

QuWARP Reconciled: An Analytical Cost-Model Assessment of Workload-Level Reuse Planning for Quantum Circuit Simulation

QuWARP Reconciled: An Analytical Cost-Model Assessment of Workload-Level Reuse Planning for Quantum Circuit Simulation

Abstract

Classical simulation of quantum circuits increasingly appears as repeated-run workloads — variational quantum eigensolver (VQE) sweeps, noisy multishot studies, and quantum error-correction (QEC) cycles — rather than isolated circuit executions. The QuWARP planner (arXiv:2609.23664v1) proposes workload-level reuse: identifying shared circuit prefixes across related tasks, materializing exact typed boundary artifacts, and reusing them when a narrow continuation-legality guardrail permits, with abstention and EXPLAIN-style auditability otherwise. This paper provides an independent analytical treatment of that design. We formalize a two-parameter cost model — shared-prefix fraction f and materialization overhead fraction α — and derive closed-form expressions for per-run and amortized speedup, break-even conditions, and feasibility constraints. We show that the reported speedup range of $2.95\times$ – $32.84\times$ implies shared-prefix fractions between roughly 0.66 and 0.98 under small overheads, that any reported speedup above $20\times$ forces materialization overhead below $\sim 5\%$ of baseline run cost, and that amortized speedup over R-run workloads grows toward the ceiling $1/(1-f+\alpha)$. We further analyze failure modes: prefix drift under parameter updates, legality violations at non-Clifford boundaries, and audit-trail overhead. Our results are model-derived characterizations of the published claims, not new measurements; we state this explicitly and identify what experiments would falsify the underlying reuse hypothesis.

1. Introduction

Classical simulation of quantum circuits is a foundational tool for the quantum computing stack: it underpins algorithm development, noise studies, and verification of quantum error correction (QEC) decoders before hardware time is spent. The dominant engineering tradition optimizes a single circuit execution — better gate kernels, better parallelism, better memory layouts. Yet practitioners rarely run a circuit once. A VQE experiment evaluates a parameterized ansatz hundreds or thousands of times with slowly changing parameters; a noise study executes the same logical circuit across many shot counts and noise realizations; a QEC study runs many rounds of nearly identical syndrome-extraction structure.

The QuWARP paper [1][2] observes that this repeated-run structure is an optimization opportunity that per-execution simulators leave on the table: closely related tasks share circuit prefixes, and if a simulator can checkpoint the state at the end of a shared prefix — an "exact typed boundary artifact" — later tasks can resume from it instead of re-simulating from the $|0\dots 0\rangle$ state. QuWARP wraps a bounded simulator execution surface (statevector mode and a stabilizer-hybrid mode), performs workload-level planning, guards each reuse with a continuation-legality check, abstains when reuse is unprofitable, and emits EXPLAIN-style traces so every reuse, abstention, or refusal is auditable. The reported headline result is a $2.95\times$ – $32.84\times$ speedup over a per-task Qrack-based denominator on reuse-positive workloads.

This paper does not replicate QuWARP. Instead, it builds a minimal explicit cost model, derives what the reported numbers *require* to be true, characterizes the planner's profitability envelope, and enumerates the conditions under which the approach must fail. Our contributions are:

1. A two-parameter analytic model (shared-prefix fraction f , materialization overhead fraction α) with closed-form speedup, break-even, and feasibility conditions (Sections 3–4).
2. Explicit arithmetic showing what the reported $2.95\times$ – $32.84\times$ range implies about f and α , including a hard feasibility constraint that rules out parameter combinations (Section 4).
3. An amortized multi-run analysis showing how speedup saturates with workload length R , with worked numbers (Section 4).
4. A falsifiability analysis: which measurements would refute the reuse hypothesis, and which structural failure modes bound the gains (Section 6).

Throughout, we distinguish sharply between numbers we compute here from stated assumptions and numbers quoted from [1][2]. No new simulations or measurements are presented.

2. Background and Related Work

Workload-level reuse in simulation. The direct subject of this study is QuWARP itself [1][2], which frames simulation as a workload-planning problem rather than a per-circuit compilation problem. Its distinctive commitments are threefold: (i) reuse is planned at the workload level, across tasks, not within a single circuit; (ii) continuation legality acts as a narrow correctness guardrail — a reuse is permitted only when resuming from a boundary artifact is provably equivalent to fresh execution; and (iii) every decision is auditable via EXPLAIN-style traces with provenance and realized-cost summaries. The abstract reports $2.95\times$ – $32.84\times$ speedups over a per-task Qrack denominator on reuse-positive workloads, across statevector and stabilizer-hybrid modes. The qualifier "reuse-positive" is important: it conditions the headline numbers on workloads where reuse is profitable, a selection we examine critically in Section 6.

Per-execution simulation engineering. The complementary tradition optimizes single runs. De Matteis and de Renzis [7] survey fast quantum circuit simulation built on hardware-accelerated general-purpose libraries, emphasizing ease of use, implementation, and maintainability alongside raw parallelism in Python-centric software stacks — a reminder that simulator engineering trades generality against peak throughput, and that a workload planner like QuWARP is orthogonal to (and composable with) kernel-level acceleration. The simulability analysis of [8] frames classical simulation around the precise notion of "classical simulation" required — exact amplitude estimation versus weak sampling. This matters directly for QuWARP: reuse of an exact typed boundary artifact is only legal when the downstream task's required simulation notion (and mode: statevector vs. stabilizer-hybrid) matches the artifact's type. The continuation-legality guardrail in [1][2] operationalizes exactly this typing discipline.

Circuit-structure optimization. The structure-optimization method for parameterized circuits [3] simultaneously optimizes ansatz structure and parameter values with small computational overhead, suiting NISQ devices. VQE sweeps of exactly this kind are QuWARP's prime reuse-positive workload, since successive structure/parameter evaluations produce circuits differing only in a few late gates. The Krotov-control monotonicity proof [4] is a different variational setting (open-system control), but it illustrates the same structural phenomenon: iterative optimizers generate sequences of nearly identical circuit executions, which is precisely the workload shape a reuse planner targets. Cost-reduction work on quantum Boolean circuits [6] — multi-strategy synthesis of Positive-Polarity Reed-Muller expansions into multi-controlled Toffoli gates — operates at the synthesis level; its cost metric (gate count) is analogous to QuWARP's realized-cost summaries, but it rewrites circuits rather than reusing execution state, so the two are complementary layers of the same stack.

Mapping and physical-level optimization. TETRIS-Q [10] tackles NISQ circuit mapping with tiling-based transient-fault reduction and parallelism optimization, jointly handling SWAP overhead and decoherence exposure where compilers typically optimize routing and placement independently. This is the hardware-mapping analogue of QuWARP's software-simulation planning: both recognize that treating executions in isolation wastes an optimization dimension. Notably, TETRIS-Q-style noise-aware mapping changes the circuit actually executed per run, which can erode prefix sharing — a failure mode we return to in Section 6.

Energy accounting as a system metric. The JPCUB line [11][12] proposes joules-per-solution — total system energy per correct answer — as a physics-grounded, system-level benchmark across quantum platforms, with a competitive landscape built from published specifications. This matters for QuWARP because reuse planning changes total work performed per solution; a $2.95\times$ – $32.84\times$ wall-clock speedup, if it translates to energy, would move a simulator's joules-per-solution proportionally, making workload-level reuse relevant to energy benchmarking, not just latency. The QWAV strategy document [11] frames the governance path for such a benchmark, indicating that reuse-aware simulation cost could become a reportable quantity in consortium settings.

Physical-circuit context and scope exclusions. The quantum electromagnetic circuits review [9] covers Hamiltonian construction for general (possibly dissipative) physical circuits; it concerns superconducting hardware rather than gate-level simulation, but contextualizes the QEC and control workloads whose repeated-cycle structure motivates workload-level reuse. The QuiX Quantum due-diligence report [13] exemplifies multi-source verification practice for photonic platforms; we adopt a similar skeptical, source-checking posture toward the single-source speedup claims of [1][2]. Finally, the access-point simulation paper [5] was withdrawn by arXiv administrators for fictitious content and pseudonymous submission; it is cited here only to record its detection and exclusion from any substantive reliance — a cautionary datapoint for automated bibliographic triage.

Collectively, these works illustrate a landscape where optimization occurs at many layers — gate synthesis, control theory, hardware mapping, system-level benchmarking — and QuWARP occupies a distinct niche by orchestrating *cross-task* reuse. We note that the bibliography is thin on independent replications of QuWARP, which constrains how strongly any analysis — including ours — can support the underlying empirical claims.

3. Methods

3.1 Workload and planner model

Following [1][2], we model a workload as an ordered set of R tasks $\{T_1, \dots, T_R\}$, each task a circuit $C_i = P_i \cdot S_i$, where S_i is a shared prefix (a gate sequence identical, up to the planner's equivalence notion, to prefixes of other tasks) and P_i is a task-specific suffix. The planner:

1. **Prefix identification:** computes, across the workload, the maximal shared prefixes and their lengths relative to full circuit length.
2. **Materialization decision:** for the first task executing a given prefix, decides whether to checkpoint the simulator state at the prefix boundary as a typed boundary artifact (exact statevector, or a stabilizer-hybrid boundary representation).
3. **Continuation legality check:** verifies that resuming from the artifact into task i 's suffix is equivalent to fresh execution — e.g., the artifact's representation supports all gates in P_i , and no measurement/adaptivity in P_i depends on information discarded at the boundary.
4. **Profitability test (abstention):** reuses only when the modeled cost of resuming is below the modeled cost of fresh execution; otherwise abstains and runs fresh.
5. **Auditing:** emits an EXPLAIN-style trace recording the decision, its provenance, and the realized cost.

We define the two model parameters:

- **$f \in [0,1]$: shared-prefix fraction** — the fraction of a representative task's execution cost covered by a shared prefix and therefore avoidable via reuse.
- **$\alpha \geq 0$: materialization overhead fraction** — the cost of creating (and, where applicable, reading) the boundary artifact, expressed as a fraction of one fresh task execution's cost.

3.2 Cost model

Let T be the cost of one fresh task execution under the per-task denominator (the Qrack-based baseline of [1][2]). Then:

- **Fresh (no reuse)**: each task costs T ; workload cost $C_{\text{fresh}} = R \cdot T$.
- **Reuse, per resumed task**: cost $(1-f) \cdot T + \alpha \cdot T$. The first task in a prefix group pays $T + \alpha \cdot T$ (fresh execution plus checkpoint write).

Per-run speedup for a resumed task:

$$S_{\text{run}} = 1/(1-f+\alpha). \text{ (Eq. 1)}$$

Amortized workload speedup over R tasks with a single prefix group (one materialization, $R-1$ resumptions):

$$S_R = R / [1+\alpha + (R-1)(1-f+\alpha)]. \text{ (Eq. 2)}$$

Break-even (reuse no worse than fresh) for a resumed task requires $1-f+\alpha \leq 1$, i.e.:

$$f \geq \alpha. \text{ (Eq. 3)}$$

Per-run speedup is bounded: $S_{\text{run}} \leq 1/\alpha$ (approached as $f \rightarrow 1$) and $S_{\text{run}} \geq 1$ whenever $f \geq \alpha$.

3.3 Stabilizer-hybrid mode

In stabilizer-hybrid execution, Clifford-fragment gates run in polynomial time on a tableau while non-Clifford gates force statevector expansion on a reduced support. A boundary artifact taken at a Clifford/non-Clifford interface can be exponentially cheaper to materialize than a full statevector artifact when the non-Clifford support is small. We model this qualitatively: the effective α for stabilizer-hybrid artifacts can be far smaller than for dense statevector artifacts, consistent with QuWARP evaluating both modes [1][2]; we do not assign it a number, since the source abstract gives no mode-split data.

3.4 Validity of the model

The model is deliberately minimal: it assumes homogeneous task costs, a single dominant prefix group, and additive overhead. Real workloads have heterogeneous task costs and multiple prefix groups; Eq. 2 then applies per group with R replaced by group size, and workload speedup is a weighted combination. We use the minimal model because the published abstract of [1][2] provides only a speedup range, and a two-parameter model is the richest model that range can constrain.

4. Analysis

All inputs below are either (a) quoted from the QuWARP abstract [1][2] or (b) assumptions we state explicitly. Every arithmetic step is shown.

4.1 What the reported speedup range implies

Input (a): reported speedups $S \in \{2.95, 32.84\}$ on reuse-positive workloads [1][2]. **Input (b, assumption):** materialization overhead $\alpha = 0.05$ as a mid-range guess for dense statevector artifacts; we also test $\alpha = 0.01$.

From Eq. 1, $S = 1/(1-f+\alpha) \implies f = 1 - 1/S + \alpha$.

Case $S = 2.95$, $\alpha = 0.05$: $1/S = 0.33898$; $f = 1 - 0.33898 + 0.05 = 0.711017$. A $2.95\times$ speedup at 5% overhead requires $\sim 71.1\%$ of each run's cost in the shared prefix.

Case $S = 32.84$, $\alpha = 0.05$: $1/S = 0.030451$; $f = 1 - 0.030451 + 0.05 = 1.019549$. **$f > 1$ is infeasible.** Therefore, if the $32.84\times$ workload ran in statevector mode with a dense artifact, α cannot be 0.05. Solving for the maximum feasible α with $f \leq 1$: $\alpha \leq 1/32.84 = 0.0304507$. The $32.84\times$ result requires materialization overhead below $\sim 3.05\%$ of a fresh run's cost.

Case $S = 32.84$, $\alpha = 0.01$: $f = 1 - 0.030451 + 0.01 = 0.979549$. With 1% overhead, $\sim 97.95\%$ of the run's cost must be shared-prefix work — plausible for QEC-style workloads where per-round syndrome-extraction structure dominates and only decoding differs, but a strong *derived requirement*, not a measurement.

Case $S = 2.95$, $\alpha = 0.01$: $f = 1 - 0.33898 + 0.01 = 0.671017$. $\sim 67.1\%$ shared prefix suffices at 1% overhead.

Summary of the feasible (f, α) region: for the low end ($2.95\times$), $f \gtrsim 0.66$ – 0.71 for $\alpha \in [0.01, 0.05]$; for the high end ($32.84\times$), $f \gtrsim 0.98$ and $\alpha \leq 0.0305$. The reported range is internally consistent only if the best workload has near-total prefix sharing and very cheap artifact materialization — exactly the profile of stabilizer-hybrid QEC cycles, and hard to reach for dense statevector VQE sweeps.

4.2 Amortized speedup saturation

Assumption: $R = 100$ tasks in one prefix group; $f = 0.90$; $\alpha = 0.02$.

Eq. 2: $1-f+\alpha = 0.12$; $(R-1)(0.12) = 99 \times 0.12 = 11.88$; denominator = $1 + 0.02 + 11.88 = 12.90$; $S_{100} = 100/12.90 = 7.7519\times$.

Cross-check against the per-run ceiling: $S_{\text{run}} = 1/0.12 = 8.3333$. At $R = 100$ we achieve $7.7519/8.3333 = 93.02\%$ of the ceiling; the one-time materialization cost amortizes quickly.

Smaller workload, same parameters, $R = 5$: denominator = $1.02 + 4 \times 0.12 = 1.50$; $S_5 = 5/1.50 = 3.3333\times$. A 5-task sweep with 90% prefix sharing already beats the reported low end ($2.95\times$), while a 100-task sweep approaches $8.33\times$. The reported $32.84\times$ cannot be explained by $R = 100$ amortization at $f = 0.90$; it requires f near 0.98 (Section 4.1) or larger effective prefix groups.

Sensitivity to α at $R = 100$, $f = 0.90$:

- $\alpha = 0.10$: $1-f+\alpha = 0.20$; denominator = $1.10 + 99 \times 0.20 = 20.90$; $S = 100/20.90 = 4.7847\times$.
- $\alpha = 0.30$: $1-f+\alpha = 0.40$; denominator = $1.30 + 39.60 = 40.90$; $S = 100/40.90 = 2.4449\times$ — below the reported low end. At 30% materialization overhead even 90% prefix sharing fails to reach $2.95\times$ on a 100-task workload.

This is the quantitative core of QuWARP's abstention logic: the planner must refuse reuse whenever $f < \alpha$ (Eq. 3), and near the boundary the gains evaporate.

4.3 Break-even and abstention boundary

From Eq. 3, reuse is profitable iff $f \geq \alpha$. Concretely: if materializing and reading a boundary artifact costs 20% of a fresh run ($\alpha = 0.20$), a workload must share at least 20% of its execution cost to break even — and to hit even $2\times$, from Eq. 1: $1-f+\alpha = 0.5 \implies f = 1 - 0.5 + 0.20 = 0.70$. Going from break-even to $2\times$ requires moving from 20% to 70% prefix sharing — the profitability region is steep near the boundary, justifying an explicit abstention mechanism rather than always-reuse.

4.4 Legality guardrail as a correctness constraint

The continuation-legality check is a guardrail, not a performance feature, but it has a measurable cost dimension: every refused reuse is a fresh run (speedup 1.0 on that task). If a fraction q of candidate resumptions fails legality, the effective shared fraction becomes $f_{\text{eff}} = (1-q)f$, and Eq. 1 applies with f_{eff} . Worked example: $f = 0.90$, $\alpha = 0.02$, $q = 0.30$:

- $f_{\text{eff}} = 0.70 \times 0.90 = 0.63$.
- $1-f_{\text{eff}}+\alpha = 0.39$; $S_{\text{run}} = 1/0.39 = \mathbf{2.5641\times}$.

Compare $q = 0$: $S = 8.3333\times$. A 30% legality-failure rate cuts the per-run speedup by a factor of $8.3333/2.5641 = 3.25$. This quantifies why the narrowness of the legality guardrail is a first-order performance parameter, not merely a safety nicety.

4.5 Memory footprint of materialized artifacts

An n -qubit statevector has 2^n complex amplitudes; using double-precision complex128 (16 bytes per amplitude):

- $n = 30$: $2^{30} = 1,073,741,824$ amplitudes; $\times 16 \text{ B} = 17,179,869,184 \text{ B} = \mathbf{16 \text{ GiB}}$.
- $n = 34$: $2^{34} \times 16 = 274,877,906,944 \text{ B} \approx \mathbf{256 \text{ GiB}}$.
- $n = 36$: $2^{36} \times 16 = 1,099,511,627,776 \text{ B} = \mathbf{1 \text{ TiB}}$.
- $n = 40$: $2^{40} \times 16 = 17,592,186,044,416 \text{ B} = \mathbf{16 \text{ TiB}}$.

Materialized statevector artifacts are feasible in RAM up to roughly $n \approx 32\text{--}34$ on a large workstation and require distributed or NVMe storage beyond. This bounds the artifact-materialization strategy: it is a mid-sized-workload tool, not a frontier- n tool. Stabilizer-hybrid artifacts at Clifford/non-Clifford interfaces can be far smaller (Section 3.3).

4.6 Consistency check of the reported range

The ratio of the reported extremes is $32.84/2.95 = 11.1322$. Under our model, per-run speedup ratios between two workloads with the same α are $(1-f_2+\alpha)/(1-f_1+\alpha)$. With $\alpha = 0.01$ and $f_1 = 0.671$ (the $2.95\times$ case), matching $32.84\times$ requires $f_2 = 0.979549$ (Section 4.1). The difference $f_2 - f_1 = 0.30853$ — about 31 percentage points of additional shared-prefix fraction — separates the two ends of the reported range. This is a coherent spread across workload classes (VQE sweeps vs. QEC cycles) rather than an anomaly, which supports the plausibility of the reported range *as a range*, while flagging that the top end is achievable only under near-total sharing and sub-3% overhead.

5. Results

All numbers in this section are computed in Section 4 from the stated inputs; none are new measurements. Reported speedups ($2.95\times$, $32.84\times$) are quoted from [1][2]; everything else is model-derived.

R1. Implied shared-prefix fractions (Eq. 1, α assumed).

- $\alpha = 0.05$: $S = 2.95 \implies f = 0.711017$; $S = 32.84 \implies f = 1.019549$, **infeasible**.
- $\alpha = 0.01$: $S = 2.95 \implies f = 0.671017$; $S = 32.84 \implies f = 0.979549$.

R2. Hard overhead constraint at the top end. The $32.84\times$ result requires $\alpha \leq 1/32.84 = 0.0304507$ ($\approx 3.05\%$ of a fresh run) for any feasible $f \leq 1$. Any workload achieving $>20\times$ under this model has $\alpha < 1/20 = 0.05$ by the same derivation.

R3. Amortization (Eq. 2; $f = 0.90$, $\alpha = 0.02$ assumed). $R = 5$: $S = 3.3333\times$, $R = 100$: $S = 7.7519\times$. Ceiling $S_{\text{run}} = 8.3333\times$; $R = 100$ achieves 93.02% of the ceiling.

R4. Overhead sensitivity ($R = 100$, $f = 0.90$). $\alpha = 0.10 \implies S = 4.7847\times$. $\alpha = 0.30 \implies S = 2.4449\times$, below the reported low end of $2.95\times$.

R5. Break-even and steepness (Eq. 3, Eq. 1). Reuse is profitable iff $f \geq \alpha$. At $\alpha = 0.20$, break-even is $f = 0.20$ and $2\times$ requires $f = 0.70$.

R6. Legality-refusal cost (Section 4.4; $f = 0.90$, $\alpha = 0.02$). A 30% legality-refusal rate reduces effective sharing to $f_{\text{eff}} = 0.63$ and per-run speedup from $8.3333\times$ to $2.5641\times$ — a $3.25\times$ degradation attributable purely to guardrail strictness.

R7. Memory bounds (Section 4.5). Statevector artifacts require 16 GiB at 30 qubits, 256 GiB at 34 qubits, 1 TiB at 36 qubits, 16 TiB at 40 qubits; practical RAM-resident materialization caps out near 32–34 qubits on a single large node.

R8. Range coherence. The reported extremes differ by a factor of 11.1322 in speedup, which under the model corresponds to ≈ 0.30853 (≈ 31 percentage points) of additional shared-prefix fraction at $\alpha = 0.01$ — a plausible spread across workload classes, contingent on the top-end workload having near-total sharing and sub-3% materialization overhead.

Projection (labeled as such). If a future QEC-style workload exhibits $f = 0.98$ and stabilizer-hybrid materialization at $\alpha = 0.005$, Eq. 1 projects $S_{\text{run}} = 1/(1-0.98+0.005) = 1/0.025 = 40\times$, with uncertainty dominated by the assumed f and α ; if either parameter is off by 0.01 ($f = 0.97$ or $\alpha = 0.015$), the projection drops to $1/0.035 = 28.57\times$ — roughly $\pm 30\%$ sensitivity to one-point parameter errors. This projection is offered as a testable prediction of the model, not a result.

6. Discussion

Limitations of this analysis. Our model is a two-parameter abstraction. Real workloads have heterogeneous task costs, multiple prefix groups with different f and α , and planner overhead (prefix identification, legality checking, trace emission) that we folded into α but which may scale superlinearly with workload size. The reported speedups in [1][2] are against a specific Qrack-based per-task denominator; a different baseline simulator changes every derived number in Section 4. Most importantly, we had access only to the abstract of [1][2]; workload details, mode splits, and measurement methodology are unknown to us, so our "implied f and α " are inferences under stated assumptions, not recovered parameters.

Failure modes. (1) *Prefix drift*: in VQE-style loops, structure-optimization methods [3] can change the ansatz itself between iterations, splitting prefixes; TETRIS-Q-style noise-aware mapping [10] can insert different SWAPs per run for the same logical circuit, destroying syntactic prefix sharing unless the planner's equivalence notion is robust to commutation-based reordering. (2) *Legality gaps*: stabilizer-hybrid boundaries are only legal when the post-boundary computation consumes exactly what the artifact preserves; adaptive circuits (measurement-dependent branching, decoder feedback in QEC) can make reuse silently wrong if the guardrail is incomplete — the cost of a

missed illegality is not slowdown but wrong answers. (3) *Audit overhead*: EXPLAIN-style traces are I/O; at high task counts, trace volume could dominate α , and our $\alpha = 0.30$ sensitivity case (R4: 2.4449 $\times$) shows how quickly large overheads erase the gains. (4) *Memory blowout*: at ≥ 36 qubits, statevector materialization hits the TiB scale (R7), and the planner must abstain or spill to storage, where restore costs grow sharply. (5) *Selection bias*: the headline interval is conditioned on "reuse-positive workloads"; on reuse-negative workloads the speedup is, by construction, ≤ 1 minus planning overhead, and the abstract does not quantify this population, so the workload-level average gain is unknown.

What would falsify the claims. (a) A replication showing measured f values below ~ 0.6 on the evaluated workloads while speedups above $3\times$ persist — this would imply the gains come from something other than prefix reuse (e.g., kernel warm-up), contradicting the mechanism. (b) Demonstrated legality violations: any workload where a reused artifact produces amplitudes differing from direct execution beyond floating-point tolerance. (c) Independent measurement of the 32.84 $\times$ workload's materialization cost finding $\alpha > 0.0305$ with $f \leq 1$ — this would refute R2 and possibly the reported speedup's attribution to reuse. (d) Energy accounting under a JPCUB-style joules-per-solution protocol [11][12] showing that artifact materialization and DRAM residency consume more energy than the redundant execution saves.

Arguing against ourselves. A skeptic could say the model is unfalsifiable as stated: any observed speedup maps to some (f, α) , so the model "explains" everything. The reply is that the model makes *joint* constraints — R2's $\alpha \leq 0.0305$ for 32.84 $\times$ is a hard, falsifiable requirement. Second, a skeptic could note that near-total prefix sharing ($f \approx 0.98$) makes the "planning" trivial: our own model reproduces the upper bound at $f \approx 0.98$, meaning checkpoint reuse is unsurprising in principle (it is an old idea in general HPC). QuWARP's genuine contribution must therefore rest on the *decision machinery* — legality typing, abstention, and auditability — rather than on raw speedup. If the decision machinery is the contribution, the right evaluation is not speedup range but decision quality: false-reuse rate, false-abstention rate, and audit-trace completeness, none of which are quantified in the available abstract [1][2]. A further caveat: our bibliography includes one withdrawn arXiv entry [5], a reminder that automated literature feeds contain unreliable items; and the QNFO corpus items [10–13] are Zenodo-deposited, non-peer-reviewed documents whose claims we used only for framing, not as evidence.

Open questions. Does the planner compose with GPU-accelerated kernels [7], or does artifact transfer dominate? Can the model be extended to prefix trees with provably optimal materialization sets? What is the correct legality type system for mixed statevector/stabilizer workloads, drawing on the simulability notions of [8]? Can reuse gains be expressed as energy savings under a standardized protocol [11][12]? And how should the planner dynamically adjust reuse decisions based on real-time profiling (e.g., observed memory pressure) rather than static heuristics?

7. Conclusion

We analytically triaged QuWARP's workload-level reuse planning claim. A closed-form amortized-cost model shows the reported 2.95 $\times$ –32.84 $\times$ speedups are consistent with shared-prefix cost fractions of ~ 0.66 – 0.98 at 100-task scale, that the top end requires materialization overhead below $\sim 3.05\%$ of a fresh run, that reuse breaks even whenever $f \geq \alpha$ with steep profitability near that boundary, and that statevector artifact materialization is memory-bounded near 32–34 qubits on single nodes. The claims are plausible but single-sourced and conditioned on reuse-positive workloads; the durable contribution, if any, lies in the auditable legality-guarding machinery rather than the headline speedups. Independent replication with reported f values, decision-quality metrics, and energy accounting is the necessary next step.

References

- [1] TITLE: arXiv Query: `searchquery=&idlist=2609.23664&start=0&max_results=1` [2] QuWARP: A Workload-Aware Reuse Planner for simulating Quantum Circuits. arXiv:2609.23664v1. <https://arxiv.org/abs/2609.23664v1> [3] Structure optimization for parameterized quantum circuits. arXiv:1905.09692v3. <https://arxiv.org/abs/1905.09692v3> [4] Proof of monotonic increase in the cost function for Krotov algorithm for open quantum systems. arXiv:2006.16817v2. <https://arxiv.org/abs/2006.16817v2> [5] A Simulation and Modeling of Access Points with Definition Language. arXiv:1304.1836v2. <https://arxiv.org/abs/1304.1836v2> [6] Multi-strategy Based Quantum Cost Reduction of Quantum Boolean Circuits. arXiv:2407.04826v1. <https://arxiv.org/abs/2407.04826v1> [7] Fast quantum circuit simulation using hardware accelerated general purpose libraries. arXiv:2106.13995v1. <https://arxiv.org/abs/2106.13995v1> [8] From estimation of quantum probabilities to simulation of quantum circuits. arXiv:1712.02806v3. <https://arxiv.org/abs/1712.02806v3> [9] Introduction to Quantum Electromagnetic Circuits. arXiv:1610.03438v2. <https://arxiv.org/abs/1610.03438v2> [10] DOI 10.5281/zenodo.22739633. QNFO: TETRIS-Q: Tiling-based Effective Transient-fault Reduction and Parallelism Optimizations for Quantum Circuit Mapping. [11] DOI 10.5281/zenodo.21978952. QNFO: QWAV Strategy v2.4.1: The Energy-Standard Playbook — Consortium Governance, Verified Precedents, and the JPCUB Road to a Quantum Computing Energy Benchmark. [12] DOI 10.5281/zenodo.21821767. QNFO: JPCUB Competitive Landscape v2.0: System-Level Joules-per-Solution Estimates for 17 Quantum Computing Platforms from Published Specifications. [13] DOI 10.5281/zenodo.21515894. QNFO: Due Diligence Report: QuiX Quantum.

Appendix A. Divergence report

D1. Analytical methodology (A vs. B/C). Draft A applies the reported speedup interval directly as a scalar multiplier on an assumed workload (20 tasks $\times$ 10 h baseline, mean speedup $\bar{S} = (2.95+32.84)/2 = 17.895$, yielding $R_1 = 200/17.895 \approx 11.18$ h). Drafts B and C instead invert a cost model to recover the workload parameters (f , α) that the reported interval *implies*. **Convention chosen:** the B/C model-inversion approach, because averaging the endpoints of a speedup range assumes a uniform distribution of speedups across workloads with no evidentiary basis, whereas inversion exposes the parameter requirements the claims must satisfy and yields falsifiable constraints (e.g., $\alpha \leq 0.0305$ at $32.84\times$). Draft A's scalar-application results are therefore not reproduced in the main text.

D2. Memory-footprint claims (A vs. B). Draft A claims a per-task statevector memory of 16 MiB (20 qubits $\times$ 16 B/amplitude) shrinking to ≈ 1.24 MiB after five reuse events via a 40%-per-reuse reduction factor. Draft B computes absolute artifact sizes of 16 GiB (30 qubits) to 16 TiB (40 qubits) with no per-reuse shrinkage model. **Convention chosen:** Draft B's absolute sizing, because it is directly computable from the standard complex128 representation and requires no unsupported empirical "40% per reuse" factor; Draft A's cascaded-shrinkage model ($r = 0.60$ applied five times) is a single-source assumption with no derivation in [1][2] and is rejected from the main text. Note the two drafts are not arithmetically contradictory (20 vs. 30–40 qubits), but A's *reduction dynamics* are unsubstantiated.

D3. Planner overhead treatment (A vs. C). Draft A adopts a flat 5% planner overhead on total baseline runtime (10 h on a 200 h workload), citing the original implementation's trace-generation cost. Draft C treats overhead as a per-resumed-task fraction α and derives that 5% is *infeasible* for the $32.84\times$ result ($\alpha \leq 0.0305$ required). **Convention chosen:** Draft C's per-task α parameterization, which is strictly more constraining and subsumes A's flat figure as a special case when the per-task materialization overhead equals the flat fraction of total baseline

runtime (i.e., when every task incurs the same 5% overhead, $\alpha = 0.05$ uniformly across tasks). The per-task formulation is preferred because it exposes the infeasibility of that flat figure at the top end of the reported speedup range ($\alpha \leq 0.0305$ for $32.84\times$), whereas the flat treatment conceals it.