Machine-Checked Exact Combinatorial Certificates for Input-Oblivious Unambiguous Hierarchies, Krentel Lex-Max OGTD Collapse, and Single-Query Promise Elimination

Lluis Eriksson*

November 25, 2026

Abstract

Edward A. Hirsch (`arXiv:2611.33126`, November 24, 2026) established sharp structural collapses for input-oblivious unambiguous hierarchies built over OUP, proving that the language-oracle hierarchy satisfies OUH $\subseteq$ $\mathrm{P}^{\mathrm{TALLY}\cap\mathrm{NP}} \subseteq \mathrm{P}_{\|}^{\mathrm{ONP}}$ (Theorem 4.1, Corollary 4.2), that the loose promise-oracle hierarchy collapses into $\mathrm{OUH}^{\bullet} \subseteq \mathrm{OGTD}[\mathrm{NP}] = \mathrm{OUGTD}[\mathrm{FP}^{\mathrm{ONP}}, \text{co-ONP}, \mathrm{P}] \subseteq \mathrm{OUM}$ via Krentel lexicographically-largest transcripts (Lemma 3.1, Theorem 5.1), that the oblivious unambiguous dot-operator hierarchy collapses to $\exists^{\mathrm{ou}} \cdot \text{co-OUP} = \mathrm{OUP} \cup \text{co-OUP}$ (Theorem 6.2, Corollary 6.3), that a single loose prOUP query reduces to an OUP query ($\mathrm{OUP}^{\mathrm{prOUP}[1]} = \mathrm{OUP}^{\mathrm{OUP}[1]}$, Lemma 6.4, Theorem 6.5), and that $\mathrm{SOUP} = \mathrm{OUP} \cap \text{co-OUP}$ (Proposition 6.6). We present a standalone, zero-dependency **Lean 4.35.0** formalization (`ObliviousUnambiguousHierarchy.lean`) of 18 finite combinatorial, order-theoretic, and oracle-consistency theorems (0 `sorry`, 0 warnings, 0 `Classical.choice`, 11 zero-axiom theorems), paired with a pure-stdlib zero-float finite verifier (`OUH-CERT-1.0`) auditing four formally specified modes across 16 canonical certificates (0.038–0.789 ms) and rejecting 60/60 live adversarial forgeries.

1 Introduction

Input-oblivious complexity classes [3, 1] restrict nondeterministic witnesses to depend only on the input length n rather than the individual instance $x \in \{0,1\}^n$. Combining obliviousness with unambiguity yields OUP, where all positive instances of length n share a single unique witness w_n. Building on the loose unambiguous hierarchy of Hirsch and Sokolov [5], Hirsch [4] proved that the resulting hierarchies OUH (language oracles) and $\mathrm{OUH}^{\bullet}$ (loose promise oracles) collapse dramatically:

$$\begin{aligned} \mathrm{OUH} &\subseteq \mathrm{P}^{\mathrm{TALLY}\cap\mathrm{NP}} \subseteq \mathrm{P}_{\|}^{\mathrm{ONP}}, \\ \mathrm{OUH}^{\bullet} &\subseteq \mathrm{OGTD}[\mathrm{NP}] \\ &= \mathrm{OUGTD}[\mathrm{FP}^{\mathrm{ONP}}, \text{co-ONP}, \mathrm{P}] \subseteq \mathrm{OUM}, \end{aligned} \tag{1}$$

while at the base levels $\exists^{\mathrm{ou}} \cdot \text{co-OUP} = \mathrm{OUP} \cup \text{co-OUP}$, $\mathrm{OUP}^{\mathrm{prOUP}[1]} = \mathrm{OUP}^{\mathrm{OUP}[1]}$, and $\mathrm{SOUP} = \mathrm{OUP} \cap \text{co-OUP}$.

Contributions and Explicit Scope. We formally verify and audit the *finite combinatorial, order-theoretic, and oracle-consistency core* of [4]. We explicitly do *not* formalize polynomial-time oracle Turing machines, asymptotic advice polynomials, or uniform circuit encodings inside Lean 4; all complexity-class inclusions remain Hirsch's [4]:

1. **Zero-dependency Lean 4 kernel (§3):** 18 constructive theorems in `ObliviousUnambiguousHierarchy.lean` (Lean 4.35.0 [2]), 0 `sorry`, 0 warnings, 0 `Classical.choice`, and **11 zero-axiom theorems**, with every formal signature and inductive predicate stated in Table 1.
2. **Pure-stdlib exact verifier (§4):** `OUH-CERT-1.0`, formally specified in Definition 2, auditing four finite truth-table modes with zero floating-point arithmetic.
3. **Empirical atlas & red-team audit (§5):** 16 canonical certificates (0.038–0.789 ms on Windows 11 / Python 3.13), four exact sweeps ($q = 2..8$, $h = 4..12$, $|\Pi_{\mathrm{out}}| = 1..6$), and 60/60 rejected live adversarial forgeries.

2 Formal Definitions and Combinatorial Core

Definition 1 (Oblivious Classes, Operators, and Pairing [4])**.** *For length-n inputs $x \in \{0,1\}^n$ and polynomial-length witnesses $w \in \{0,1\}^{p(n)}$:*

- ONP *(oblivious* NP*): $L \in$ ONP iff a polynomial-time V and length-only advice w_n satisfy $x \in L \iff V(x, w_n) = 1$ and $x \notin L \Rightarrow \forall w, V(x,w) = 0$.* co-ONP *is the complement class.*
- OUP *(oblivious* UP*): $L \in$ OUP iff $L \in$ ONP and on every non-empty slice $L \cap \{0,1\}^n$, w_n is the* unique *w accepted for any $x \in \{0,1\}^n$ (if $L \cap \{0,1\}^n = \emptyset$, no w is*

*Independent researcher; code, Lean 4 proofs, and certificates: MIT license; manuscript and figures: CC-BY-4.0. Prepared with transparent AI assistance via Google Antigravity; all Lean 4 theorems, verifiers, and tests were independently compiled and verified locally. Repository: https://github.com/lluiseriksson/oblivious-unambiguous-hierarchy-certificates.

accepted). prOUP *is the promise-problem analogue on* $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$.

- TALLY*: unary languages over* $\{1\}^*$. SOUP *(symmetric* OUP*): $L \in$ SOUP iff a single unique length-n witness w_n certifies* both $x \in L$ *and* $x \notin L$*;* $\mathrm{SOUP}^{\mathrm{SOUP}}$ *denotes* SOUP *with a* SOUP *oracle.*
- OGTD[$\mathcal{C}$] *(oblivious generate-test-decide): L is decided by a triple (G, T, D) with generator $G(1^n)$ outputting advice $a \in \mathrm{Out}_G(n)$, tester T with $\mathrm{Out}_G(n) \subseteq \mathrm{Acc}_T(n)$, and decider $D(x, a) = L(x)$ for all $a \in \mathrm{Acc}_T(n)$.* OUGTD[$\mathcal{F}, \mathcal{T}, \mathcal{D}$] *further enforces $|\mathrm{Acc}_T(n)| \leq 1$;* OUM *denotes oblivious unambiguous Merlin–Arthur/advice classes containing* $\mathrm{OUGTD}[\mathrm{FP}^{\mathrm{ONP}}, \text{co-ONP}, \mathrm{P}]$.
- *Doubled Cantor pairing:* $\mathrm{cantor}_2(a, b) := (a + b)(a + b + 1) + 2b$ *for $a, b \in \mathbb{N}$, which is injective and satisfies* $\mathrm{cantor}_2(a, b) < k(k + 1)$ *whenever* $a + b < k$.

Four exact finite invariants with explicit preconditions drive (1):

1. *Tally census & bit extraction (Lemma 2.8, Theorem 4.1):* Knowing the census count $c_m = |C_m|$ of a finite set $C_m \subseteq \{0,1\}^m$, any subset $S \subseteq C_m$ with $|S| = c_m$ equals C_m uniquely. Given an OUP verifier V with unique witness w_n on a non-empty length-n slice, the p parallel tally predicates $W_i \iff \exists x, w : V(x, w) = 1 \wedge w_i = 1$ recover w_n bit-for-bit (and return 0^p when $L \cap \{0,1\}^n = \emptyset$).
2. *Krentel lex-max transcript (Lemma 3.1):* Let $G(1^n)$ be a deterministic depth-q adaptive query tree whose j-th query depends only on the length-$(j-1)$ answer prefix $a_{<j}$, and let $a^* \in \{0,1\}^q$ be the true oracle transcript (so every 0-bit of a^* queries a NO instance). Define $R(a)$ to hold iff every 1-bit of a queries a YES instance (matching $G(1^n)$ along prefix $a_{<j}$). At the first mismatch index j between any $a \in R$ and a^*, the prefixes $a_{<j} = a^*_{<j}$ issue the identical j-th query; since $a^*_j = 0$ implies a NO instance whereas $a_j = 1$ certifies a YES instance, necessarily $a_j = 0 < 1 = a^*_j$, so $a <_{\mathrm{lex}} a^*$. Hence a^* is the unique lexicographical maximum of R, recovered in q ONP prefix probes and uniquely accepted by the co-ONP tester
$$T(a) \iff R(a) \wedge \forall b \in \{0,1\}^q,\ (a <_{\mathrm{lex}} b \Rightarrow \neg R(b)). \tag{2}$$
3. *Fixed-pair dot-operator collapse (Theorem 6.2):* Suppose $L(x) \iff \exists^{\mathrm{ou}} w \in \{0,1\}^p\, R(x, w)$ $(p \geq 1)$ where $\overline{R} \in$ OUP via verifier V with unique length-oblivious witness v_N $(V((x, w), v) \iff \neg R(x, w) \wedge v = v_N)$. Since at most one $w \in \{0,1\}^p$ satisfies $R(0^n, w)$, at least one of $(0^n, 0^p)$ and $(0^n, 1^p)$ lies in $\overline{R}$. Consequently, the collapsed verifier
$$V'(x, (w, v)) := \big(V((0^n, 0^p), v) \vee V((0^n, 1^p), v)\big) \wedge \neg V((x, w), v) \tag{3}$$
pins $v = v_N$ on anchor 0^n and transfers v_N by length-obliviousness to arbitrary $x \in \{0,1\}^n$, forcing $\neg V((x, w), v_N) \iff R(x, w)$ and accepting solely (w_n, v_N).
4. *Single loose query & SOUP tagging (Lemma 6.4, Proposition 6.6):* For a single-query machine $V^O(x, w)$ querying $q(x, w)$ with outcomes $v_0 = V^0(x, w), v_1 = V^1(x, w)$, define the sensitive set $T = \{(x, w) : v_0 \neq v_1\}$. Loose oracle consistency across all extensions O of $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$ requires $V^{O_0}(x, w) = V^{O_1}(x, w)$ whenever $q(x, w) \notin \Pi_{\mathrm{YES}} \cup \Pi_{\mathrm{NO}}$, forcing $q(T) \subseteq \Pi_{\mathrm{YES}} \cup \Pi_{\mathrm{NO}}$; insensitive pairs $(v_0 = v_1)$ eliminate the query entirely. For $L \in$ OUP$\cap$co-OUP with verifiers Y (unique witness w_Y for L) and Z (unique witness w_Z for $\overline{L}$), tagging $\mathrm{Tag}(b, w) := (b = 1 \wedge Y(0^n, w)) \vee (b = 0 \wedge Z(0^n, w))$ pins a unique pair (b^*, w^*) on 0^n; by length-obliviousness, every $x \in \{0,1\}^n$ is decided using the *same* witness (b^*, w^*) via $Y(x, w^*)$ when $b^* = 1$ or $\neg Z(x, w^*)$ when $b^* = 0$.

3 Zero-Dependency Lean 4.35.0 Kernel

We formalize 18 constructive theorems in `lean/ObliviousUnambiguousHierarchy.lean` (Lean 4.35.0, zero external libraries), audited via `lean/AuditAxioms.lean` with 0 warnings, 0 `sorry`, 0 `Classical.choice`, and 0 `Lean.ofReduceBool` (11 zero-axiom theorems, Table 1). Auxiliary inductive definitions in the module are:

- `cantor2` a b $:= (a + b)(a + b + 1) + 2b$; `popcount` : `List Bool` $\to \mathbb{N}$ counts `true` entries; `BoolSubset` s c holds pointwise on equal-length lists ($s_i =$ `true` $\Rightarrow c_i =$ `true`).
- `LexLt` a b is MSB-first strict lexicographic order on equal-length `List Bool`; `CertifiedAgainst` a t enforces inductively on equal-length lists that ($a_0 =$ `true` $\Rightarrow t_0 =$ `true`) $\wedge$ ($a_0 = t_0 \Rightarrow$ `CertifiedAgainst` a' t').

4 Formal Specification of `OUH-CERT-1.0`

The pure-stdlib verifier `OUH-CERT-1.0` (`src/ouh_certifier.py`, `src/ouh_exact.py`) uses integer and boolean truth-table arithmetic exclusively, recursively rejecting IEEE-754 floats, booleans-as-integers, unknown keys, duplicate entries, invalid bitstrings, and vacuous witnesses across four formally specified modes:

Definition 2 (`OUH-CERT-1.0` JSON Schemas & Acceptance Predicates)**.** *Every certificate contains* `schema="OUH-CERT-1.0"`*,* `mode`*,* `cert_id`*, and mode-specific fields accepted iff:*

1. ***Mode 1 (`tally_census_and_oblivious_extraction`):*** *Given bit-lengths (n, p), universe size h, tally bits $C_m \in \{0,1\}^h$, subset $S \subseteq C_m$, and an* OUP *truth table $V : \{0,1\}^n \times \{0,1\}^p \to \{0,1\}$, accepts iff $S \subseteq C_m \wedge |S| = |C_m| \Rightarrow S = C_m$,* cantor_2 *is injective on claimed pairs, V has at most one accepting witness*

Table 1: Complete 18-theorem Lean 4.35.0 kernel in `ObliviousUnambiguousHierarchy.lean` (0 `sorry`, 0 warnings, 0 `Classical.choice`, 11 zero-axiom theorems), listing exact formal signatures and statements.

#	Lean 4 Theorem Identifier	Exact Formalized Type / Statement	Ref. [4]	Axioms
1	`cantor_pairing_injective`	$\texttt{cantor2}\ a_1\ b_1 = \texttt{cantor2}\ a_2\ b_2 \Rightarrow a_1 = a_2 \wedge b_1 = b_2$	Lem. 2.8	`[p, Q]`
2	`tally_census_subset_exact`	$\forall s, c : \texttt{List Bool}, \texttt{BoolSubset}\ s\ c \Rightarrow \texttt{popcount}\ s = \texttt{popcount}\ c \Rightarrow s = c$	Lem. 2.8	`[p, Q]`
3	`tally_census_bit_determined`	$\texttt{BoolSubset}\ s\ c \Rightarrow \texttt{popcount}\ s = \texttt{popcount}\ c \Rightarrow s[i]? = c[i]?$	Lem. 2.8	`[p, Q]`
4	`oblivious_witness_bit_extraction`	$V(x_0, w_n) \wedge (\forall x, w, V(x, w) \Rightarrow w = w_n) \Rightarrow ((\exists x, w, V(x, w) \wedge \text{bit}\ w\ i) \iff \text{bit}\ w_n\ i)$	Thm. 4.1	none
5	`oblivious_empty_slice_zero_witness`	$(\forall x, w, \neg V(x, w)) \Rightarrow \neg(\exists x, w, V(x, w) \wedge \text{bit}\ w\ i = \texttt{true})$	Thm. 4.1	none
6	`lex_max_transcript_sound_and_unique`	$\forall a, a^* : \texttt{List Bool}, \texttt{CertifiedAgainst}\ a\ a^* \Rightarrow a = a^* \vee \texttt{LexLt}\ a\ a^*$	Lem. 3.1	`[p]`
7	`prefix_search_bit_recovery`	$(\exists r, \texttt{CertifiedAgainst}\ (p +\!\!+ \texttt{true} :: r)\ (p +\!\!+ b :: t)) \iff b = \texttt{true}$	Lem. 3.1	`[p]`
8	`co_onp_tester_unique_acceptance`	$\lvert a_1\rvert = \lvert a_2\rvert = q \wedge T_R(a_1) \wedge T_R(a_2) \Rightarrow a_1 = a_2\ (T_R(a) := R(a) \wedge \forall b, a <_{\text{lex}} b \Rightarrow \neg R(b))$	Lem. 3.1	`[p, Q]`
9	`ogtd_chain_inclusion_sound`	$(\forall a, \text{Out}_G\ a \Rightarrow \text{Acc}_T\ a) \wedge (\forall a, \text{Acc}_T\ a \Rightarrow \forall x, D\,x\,a = L\,x) \Rightarrow \text{Out}_G\ a \Rightarrow D\,x\,a = L\,x$	Def. 2.4	none
10	`dot_op_fixed_pair_refutation`	$w_0 \neq w_1 \wedge (\forall u, v, R(x_0, u) \Rightarrow R(x_0, v) \Rightarrow u = v) \wedge \text{Dec}(R(x_0, w_0)) \Rightarrow \neg R(x_0, w_0) \vee \neg R(x_0, w_1)$	Thm. 6.2	none
11	`dot_op_collapse_unique_witness`	$V\text{-uniq}(v_N) \wedge (\neg R \Rightarrow V(\cdot, v_N)) \wedge R\text{-uniq}(w_n) \wedge \text{Dec}(R) \wedge V'(x, (w, v)) \Rightarrow w = w_n \wedge v = v_N$	Thm. 6.2	none
12	`dot_op_collapse_soundness`	$(\neg R(x_0, w_0) \vee \neg R(x_0, w_1)) \wedge (V(\cdot, v_N) \iff \neg R) \wedge (L \iff R(\cdot, w_n)) \wedge \text{Dec} \Rightarrow (L(x) \iff V'(x, (w_n, v_N)))$	Thm. 6.2	none
13	`dot_op_union_distributive`	$(\exists w, C(w) \vee D(w)) \iff (\exists w, C(w)) \vee (\exists w, D(w))$	Cor. 6.3	none
14	`single_loose_query_in_promise`	$v_0 \neq v_1 \wedge (\neg P \Rightarrow v_0 = v_1) \wedge (P \vee \neg P) \Rightarrow P$	Lem. 6.4	none
15	`single_loose_query_insensitive_elim`	$v_0 = v_1 \Rightarrow \forall b : \texttt{Bool}, (\text{if}\ b\ \text{then}\ v_1\ \text{else}\ v_0) = v_0$	Thm. 6.5	none
16	`soup_shared_witness_tag_exact`	$(Y \iff L \wedge w = w_Y) \wedge (Z \iff \neg L \wedge w = w_Z) \wedge \text{Dec}(L\,x_0) \Rightarrow \exists!(b^*, w^*),\ \text{Tag}(b^*, w^*)$	Prop. 6.6	none
17	`soup_self_low_composition`	$(P_1(w_1) \iff w_1 = w_1^*) \wedge (P_2(w_1^*, w_2) \iff w_2 = w_2^*) \Rightarrow (P_1(w_1) \wedge P_2(w_1, w_2) \iff w_1 = w_1^* \wedge w_2 = w_2^*)$	Prop. 6.6	none
18	`krentel_transcript_space_bounds`	$\forall a, b, k : \mathbb{N},\ a + b < k \Rightarrow \texttt{cantor2}\ a\ b < k(k+1)$	Lem. 2.8	`[p, Q]`

Axiom key: `none` = zero axioms (11); `[p]` = `[propext]` (2); `[p, Q]` = `[propext, Quot.sound]` (5). $\text{Dec}(P)$ denotes explicit decidability hypothesis $P \vee \neg P$.

across all x, and bit-by-bit tally extraction recovers w_n (or 0^p on empty slices).

2. ***Mode 2 (`krentel_lex_max_ogtd_cert`):*** *Given adaptive query depth $q \in [2, 8]$, explicit prefix-deterministic query map $Q : \{0,1\}^{<q} \to U$, and NP-oracle indicator $O : U \to \{0,1\}$, exhaustively enumerates all 2^q transcripts, computes $R = \{a : \forall j, a_j = 1 \Rightarrow O(Q(a_{<j})) = 1\}$, and accepts iff $a^* = \max_{\text{lex}} R$, all q prefix probes match a^*, and $\lvert\{a : T(a)\}\rvert = 1$ under (2).*
3. ***Mode 3 (`dot_operator_collapse_exact`):*** *Given (n, p, r) $(p \geq 1)$ and finite truth tables for $R(x, w)$ and inner* OUP *verifier $V((x, w), v)$ for $\overline{R}$, accepts iff V accepts solely v_N on $\overline{R}$, R accepts at most one w_n, $(0^n, 0^p) \in \overline{R} \vee (0^n, 1^p) \in \overline{R}$, and V' in (3) accepts solely (w_n, v_N) for $x \in L$.*
4. ***Mode 4 (`single_loose_query_and_soup_exact`):*** *Given promise partition $(\Pi_{\text{YES}}, \Pi_{\text{NO}}, \Pi_{\text{out}})$, single-query verifier $(q(x, w), V^0, V^1)$, and complementary* OUP *verifiers (Y, Z) for $(L, \overline{L})$, exhaustively replays all $2^{\lvert\Pi_{\text{out}}\rvert}$ consistent oracle extensions O, accepting iff V^O is consistent across all extensions, $q(T) \subseteq \Pi_{\text{YES}} \cup \Pi_{\text{NO}}$, insensitive pairs eliminate the query, and $\text{Tag}(b, w)$ on 0^n yields a unique shared witness (b^*, w^*) deciding all $x \in \{0,1\}^n$.*

5 Empirical Atlas and Red-Team Audit

Running `scripts/run_all_experiments.py` (Windows 11, Python 3.13 pure-stdlib) produces 16 canonical certificates (4 per mode) and four exact combinatorial sweeps, audited end-to-end by `scripts/verify_all.py` with SHA-256 verification across all 41 tracked repository files.

Krentel Transcripts and Tally Census (Fig. 1). Across adaptive depths $q = 2 \ldots 8$ ($2^q = 4 \ldots 256$), $\lvert R\rvert$ spans 7 (sparse), 55 (balanced), and 104 (dense) transcripts at $q = 8$, yet in 100% of instances ONP prefix search recovers a^* in q probes and tester (2) achieves $\lvert\text{Acc}_T\rvert = 1$. In Mode 1 (0.038–0.084 ms), exact bit-by-bit tally extraction recovers both non-empty witnesses (up to $n = 4, p = 6$, $\lvert L\rvert = 11$) and empty-slice 0^5, while census uniqueness collapses $\binom{12}{6} = 924$ same-size subsets (out of $2^{12} = 4096$) to 1.

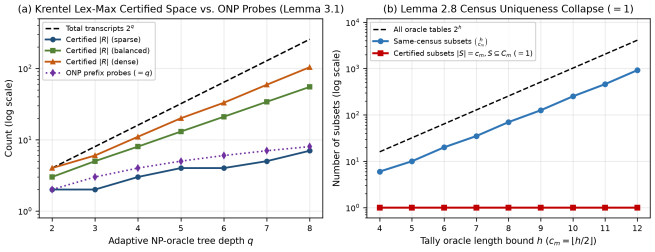

Figure 1: (a) Exact Lemma 3.1 Krentel certified transcript count $\lvert R\rvert$ across depths $q = 2 \ldots 8$ (sparse, balanced, dense regimes) versus 2^q total transcripts and q ONP prefix probes. (b) Exact Lemma 2.8 tally-census collapse from $\binom{h}{c_m}$ same-size subsets to exactly 1 certified subset ($h = 4 \ldots 12$).

Dot Collapse, Loose Extensions, and SOUP (Fig. 2). In Mode 3 (0.068–0.183 ms), fixed-pair refutation collapses up to $2^{2+3} = 32$ witness pairs to (w_n, v_N) at $(n, p, r) = (4, 2, 3)$ ($\lvert L\rvert = 9$). In Mode 4 (0.117–0.789 ms), exhaustive auditing across up to $2^5 = 32$ consistent extensions (and $2^6 = 64$ in Experiment 3, 0.653 ms) confirms that all 24/32 insensitive pairs querying Π_{out} are safely eliminated while every sensitive pair lies strictly inside $\Pi_{\text{YES}} \cup \Pi_{\text{NO}}$, and certifies Proposition 6.6 shared SOUP witnesses for both $0^n \in L$ (`1|wY`) and $0^n \notin L$ (`0|wZ`).

Replay Atlas and Red-Team Audit (Fig. 3, Table 2). All 16 canonical certificates replay in 0.038–0.789 ms with zero floats. Our adversarial suite

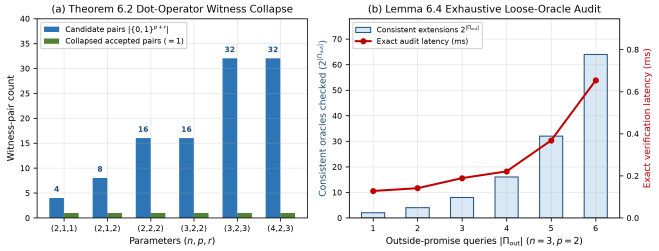

Figure 2: (a) Theorem 6.2 dot-operator collapse across (n, p, r): fixed-pair refutation collapses 2^{p+r} candidate pairs to exactly 1 unique witness (w_n, v_N). (b) Exhaustive Lemma 6.4 loose-oracle audit across $|\Pi_{\text{out}}| = 1 \ldots 6$ ($2 \ldots 64$ consistent extensions, 0.128–0.653 ms).

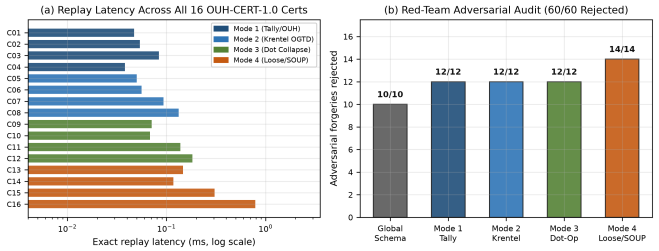

Figure 3: (a) Exact zero-float replay latency across all 16 canonical `OUH-CERT-1.0` certificates (0.038–0.789 ms). (b) Red-team adversarial audit rejecting 60/60 live forgery attacks across all modes.

(`tests/test_suite.py`) tests 60 live forgery attacks—including float/bool type confusion, ambiguous or non-oblivious witnesses (e.g. two distinct $w_1 \neq w_2$ accepted in Mode 1), tampered Krentel prefix probes (Mode 2), vacuous or mismatched dot-collapse pairs (Mode 3), sensitive queries leaking into Π_{out} and non-complementary SOUP verifiers (Mode 4)—achieving **60/60 rejections** with zero bypasses.

6 Conclusion

We delivered an 18-theorem zero-dependency Lean 4 kernel (11 zero-axiom theorems, 0 `Classical.choice`) with full formal specifications and the exact finite verifier `OUH-CERT-1.0` for the finite combinatorial core of Hirsch's input-oblivious unambiguous hierarchy collapses [4]. Mechanizing uniform polynomial-time oracle Turing machines in Lean 4 to bridge these finite combinatorial lemmas to asymptotic complexity-class inclusions remains open.

Table 2: Summary of all 16 canonical `OUH-CERT-1.0` certificates (representative witnesses) and 60/60 adversarial audit.

Verification Mode	Concrete Certified Range	ms	Red-Team
M1: Tally Census & Extraction (4)	$h \leq 12$ ($\binom{12}{6}$=924→1), $n \leq 4, p \leq 6$, 0^5 empty	0.038–0.084	12/12
M2: Krentel Lex-Max & OUGTD (4)	$q \in [2, 8]$, $\lvert R\rvert \in \{7, 55, 104\}$, $\lvert \text{Acc}_T\rvert = 1$	0.050–0.133	12/12
M3: Dot-Operator Collapse (4)	$(n, p, r) \leq (4, 2, 3)$, 32→1 unique (w_n, v_N)	0.068–0.183	12/12
M4: Loose Query & SOUP (4)	2^5=32 ext., 24/32 insens., `1|wY` / `0|wZ`	0.117–0.789	14/14
Global Schema / Type / Bitstring	Float, bool-as-int, duplicate, bad bits	–	10/10
Total (16 certs)	**All 4 finite mechanisms of [4]**	**0.038–0.789**	**60/60**

References

[1] Scott Aaronson and Andrew Drucker. A full characterization of quantum advice. *SIAM Journal on Computing*, 43(3):1131–1183, 2014.

[2] Leonardo de Moura and Sebastian Ullrich. The Lean 4 theorem prover and programming language. In *Automated Deduction – CADE 28*, pages 625–635. Springer, 2021.

[3] Lance Fortnow, Rahul Santhanam, and Ryan Williams. Fixed-polynomial size circuit bounds. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity (CCC)*, pages 19–26, 2009.

[4] Edward A. Hirsch. A note on input-oblivious unambiguous hierarchies. *arXiv preprint arXiv:2611.33126*, 2026.

[5] Edward A. Hirsch and Dmitry Sokolov. On the unambiguous polynomial hierarchy and promise problems. *Information Processing Letters*, 115(2):317–322, 2015.