to distinguish multipedes.

Lower bounds for integer programming algorithms

Tseitin equations and CFI graphs can be defined over any finite field, not just $\mathbb{Z}_2$.

A combination of $\mathbb{Z}_2$-and $\mathbb{Z}_3$-CFI structures yields hard instances for algorithms based on *integer linear programming*.

Theorem (Berkholz, Grohe, 2017; Lichter, P. 2025)

- *Any sublinear level of the natural integer programming relaxation of graph isomorphism fails to distinguish all graphs.*
- *There is a tractable CSP which is not solved by almost all currently studied CSP algorithms based on integer programming.*

Tseitin equations and CFI graphs can be defined over any finite field, not just $\mathbb{Z}_2$.

A combination of $\mathbb{Z}_2$-and $\mathbb{Z}_3$-CFI structures yields hard instances for algorithms based on *integer linear programming*.

Theorem (Berkholz, Grohe, 2017; Lichter, P. 2025)

- *Any sublinear level of the natural integer programming relaxation of graph isomorphism fails to distinguish all graphs.*
- *There is a tractable CSP which is not solved by almost all currently studied CSP algorithms based on integer programming.*

Open problem: To get hard examples for more CSP algorithms, a *non-Abelian* CFI construction seems to be needed.

Homomorphism indistinguishability

Two graphs G, H are called **homomorphism-indistinguishable** over a graph class $\mathcal{F}$ if every graph $F \in \mathcal{F}$ has the *same numbers of homomorphisms* into G and H.

Two graphs G, H are called **homomorphism-indistinguishable** over a graph class $\mathcal{F}$ if every graph $F \in \mathcal{F}$ has the *same numbers of homomorphisms* into G and H.

- Isomorphism is homomorphism-indistinguishability over all graphs [Lovász, 1967].
- $\mathcal{C}^k$-equivalence is homomorphism-indistinguishability over all graphs of treewidth $\leq k$ [Dvořák, 2010].
- Cospectrality is homomorphism-indistinguishability over all cycles.
- ...

Roberson showed how to use the CFI construction to generate, given G, two graphs G_0, G_1 such that

$$\mathrm{hom}(G, G_0) \neq \mathrm{hom}(G, G_1).$$

Homomorphism indistinguishability

Roberson showed how to use the CFI construction to generate, given G, two graphs G_0, G_1 such that

$$\hom(G, G_0) \neq \hom(G, G_1).$$

This idea has numerous applications, such as:

Theorem (Roberson, 2022)

Homomorphism indistinguishability over graphs of bounded degree is not isomorphism.

Theorem (Lichter, P., Seppelt, 2024)

Equivalence in linear-algebraic logic is not captured by any homomorphism indistinguishability relation.

[1] Albert Atserias, Andrei Bulatov, and Anuj Dawar. **"Affine systems of equations and counting infinitary logic".** In: Theoretical Computer Science 410.18 (2009), pp. 1666–1683.

[2] Albert Atserias, Andrei A. Bulatov, and Víctor Dalmau. **"On the Power of k-Consistency".** In: Automata, Languages and Programming, 34th International Colloquium, ICALP 2007, Wroclaw, Poland, July 9-13, 2007, Proceedings. Ed. by Lars Arge et al. 4596. Lecture Notes in Computer Science. Springer, 2007, pp. 279–290. 10.1007/978-3-540-73420-8_26.

[3] Christoph Berkholz and Martin Grohe. **"Limitations of algebraic approaches to graph isomorphism testing".** In: International Colloquium on Automata, Languages, and Programming. Springer. 2015, pp. 155–166.

[4] Christoph Berkholz and Martin Grohe. **"Linear Diophantine Equations, Group CSPs, and Graph Isomorphism".** In: Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19. Ed. by Philip N. Klein. SIAM, 2017, pp. 327–339. 10.1137/1.9781611974782.21.

References ii

[5] Jin-yi Cai, Martin Fürer, and Neil Immerman. **"An optimal lower bound on the number of variables for graph identification".** In: Combinatorica 12 (1992), pp. 389–410.

[6] Zdeněk Dvořák. **"On recognizing graphs by numbers of homomorphisms".** In: Journal of Graph Theory 64.4 (2010), pp. 330–342.

[7] Yuri Gurevich and Saharon Shelah. **"On finite rigid structures".** In: The Journal of Symbolic Logic 61.2 (1996), pp. 549–562.

[8] L. Hella. **"Logical Hierarchies in PTIME".** In: Information and Computation 129 (1996), pp. 1–19.

[9] M. Lichter and B. Pago. **Limitations of Affine Integer Relaxations for Solving Constraint Satisfaction Problems.** Ed. by Keren Censor-Hillel et al. Dagstuhl, Germany, 2025. 10.4230/LIPIcs.ICALP.2025.166. https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2025.166.

[10] M. Lichter, B. Pago, and T. Seppelt. **“Limitations of Game Comonads for Invertible-Map Equivalence via Homomorphism Indistinguishability”.** In: 32nd EACSL Annual Conference on Computer Science Logic (CSL 2024). Ed. by Aniello Murano and Alexandra Silva. 288. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 36:1–36:19. ISBN: 978-3-95977-310-2. 10.4230/LIPIcs.CSL.2024.36. https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CSL.2024.36.

[11] László Lovász. **“Operations with structures”.** In: Acta Mathematica Hungarica 18.3-4 (1967), pp. 321–328.

[12] Daniel Neuen and Pascal Schweitzer. **“An exponential lower bound for Individualization-Refinement algorithms for Graph Isomorphism”.** In: CoRR abs/1705.03283 (2017). arXiv: 1705.03283. http://arxiv.org/abs/1705.03283.

[13] David E. Roberson. **Oddomorphisms and homomorphism indistinguishability over graphs of bounded degree.** 2022. arXiv: 2206.10321 [math.CO]. https://arxiv.org/abs/2206.10321.