Presentation Theory IV: Verifiable Access, Audit Lower Bounds, and Residual Indistinguishability

Luca Blanchi

June 2026

Abstract

This note develops a fourth layer of Presentation Theory. The first part introduced presentation systems, observables, fibres, verification costs, and bounded-image principles. The second part developed controlled transfer packages. The third part treated normal-form compilation. Here the central object is verifiable access: an untrusted producer may output a claim and additional verification data, but a bounded checker can only trust the parts of that data that are tied back to the input through declared access.

The formal results are elementary but reusable. Verification data define accepted regions inside a positive language. Short data give few regions, yielding cover and capacity lower bounds. Finite-stage audit systems identify the exact computability content of uniform audit bounds: for a c.e. positive language presented as a union of decidable audit stages, the optimal audit-width profile has the same Turing degree as the language, and a computable majorant is equivalent to decidability. Transcripts give the operational soundness principle: if a positive input and a negative input have the same transcript with the same verification data, no sound checker can accept the positive input. Residual indistinguishability turns this into a lower-bound method: whenever every low-budget audit leaves a residual fibre containing opposite completions invisible to the allowed observables, every sound audit must pay a corresponding cost. These lower bounds pull back through transfer packages when checkers and transcripts pull back with controlled overhead.

The main conditional application is to Lovasz homomorphism separation for Cai–Furer–Immerman graphs. Lovasz guarantees that non-isomorphic finite graphs are separated by some homomorphism count. The audit question asks how much it costs to verify such a separation under bounded access. Assuming the standard residual CFI indistinguishability input with pinned boundary, CFI graphs over bounded-degree bases of linear treewidth have linear audit lower bounds: any auditable Lovasz separator checked using homomorphism counts from test graphs of treewidth at most k, together with controlled local support, guards, and terminal modules, must pay linear total audit cost. Thus low-treewidth homomorphism observables plus sublinear verified local information do not soundly audit CFI non-isomorphism. The same lower bound applies to proof-carrying canonical forms whose checker is Weisfeiler–Leman-local.

1 Purpose

Presentation Theory studies mathematics through declared access systems. A presentation describes an object. An observable extracts information from it. A normal form rigidifies it. A transfer package moves access from one context to another.

This note studies a different question:

when is an output actually auditable under bounded access?

The output may come from a proof search, a heuristic, an algorithm, a normal-form compiler, a transfer package, or an AI-assisted exploration. The checker is the trusted object. It receives

an input, a claim, and auxiliary verification data. It may query only the allowed observables, local pieces of the input, guard data, or declared terminal modules.

The guiding distinction is

$$\text{producing a claim} \quad \neq \quad \text{making the claim verifiable under the declared access.}$$

Auxiliary data do not create soundness merely by being written down. They help only insofar as the checker can compare them with the input using the access it has been given. This note gives a small calculus for measuring that comparison.

2 Bounded Audit Systems

Definition 2.1 (Audit system)**.** An audit system is a tuple

$$\mathfrak{A} = (P, L, \mathcal{A}, \mathcal{D}, \mathcal{V}, \mathcal{T}, \mu).$$

Here P is a class of inputs, $L \subseteq P$ is the positive language, $\mathcal{A}$ is the declared access model, $\mathcal{D}$ is the class of finite verification data, $\mathcal{V}$ is the class of checkers, $\mathcal{T}$ is the class of terminal modules that may be invoked, and μ is a cost scale recording data length, query cost, verification time, guard cost, and terminal-module weight.

The access model may include local queries, bounded observables, partial normal forms, bounded homomorphism counts, fragments of a proof system, or calls to declared modules. The terminal modules are part of the system, not free background knowledge. If a terminal module supplies information about a global invariant, the cost scale must declare the weight of that information.

Definition 2.2 (Total audit cost)**.** For a checker V, input p, and verification datum a, write

$$\mathrm{Cost}(V, p, a) = |a| + Q(V, p, a) + N(V, p, a) + G(V, p, a) + T(V, p, a).$$

Here $|a|$ is the size of the datum, Q is query or observable cost, N is internal verification cost, G is guard cost, and T is terminal-module weight. The exact codomain may be $\mathbb{N}$, $\mathbb{N}^r$, or another resource scale; inequalities below are interpreted in the declared scale.

Remark 2.3. Nothing in the formalism requires these five terms to be independent. They are separated because applications often pay for missing information in different places. A short datum may force many input queries. A low-query checker may rely on a strong terminal module. A normal-form package may reduce queries but increase guard cost.

Definition 2.4 (Terminal charging policy)**.** A terminal charging policy is part of the declared audit system. It assigns a weight to each terminal module call, possibly depending on the claim being decided and on the residual instance to which the module is applied. The policy is *honest for a family of lower bounds* if a module that directly decides one of the residual hard invariants is charged at least the lower-bound scale assigned to that residual invariant.

This definition is not a restriction on mathematics; it is a bookkeeping convention. If an audit model declares a unit-cost module for a global invariant, then the corresponding audit lower bound is false in that model by definition. The point of the formalism is to make such power visible in the cost ledger.

3 Accepted Regions and Audit Covers

Fix an audit system $\mathfrak{A}$. For a checker V and verification datum a, define the accepted region

$$R_{V,a} = \{p \in P : V^{\mathcal{A}}(p, a) = 1\}.$$

Definition 3.1 (Soundness and completeness)**.** The pair $(V, \mathcal{D})$ is sound for L if

$$R_{V,a} \subseteq L$$

for every $a \in \mathcal{D}$. It is complete if

$$L \subseteq \bigcup_{a \in \mathcal{D}} R_{V,a}.$$

Thus verification data give a cover of L by sound regions recognized by bounded checkers.

Definition 3.2 (Audit cover number)**.** Let $\mathcal{R}_{\mathcal{A}}$ be a chosen class of $\mathcal{A}$-verifiable sound regions. Define

$$\mathrm{Cov}_{\mathcal{A}}(L) = \min\{N : L = R_1 \cup \cdots \cup R_N,\ R_i \in \mathcal{R}_{\mathcal{A}},\ R_i \subseteq L\}.$$

Theorem 3.3 (Cover lower bound)**.** *Suppose L has a sound and complete audit system whose verification data have binary length at most s. Then*

$$\mathrm{Cov}_{\mathcal{A}}(L) \leq 2^{s+1}.$$

Consequently

$$s \geq \log_2 \mathrm{Cov}_{\mathcal{A}}(L) - 1.$$

Proof. Each datum a of length at most s defines the sound accepted region $R_{V,a} \subseteq L$. Completeness says that these regions cover L. There are fewer than 2^{s+1} binary strings of length at most s. Hence L is covered by fewer than 2^{s+1} sound regions. □

Often one lower-bounds a cover by choosing a hard finite family.

Definition 3.4 (Audit capacity)**.** For $H \subseteq L$, define

$$\mathrm{Cap}_{\mathcal{A}}(H, L) = \max_{R \in \mathcal{R}_{\mathcal{A}},\ R \subseteq L} |R \cap H|.$$

Theorem 3.5 (Capacity lower bound)**.** *If a sound audit system is complete on a finite set $H \subseteq L$ using verification data of length at most s, then*

$$s \geq \log_2 |H| - \log_2 \mathrm{Cap}_{\mathcal{A}}(H, L) - 1.$$

Proof. Every accepted region covers at most $\mathrm{Cap}_{\mathcal{A}}(H, L)$ elements of H. Fewer than 2^{s+1} data strings are available. Therefore

$$|H| \leq 2^{s+1} \mathrm{Cap}_{\mathcal{A}}(H, L),$$

which is the claimed inequality. □

4 Transcripts

The cover point of view is global. The operational point of view is local: a checker only sees its transcript.

Definition 4.1 (Transcript)**.** For a deterministic checker V, input p, and datum a, the transcript

$$\mathrm{Tr}_V(p, a)$$

records the queries made, answers received, observable values inspected, guard checks performed, terminal modules invoked, and verification steps whose results affect acceptance.

If randomized checkers are used, one fixes the random string and applies the deterministic statement to each branch. This note uses deterministic checkers.

Theorem 4.2 (Transcript soundness principle)**.** *Let $p \in L$ and $q \notin L$. If for some checker V and datum a,*

$$\mathrm{Tr}_V(p, a) = \mathrm{Tr}_V(q, a)$$

and V accepts p with datum a, then V is not sound for L.

Proof. A deterministic checker's output is a function of its transcript. If the transcripts are equal and V accepts p, it also accepts q with the same datum. Since $q \notin L$, this contradicts soundness. □

Corollary 4.3 (Unchecked information has no soundness value)**.** *If a bit of the verification datum never affects a transcript through a check against the input, an observable, a guard, or a charged terminal module, then changing that bit cannot improve soundness.*

5 Finite-Stage Audit Width

The cover and transcript principles are qualitative unless the finite stages of an audit system are declared. The effective skeleton is the same one that appears in Part I for filtered presentations.

Definition 5.1 (Finite-stage audit system)**.** Let $P \subseteq \Sigma^*$ be a decidable input universe and let $L \subseteq P$ be a positive language. A finite-stage audit system for L is an increasing sequence

$$A_0 \subseteq A_1 \subseteq A_2 \subseteq \cdots \subseteq P$$

of decidable sets such that

$$L = \bigcup_{t \geq 0} A_t.$$

The set A_t consists of inputs accepted by audits of stage at most t, after all datum length, support, observable, guard, and terminal costs have been charged into the stage parameter.

For $p \in L$, define the least audit stage

$$a_L(p) = \min\{t : p \in A_t\}.$$

The finite-window audit-width profile is

$$H_L(n) = \max\{a_L(p) : p \in L,\ |p| \leq n\},$$

with value 0 if there is no positive input of length at most n.

Theorem 5.2 (Audit-width degree theorem)**.** *For a finite-stage audit system with finite input balls,*

$$H_L \equiv_T L.$$

Consequently, L is decidable if and only if H_L has a computable majorant.

Proof. Given H_L, decide whether $p \in L$ by setting $n = |p|$ and testing the decidable predicate $p \in A_{H_L(n)}$. If $p \in L$, then $a_L(p) \leq H_L(n)$, so the test accepts. If the test accepts, then $p \in A_t$ for some t, hence $p \in L$.

Conversely, given an oracle for L, list all inputs $p \in P$ with $|p| \leq n$, keep the positive ones, and for each positive p search for the least t with $p \in A_t$. The search terminates by completeness of the audit stages. Taking the maximum computes $H_L(n)$.

If H_L has a computable majorant g, the same finite test $p \in A_{g(|p|)}$ decides L. If L is decidable, the preceding computation makes H_L computable, hence a computable majorant exists. □

Corollary 5.3 (Undecidability forces unbounded audit budgets)**.** *If L is c.e. but undecidable in a declared finite-stage audit system, then no computable function uniformly bounds the audit stage required for all positive inputs of length at most n.*

This statement does not replace residual indistinguishability lower bounds. It gives the computability obstruction behind them: whenever a bounded audit family would provide a computable stage bound for a known undecidable positive language, such a family cannot be sound and complete.

6 A Basic Parity Lower Bound

The simplest model already shows the main phenomenon.

Let $P_m = \mathbb{F}_2^m$, and let

$$L_{\mathrm{odd}} = \left\{ x \in \mathbb{F}_2^m : \sum_{i=1}^{m} x_i = 1 \right\}.$$

A checker may read coordinates of x, read arbitrary verification data, and compute without restriction.

Lemma 6.1 (Unqueried flip)**.** *Every deterministic sound and complete checker for L_{odd} must read all m coordinates on every accepting computation.*

Proof. Let $x \in L_{\mathrm{odd}}$ be accepted with datum a. Suppose the checker reads only the coordinates in $S \subsetneq \{1, \ldots, m\}$. Choose $j \notin S$, and put $y = x + e_j$. Then $y \notin L_{\mathrm{odd}}$, while $x_i = y_i$ for every queried coordinate. With the same datum a, the checker has the same transcript on x and on y. It therefore accepts y, contradicting soundness. □

The lemma does not say that parity is hard to compute. It says that an audit of a parity claim cannot leave an unchecked coordinate when the datum is untrusted.

7 Residual Indistinguishability

The parity lemma is a special case of a more flexible principle.

Definition 7.1 (Residual pair)**.** Let $p \in P$, let S be the part of the input controlled by the transcript, and let $\mathcal{O}_{\leq r}$ be the observables available at budget r. A residual pair relative to (p, S, r) is a pair $p_0, p_1 \in P$ such that:

(i) p_0 and p_1 agree on all controlled data in S;

(ii) $O(p_0) = O(p_1)$ for all $O \in \mathcal{O}_{\leq r}$ inspected by the checker;

(iii) $p_0 \in L$ and $p_1 \notin L$, or conversely.

Theorem 7.2 (Residual indistinguishability lower bound)**.** *Assume that every accepting transcript of total budget $< B$ leaves a residual pair compatible with that transcript. Then no sound checker can accept all positive inputs with total audit cost $< B$.*

Proof. Suppose $p \in L$ is accepted with datum a and cost $< B$. By hypothesis, the transcript leaves a compatible residual pair p_0, p_1 with opposite truth values for L. Choose the member of the pair that has the same truth value as p and the member with the opposite truth value. Compatibility means that the checker has the same transcript on both with datum a. By the transcript soundness principle, accepting the positive member forces acceptance of the negative member. This contradicts soundness. □

Definition 7.3 (Residual audit width)**.** Let $\mathcal{O}_{\leq k}$ be an observable family indexed by a width parameter k. For a pair p, q, define the residual audit width

$$\mathrm{RAW}(p, q)$$

to be the least value of

$$k + \text{controlled support} + \text{guard cost} + \text{terminal weight}$$

among sound audits that separate the truth values of the relevant claim on p, q.

The definition is intentionally schematic. In applications the support is a set of vertices, a set of rows, a family of sampled constraints, a bounded presentation subword, or a controlled piece of a fibre.

8 Pullback of Audit Lower Bounds

Part II of Presentation Theory developed transfer packages. Audit lower bounds transfer when checkers can be pulled back.

Definition 8.1 (Audit pullback)**.** Let $T : P \to Q$ be a map and let $L_P \subseteq P$, $L_Q \subseteq Q$ satisfy

$$p \in L_P \quad \Longleftrightarrow \quad T(p) \in L_Q.$$

An audit pullback with overhead ω assigns to each checker V_Q for L_Q a checker T^*V_Q for L_P such that every transcript and datum of cost b for V_Q pulls back to a transcript and datum of cost at most $\omega(b)$ for T^*V_Q.

Theorem 8.2 (Pullback lower bound)**.** *Suppose $T : P \to Q$ admits an audit pullback with overhead ω. If every sound audit for L_P requires cost at least $B(n)$, then every sound audit for L_Q requires cost at least any b satisfying*

$$\omega(b) < B(n).$$

In particular, for scalar costs and multiplicative overhead

$$\omega(b) \leq Cb,$$

the target cost is $\Omega(B(n)/C)$.

Proof. If L_Q had a sound audit of cost b with $\omega(b) < B(n)$, pulling it back would give a sound audit for L_P of cost $< B(n)$, contradicting the source lower bound. □

9 Auditable Lovasz Separation

For finite graphs, Lovasz's theorem says that two graphs G, H are isomorphic if and only if

$$\mathrm{Hom}(F, G) = \mathrm{Hom}(F, H)$$

for every finite graph F. Hence every non-isomorphic pair has a homomorphism-count separator.

The audit question is different. If a producer claims that F separates G and H, a checker must verify the relevant homomorphism counts under its declared access. We therefore distinguish existence of a separator from auditable separation.

Definition 9.1 (Lovasz audit package)**.** Let G, H be finite graphs. A Lovasz audit package consists of finite test graphs $F_1, \ldots, F_m$, declared values for $\mathrm{Hom}(F_i, G)$ and $\mathrm{Hom}(F_i, H)$, verification data for those values, controlled input support, guard data, and terminal modules. It is sound if the checker accepts only when at least one declared inequality

$$\mathrm{Hom}(F_i, G) \neq \mathrm{Hom}(F_i, H)$$

is correct.

Definition 9.2 (Auditable separation width)**.** For a declared audit model, define

$$\mathrm{ASW}(G, H)$$

to be the minimum of

$$\max_i \mathrm{tw}(F_i) + |S| + G_{\mathrm{guard}} + T_{\mathrm{term}} + \lceil \log_2 m \rceil$$

over all sound Lovasz audit packages separating G and H. Here S is the controlled support actually checked against the input.

Remark 9.3. The term $\lceil \log_2 m \rceil$ records the cost of selecting one test from a finite family. It can be omitted in variants where the whole family is checked in parallel and charged elsewhere.

Theorem 9.4 (Auditable separator lower bound)**.** *Let $\mathcal{O}_{\leq k}$ be the homomorphism-count observables*

$$\{\mathrm{Hom}(F, -) : \mathrm{tw}(F) \leq k\}.$$

Suppose that every support S with

$$k + |S| + G_{\mathrm{guard}} + T_{\mathrm{term}} < B$$

leaves a residual pair of graph-pair inputs with the same controlled support and the same $\mathcal{O}_{\leq k}$-values but opposite truth value for non-isomorphism. Then

$$\mathrm{ASW}(G, H) \geq B$$

for the corresponding input family.

Proof. This is the residual indistinguishability lower bound applied to the language of non-isomorphic graph pairs, with the bounded homomorphism counts as observables. □

10 CFI Inputs and the Residual Parity Mechanism

The Cai–Furer–Immerman construction turns a global parity choice on a connected base graph into a pair of non-isomorphic graphs that are hard for bounded-variable counting logic and Weisfeiler–Leman refinement.

We use the following standard structural input from the CFI theory.

Definition 10.1 (Pinned residual CFI property)**.** A family of connected base graphs B_n has the pinned residual CFI property with constant c if the following holds. Let $S \subseteq V(B_n)$ be a controlled set of base vertices, and include in S every base vertex whose CFI gadget, incident boundary relation, or pinned local twist datum is inspected by the checker. If

$$k + c < \mathrm{tw}(B_n - S),$$

then there are two CFI completions, compatible with all pinned boundary data on S, whose global CFI parities differ but which are indistinguishable by the allowed k-width CFI observables. In the homomorphism-count version, these observables are the counts from test graphs of treewidth at most k, up to the standard indexing convention relating bounded-treewidth homomorphism counts and Weisfeiler–Leman dimension.

Remark 10.2. This is the only non-formal input in the CFI application. It is the residual form of the standard CFI lower bound: bounded-width Weisfeiler–Leman and bounded-treewidth homomorphism counts, with the usual shift in indices, do not see the remaining CFI parity while the residual base still has treewidth above the width budget. The constant c absorbs the chosen convention for WL dimension, treewidth indexing, gadget size, and charged boundary data.

Remark 10.3 (What the CFI input must prove)**.** The pinned residual property is stronger than the bare inequality

$$\mathrm{tw}(B_n - S) \geq \mathrm{tw}(B_n) - |S|.$$

One must also know that the unchecked part of the CFI instance still contains a residual parity degree of freedom compatible with the pinned boundary data, and that changing that residual parity is invisible to the allowed bounded-width observables. This is the CFI-specific ingredient; the audit argument below only consumes it as an input.

Lemma 10.4 (Treewidth after controlled deletion)**.** *For every graph B and every $S \subseteq V(B)$,*

$$\mathrm{tw}(B - S) \geq \mathrm{tw}(B) - |S|.$$

Proof. Take a tree decomposition of $B - S$ of width $\mathrm{tw}(B - S)$. Add every vertex of S to every bag. This gives a tree decomposition of B of width at most $\mathrm{tw}(B - S) + |S|$. Therefore $\mathrm{tw}(B) \leq \mathrm{tw}(B - S) + |S|$, which is the claimed inequality. □

11 Linear Audit Lower Bounds for CFI

Let B_n be connected bounded-degree base graphs with

$$\mathrm{tw}(B_n) \geq \beta n$$

for some $\beta > 0$. Let

$$X_n^0 = \mathrm{CFI}(B_n, 0), \qquad X_n^1 = \mathrm{CFI}(B_n, 1)$$

denote the two parity classes in a fixed CFI convention. They are non-isomorphic.

Theorem 11.1 (Conditional auditable CFI separation lower bound)**.** *Assume the family B_n has the pinned residual CFI property, and assume the terminal charging policy is honest for residual CFI parity. Any sound Lovasz audit package separating X_n^0 from X_n^1, checked using homomorphism counts from test graphs of treewidth at most k, controlled support S, guard cost G, and terminal-module weight T, satisfies*

$$k + |S| + G + T \geq \mathrm{tw}(B_n) - O(1).$$

In particular,

$$k + |S| + G + T \geq \Omega(n).$$

Proof. Suppose, toward a contradiction, that an accepted audit has

$$k + |S| + G + T < \mathrm{tw}(B_n) - C$$

for a constant C large enough to absorb the CFI indexing and boundary constant in the pinned residual property. Charge all guard information and terminal calls that reveal base vertices or boundary parity into the effective support, and charge every global terminal call by its declared terminal weight. Honesty of the terminal charging policy means that a terminal module deciding the residual CFI parity contributes at least the residual lower-bound scale to T.

For the part of the audit not already charged as a terminal solution of the residual parity problem, the effective support still has total cost $< \mathrm{tw}(B_n) - C$. By the treewidth deletion lemma,

$$\mathrm{tw}(B_n - S) \geq \mathrm{tw}(B_n) - |S|.$$

Hence

$$k + c < \mathrm{tw}(B_n - S)$$

after increasing C if necessary. The pinned residual CFI property gives two completions compatible with all controlled and pinned boundary data and indistinguishable by the homomorphism observables of treewidth at most k, but with opposite global CFI parity.

Apply this to the graph-pair input. Keeping the controlled data and all observed homomorphism values fixed, change the residual parity on one side of the pair. One completion represents a non-isomorphic even/odd pair, while the other represents an isomorphic same-parity pair. The checker sees the same transcript with the same verification data in both cases. By the transcript soundness principle, it cannot soundly accept the non-isomorphism claim. This contradiction proves the lower bound. □

Corollary 11.2 (Linear lower bound and full-support upper bound)**.** *Under the same hypotheses,*

$$\mathrm{ASW}(X_n^0, X_n^1) = \Omega(n).$$

Moreover,

$$\mathrm{ASW}(X_n^0, X_n^1) = O(n)$$

for the Lovasz audit model in which full linear-width homomorphism access and full linear controlled support have linear cost. Thus in that full-support model $\mathrm{ASW}(X_n^0, X_n^1) = \Theta(n)$.

Proof. The theorem gives the lower bound. For the upper bound in the full-support audit model, Lovasz separation gives some finite test graph separating the two non-isomorphic graphs. Since the CFI graphs have $O(n)$ vertices for bounded-degree bases, the usual finite reconstruction argument gives a separating finite test of size $O(n)$, hence treewidth $O(n)$. Full controlled support lets the checker verify the relevant input data directly at linear cost, so the separator is auditable within $O(n)$ declared cost. □

12 Canonical Forms

Normal-form compilation studies how raw descriptions are turned into rigid data. The present audit layer asks whether a proposed normal datum can be checked economically.

Definition 12.1 (Auditable canonical-form package)**.** For a graph class $\mathcal{G}$, an auditable canonical-form package assigns to each $G \in \mathcal{G}$ a normal datum $N(G)$ and verification data a_G. A checker accepts (G, N, a_G) only when N is the declared canonical form of G. The package is sound if accepted normal data satisfy

$$G \cong H \quad \Longleftrightarrow \quad N(G) = N(H)$$

within the accepted domain.

Corollary 12.2 (No sublinear WL-local audit for CFI canonical forms)**.** *Let B_n, X_n^0, and X_n^1 be as above. Any sound canonical-form package for the CFI family whose checker is k-WL-local and whose verification uses controlled support S, guard cost G, and terminal-module weight T, satisfies*

$$k + |S| + G + T \geq \Omega(n)$$

on at least one of the two parity classes.

Proof. If both canonical forms were auditable with sublinear total cost, apply the checker to X_n^0 and X_n^1. Since the graphs are non-isomorphic, sound canonical data must differ. The two accepted audits therefore give an auditable separator for the non-isomorphism claim: verify both normal data and compare them. This contradicts the CFI separation lower bound. □

13 Relation with Proof Complexity and Learning-Style Checks

The framework overlaps with proof complexity, but it is not tied to a single proof system. A proof system becomes an audit system once one specifies the input access, allowed verification data, checker, and cost. Conversely, an audit system may use observables, local queries, normal forms, transfer packages, and terminal modules that are not naturally encoded as a standard propositional proof system.

The CFI result should therefore not be read as a new proof-complexity lower bound for every algebraic or semidefinite system. Many such lower bounds are already known in more precise proof systems. The contribution here is the access statement:

bounded WL or bounded-treewidth homomorphism access + sublinear checked local data

does not audit CFI separation unless the missing global parity information is paid somewhere else.

The same caution applies to graph-learning language. Message-passing graph neural networks are often compared with 1-WL, and higher-order variants with higher-dimensional WL. The theorem implies a conditional audit obstruction for explanations checked only through such WL-like access, but this is a corollary of the access model, not a theorem about all graph-learning architectures.

14 Connection with the Previous Parts

Part I introduced verification-length profiles for fibres. The present note refines that idea from membership data to interactive access: one asks not only how long the verification data are, but which input features the checker actually binds them to.

Part II introduced controlled transfer. Section 7 shows that audit lower bounds transfer when checkers and transcripts pull back with controlled overhead.

Part III treated normal-form compilation. Section 11 shows the audit obstruction for canonical forms: a normal form may exist, and may even be computable, while its correctness is not cheaply auditable under a restricted access model.

The common pattern is:

bounded access leaves a residual fibre;
verification data must close that fibre or pay for it.

15 Scope

The results in this note are model-relative in the same sense as the other parts of Presentation Theory. The access model declares which terminal modules are available and what they cost. Audit lower bounds then state that the missing global information must be paid through observable width, controlled support, guard data, or terminal-module weight.

The CFI theorem is also deliberately separated into a formal audit argument and a standard CFI residual input. The formal part says that residual indistinguishability gives audit lower bounds. The CFI input says that, after sublinear controlled support is removed from a linear-treewidth base, the remaining CFI parity is still invisible to sublinear WL or bounded-treewidth homomorphism observables. This separation is useful: the same audit theorem can be reused whenever another domain supplies a residual indistinguishability theorem.

References

References

[1] J.-Y. Cai, M. Furer, and N. Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica* 12 (1992), 389–410.

[2] Z. Dvorak. On recognizing graphs by numbers of homomorphisms. *Journal of Graph Theory* 64 (2010), 330–342.

[3] H. Dell, M. Grohe, and G. Rattan. Lovasz meets Weisfeiler and Leman. *ICALP 2018*, LIPIcs 107, Article 40.

[4] M. Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory.* Cambridge University Press, 2017.

[5] L. Lovasz. Operations with structures. *Acta Mathematica Academiae Scientiarum Hungaricae* 18 (1967), 321–328.

[6] L. Lovasz. *Large Networks and Graph Limits.* American Mathematical Society, 2012.

[7] D. Neuen. Homomorphism indistinguishability over graph classes. Preprint, 2023.