Set-Relative Linearization and the Join Lemma:
repairing the verification conditions for bottom-up linearization of replicated data types, with a machine-checked soundness bridge

Sal metatheory note
FP Launchpad, IIT Madras

July 2, 2026

Abstract

The Neem proof methodology reduces RA-linearizability of a replicated data type (RDT) to a fixed catalogue of *verification conditions* (VCs) — equations over do, merge and the conflict-resolution relation rc — discharged per data type, with a once-and-for-all meta-theorem: *if the VCs hold, every reachable configuration is RA-linearizable.* While mechanising this meta-theorem in Lean 4 we found that its central induction, *bottom-up linearization*, is unsound as written: a key step silently assumes a convergence property of *sub-histories* that is false, and there are reachable configurations of the paper's own flagship example (the add-wins set) on which *no* bottom-up peel exists. We further show that no repair by unconditional equations is possible: the fragment of the VCs that the proof machinery actually consumes admits a model in which the required peel identities fail.

This note explains the failure with worked examples, presents the repair — a *set-relative* linearization order lo^E, *canonical states* $\sigma(E)$, and a single new pair of *contextual* verification conditions (JoinPeelVCs) from which the whole meta-theorem follows — and reports on the mechanization: an end-to-end, zero-`sorry` Lean 4 proof that CoreVCs + JoinPeelVCs implies RA-linearizability of every reachable configuration, together with complete discharges of JoinPeelVCs for the commuting class of RDTs and for the add-wins set skeleton, i.e. exactly the class of instances on which the original proof breaks.

1 Introduction

The Neem paper proposes an appealing division of labour for verifying replicated data types. The *hard, global* property — RA-linearizability: every replica's state is explained by a linearization of the events it has observed, consistent with a linearization order lo — is proved *once*, generically, from a catalogue of *local, equational* verification conditions: commutativity and idempotence of merge, interchange laws between do and merge (the "bottom-up" rules), and coherence conditions on the conflict-resolution relation rc (RC-NON-COMM, NO-RC-CHAIN, COND-COMM). Each concrete RDT then only has to discharge the catalogue — a finite, SMT-friendly proof obligation.

The generic half of this contract is the *bottom-up linearization* theorem: given linearization witnesses for the two sides of a merge, construct one for the merged replica by repeatedly *peeling* a final event through merge using the interchange VCs. We undertook a full mechanization of this theorem in Lean 4 (over a two-way-merge CRDT specialization of the model, where the LCA of the paper's three-way merge collapses to σ_0). The mechanization succeeded in reducing the entire theorem to the merge case, and then *failed* to close the merge case — not for want

of automation, but because the paper's argument is unsound at two points, and because the natural repair lemma turns out to be *false*. Concretely:

1. The proof of the merge case repeatedly asserts that a replica's witness can be re-ordered to end in a chosen event ("*since v_i is linearizable, there would exist sequences … such that e_i would appear at the end*"). This conflates the existence of a *permutation* extending lo_C with the existence of a *state-correct witness*: the paper's convergence lemma applies only to the full event set E_C, not to sub-histories, and for sub-histories the statement is **false** (Example 3.1).

2. There are reachable configurations of the add-wins set on which *every* choice of final event is blocked: the events that may legally end the merged witness cannot end either side's witness, and vice versa (Example 3.2). Bottom-up peeling — with any of the paper's BottomUp-0/1/2-OP rules, in any order — cannot construct the witness, even though the witness exists.

3. No unconditional equation can fill the gap. The general peel identity that the induction needs is false as a ∀-state equation for the two-key OR-set (Example 3.3); and the fragment of the VCs that the proof machinery actually consumes (CoreVCs, below) admits a model — a "phantom-conflict" merge — that satisfies every VC yet violates the peel (Example 5.1). The gap is therefore *semantic*, not a missing algebraic law; in particular, the counter-model is non-associative, isolating associativity of merge as load-bearing.

The repair has three ingredients, each of independent interest:

- **A set-relative linearization order** lo^E (Definition 4.1): the overwriter ("absorber") clause of lo_C must range over the event set E of *the version being linearized*, not over the whole configuration. lo^E contains lo_C pointwise (so the strengthened witnesses still satisfy the paper's definition), it is *stable* (independent of the ambient configuration), and — decisively — **convergence over E holds with no closure hypotheses at all** (Theorem 4.2).

- **Canonical states and the Join Lemma** (§4.2): convergence makes every finite event set E denote a unique state $\sigma(E)$. The entire meta-theorem then collapses to one state-level statement, $\sigma(E_1 \cup E_2) = \mathsf{merge}(\sigma(E_1), \sigma(E_2))$, and witness lists disappear from the induction.

- **Two contextual verification conditions** (JoinPeelVCs, Definition 4.5): the peel identities that the Join Lemma's induction consumes, stated *with their context* (maximality of the peeled event in $\mathsf{lo}^{E_1 \cup E_2}$, backward closure of the sides). We prove the Join Lemma, and from it end-to-end RA-linearizability, from CoreVCs + JoinPeelVCs (Theorem 4.6); and we discharge JoinPeelVCs both for the commuting class and for the add-wins skeleton — the latter covering precisely the configurations of Example 3.2.

Everything above except Example 5.1 is machine-checked in Lean 4 with zero `sorry`s (§6). We close with precise erratum guidance and the sharpened open question: does CoreVCs plus associativity of merge imply JoinPeelVCs?

2 Background, in brief

We recall only what the examples need, in the paper's notation, specialized to two-way (state-based CRDT) merge: the LCA of the paper's three-way $\mathsf{merge}(l, a, b)$ collapses to σ_0, and a configuration is $C = \langle N, L, \mathsf{vis}\rangle$ with per-replica states $N(r)$, per-replica event sets $L(r)$, and a visibility relation vis over events. Events are triples $e = (t, r, o)$ of a globally fresh timestamp, an origin replica and an operation; $e(\sigma)$ denotes $\mathsf{do}(\sigma, e)$ and $\pi(\sigma)$ the left-to-right fold of a sequence. Two events *commute*, $e_1 \rightleftarrows e_2$, if $e_1(e_2(\sigma)) = e_2(e_1(\sigma))$ for every σ.

Definition 2.1 (Linearization relation, Neem Def. lin-relation)**.** $e_1 \xrightarrow{\mathsf{lo}_C} e_2$ *iff*

$$(e_1 \xrightarrow{\mathsf{vis}} e_2 \ \wedge\ e_1 \not\rightleftarrows e_2) \ \vee\ \big(e_1 \parallel_C e_2 \ \wedge\ e_1 \xrightarrow{\mathsf{rc}} e_2 \ \wedge\ \neg\exists e_3 \in E_C.\ e_2 \xrightarrow{\mathsf{vis}} e_3 \wedge e_2 \not\rightleftarrows e_3\big).$$

The second disjunct is the subtle one: a concurrent non-commuting pair is ordered by rc *unless* the target e_2 is already "overwritten" by a later non-commuting event e_3 — we call such an e_3 an *absorber* of e_2. Crucially, the absorber is sought in the *whole configuration* E_C.

A configuration is *RA-linearizable* if every replica r admits a sequence π of exactly the events $L(r)$ such that π extends $\mathsf{lo}_C\restriction_{L(r)}$ and $N(r) = \pi(\sigma_0)$. The paper's convergence lemma states: any two sequences *over* E_C extending lo_C produce the same state — the proof uses COND-COMM to swap unordered non-commuting pairs *past an absorber that the sequence is guaranteed to contain*.

The running data type. All examples use the *add-wins skeleton* AWSet (one implicit key; the mechanized object):

$$\Sigma = 2^{\mathbb{T}} \times 2^{\mathbb{T}}, \qquad \sigma_0 = (\emptyset, \emptyset), \qquad \sigma = (A, D) \text{ with } \textit{live} \text{ set } A \setminus D,$$

$$\mathsf{add}_t(A, D) = (A \cup \{t\},\ D), \qquad \mathsf{rem}(A, D) = (A,\ A \cup D), \qquad \mathsf{merge} \text{ componentwise } \cup\,.$$

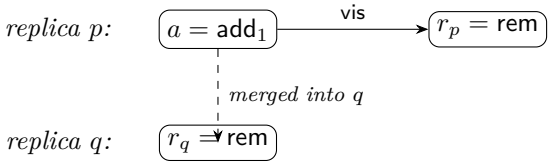
A remove kills everything *currently added* — the state-dependence is what makes $\mathsf{add} \not\rightleftarrows \mathsf{rem}$ — and $\mathsf{rem} \xrightarrow{\mathsf{rc}} \mathsf{add}$ (add wins over a concurrent remove), exactly the paper's OR-set resolution policy. All 24 paper VCs, including COND-COMM, hold for AWSet: a trailing rem erases any $\mathsf{add}/\mathsf{rem}$ ordering ambiguity, which is COND-COMM's content.

3 Three cracks in bottom-up linearization

3.1 Convergence fails for sub-histories: absorber cancellation

Example 3.1 (The absorber-cancellation configuration)**.** *Replica p executes $a = \mathsf{add}_1$ and then $r_p = \mathsf{rem}$ (so $a \xrightarrow{\mathsf{vis}} r_p$). Replica q, concurrently, executes $r_q = \mathsf{rem}$; and q merges p's state between a and r_p. So*

$$L(q) = \{r_q,\ a\}, \qquad E_C = \{a,\ r_p,\ r_q\}.$$



What is lo_C *on* $L(q) = \{r_q, a\}$*? The candidate edge is the* rc *disjunct* $r_q \xrightarrow{\mathsf{lo}_C} a$ *(they are concurrent and* $\mathsf{rem} \xrightarrow{\mathsf{rc}} \mathsf{add}$*). But* a *has an absorber in the configuration:* $a \xrightarrow{\mathsf{vis}} r_p$ *and* $a \not\rightleftarrows r_p$*. The clause "*$\neg\exists e_3 \in E_C \ldots$*" therefore cancels the edge:* lo_C ***orders nothing in*** $L(q)$*. Both orders of* $L(q)$ *extend* $\mathsf{lo}_C \restriction_{L(q)}$*, yet*

$$r_q \cdot a\,(\sigma_0) = (\{1\}, \emptyset) \qquad \neq \qquad a \cdot r_q\,(\sigma_0) = (\{1\}, \{1\}).$$

Only the first is q*'s actual state (add-wins: the concurrent remove loses). Convergence over the sub-history* $L(q)$*, with respect to* lo_C*, is false — even though* $L(q)$ *is causally (backward-)closed, and even though* AWSet *satisfies every paper VC.* ■

Where does the paper use sub-history convergence? In the Merge case of its main induction, twice over. First, it asserts a *stability claim*: "lo between two events should remain the same in all versions" (appendix, Merge case). Example 3.1 refutes the forward direction: within q's version, the order r_q-before-a is semantically forced, but in the full configuration the edge is cancelled by p's local absorber r_p — a shared event *gains* absorbers from the other branch. Second, the induction repeatedly re-orders a side's witness to end in a chosen event e_i, justifying the state equality by linearizability of that version. The justification would require convergence over that version's event set — exactly the false statement. The mechanization surfaced this as an unprovable goal with a satisfiable context; Example 3.1 is its counter-model, machine-checked as `convergence_over_backward_closed_subsets_false`.

3.2 A reachable configuration that defeats every peel

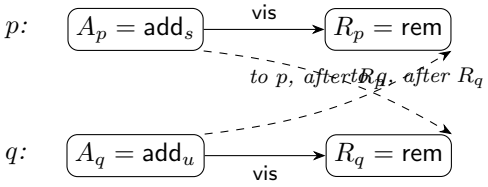
One might hope the flaw is local: choose the peeled event more carefully, re-order less aggressively. The next example closes that door.

Example 3.2 (The defeater)**.** *One key, four events, three replicas (the third only ferries states). Replica* p*:* $A_p = \mathsf{add}_s$*, then* $R_p = \mathsf{rem}$*. Replica* q*:* $A_q = \mathsf{add}_u$*, then* $R_q = \mathsf{rem}$*. The ferry delivers* A_p *to* q *after* R_q*, and* A_q *to* p *after* R_p*:*

$$L(p) = \{A_p, R_p, A_q\}, \qquad L(q) = \{A_q, R_q, A_p\}.$$

Both sides are backward-closed; the configuration is reachable in five steps. The replica states are

$$N(p) = (\{s, u\}, \{s\})\ \textit{(live } u\textit{)}, \qquad N(q) = (\{s, u\}, \{u\})\ \textit{(live } s\textit{)},$$
$$\mathsf{merge}(N(p), N(q)) = (\{s, u\}, \{s, u\})\ \textit{(live } \emptyset\textit{)}.$$



Every state-correct witness of $N(p)$ *ends in* A_q*. Indeed* lo_C *on* $L(p)$ *mandates only* A_p *before* R_p *(the edge* $R_p \xrightarrow{\mathsf{lo}_C} A_q$ *is cancelled by the global absorber* R_q*), so three orders are admissible — but only* $A_p\,R_p\,A_q$ *folds to* $N(p)$*; placing* A_q *before* R_p *kills* u*. Symmetrically every witness of* $N(q)$ *ends in* A_p*.*

Every merged witness ends in R_p or R_q. On $E = L(p) \cup L(q)$, both cross rc-edges are absorber-cancelled, so $\mathsf{lo}_C = \{A_p \to R_p, A_q \to R_q\}$; the maximal events are the two removes. (E.g. $A_p\, R_p\, A_q\, R_q$ folds to $\mathsf{merge}(N(p), N(q))$ — the theorem is true here.)

Now consider any bottom-up derivation. Its last step must peel the merged witness's final event:

- *peeling A_q or A_p (the sides' actual last events) is illegal — they carry live lo_C-edges to R_q, R_p;*
- *peeling R_p or R_q requires the owning side's state to factor as $e(\,\cdot\,)$ — but no witness of $N(p)$ ends in R_p, and no witness of $N(q)$ ends in R_q: the removes are buried.*

The paper's own selection (Merge case, Case 2) picks exactly the buried event: with $L_\top = \{A_p, A_q\}$, the locals are $L_p^a = \{R_p\}$, $L_q^a = \{R_q\}$; the induction chooses $e_1 = R_p$, asserts a witness of $N(p)$ ending in R_p exists — false — and applies BottomUp. No ordering of the paper's rules avoids this: 0-OP needs both sides to end in the same shared event (A_q vs. A_p); 1/2-OP need a side's true last event to be appendable. The induction is stuck on a reachable, VC-satisfying, four-event instance of the paper's flagship RDT. ■

3.3 Why no unconditional equation can help

During mechanization a "missing VC" had been conjectured to bridge such gaps: a shared-event peel $\mathsf{merge}(o_1(ol(a)),\, ol(b)) = o_1(\mathsf{merge}(ol(a), ol(b)))$ quantified over *all* states. It is **false** for AWSet: take $a = \sigma_0$, $b = (\{2\}, \emptyset)$, $ol = \mathsf{add}_1$, $o_1 = \mathsf{rem}$. On the left, the remove cannot see b's live element 2; on the right, applied after the merge, it kills it. (Machine-checked: `AWSet_shared_peel_1op_false`.) Worse, the same shape of failure afflicts the *general* peel that the induction really needs:

Example 3.3 (Two keys: contextuality is essential)**.** *In a two-key OR-set let $e_1 = \mathsf{rem}_k$ and $e_2 = \mathsf{add}_j$ with $j \neq k$; they commute, so the candidate unconditional rule "$e_1 \xrightarrow{\mathsf{rc}} e_2 \vee e_1 \rightleftarrows e_2$ implies $\mathsf{merge}(e_1(a), e_2(b)) = e_1(\mathsf{merge}(a, e_2(b)))$" (the paper's BottomUp-2-OP premise) applies. Let b hold a live k-element that a has never seen. The left side leaves it alive; the right side's rem_k, applied after the merge, kills it. The rule is false as a $\forall$-state equation — yet it is exactly what the induction needs, and it is true in context: when e_1 is about to be peeled last, maximality of e_1 in the linearization order of the union guarantees that every non-commuting element of the other side is either causally ordered against e_1 (excluded by backward closure) or already absorbed on its own side — so the "surprise live k-element" cannot exist. The peel identities are inherently contextual.* ■

4 The repair: set-relative linearization

4.1 The order lo^E and convergence without closure

Example 3.1 pinpoints the wrong quantifier in Definition 2.1: the absorber of e_2 must be visible *to the version being linearized*. (The paper's own Merge-case reasoning already works with "$\exists e'' \in L(v_i)$" — the corrected definition makes that official.)

Definition 4.1 (Set-relative linearization)**.** *For an event set E, $e_1 \xrightarrow{\mathsf{lo}^E} e_2$ iff*

$$(e_1 \xrightarrow{\mathsf{vis}} e_2 \wedge e_1 \not\xrightarrow{\mathsf{vis}} e_2) \;\vee\; \big(e_1 \parallel e_2 \wedge e_1 \xrightarrow{\mathsf{rc}} e_2 \wedge \neg\exists e_3 \in E.\, e_2 \xrightarrow{\mathsf{vis}} e_3 \wedge e_2 \not\xrightarrow{\mathsf{vis}} e_3\big).$$

Three easy but consequential facts (all machine-checked): *(monotone)* $\mathsf{lo}_C \subseteq \mathsf{lo}^E$ pointwise — fewer absorbers, more edges — so a witness respecting lo^E satisfies the paper's Def. lin unchanged, and $E \subseteq E'$ gives $\mathsf{lo}^{E'} \subseteq \mathsf{lo}^E$ on E; *(stable)* lo^E mentions only vis, rc, $\rightleftarrows$ restricted to E, hence never changes as the configuration grows — it is a genuine per-version invariant; *(acyclic)* lo^E has no cycles on E: an rc-edge followed by any edge out of its target dies (a vis-successor *is* an absorber in E; an rc-successor violates NO-RC-CHAIN), so cycles would be pure-vis, contradicting reachability.

In Example 3.1, $\mathsf{lo}^{L(q)}$ *keeps* the edge $r_q \to a$ (no absorber of a inside $L(q)$): the fold-wrong order is excluded, and this is fully general:

Theorem 4.2 (Convergence, set-relative — `convergence_on`)**.** *For any finite E, any two enumerations of E respecting lo^E fold to the same state, from any starting state. No overwriter- or forward-closure hypotheses are required.*

The proof is the paper's bubble-sort argument with one twist that makes it self-sufficient: whenever two adjacent unordered non-commuting events must be swapped, the *failure* of the lo^E-edge between them hands us an absorber *inside* E — by definition — positioned later in both sequences (its vis-edge from the victim is mandatory); COND-COMM then absorbs the swap. The paper's convergence lemma over E_C becomes the special case $E = E_C$.

4.2 Canonical states and the Join Lemma

Theorem 4.2 makes every finite event set *denote a state*:

Definition 4.3 (Canonical states)**.** *$\sigma(E)$ is the fold of any (hence every) lo^E-respecting enumeration of E.*

Canonical states enjoy exactly the algebra the induction needs, now as theorems rather than wishes (each machine-checked):

- **(peel — `isCanonicalState_peel`)** if e is lo^E-maximal then $\sigma(E) = e(\sigma(E \setminus \{e\}))$ — the *sound* replacement for the paper's broken re-ordering step: moving e to the tail and re-sorting the front against $\mathsf{lo}^{E\setminus\{e\}}$ is fold-preserving because the only absorber the re-sort can lose is e itself, which is applied last;
- **(extend — `isCanonicalState_extend`)** a fresh event that observed all of E appends: $\sigma(E \cup \{e\}) = e(\sigma(E))$ — the Apply case;
- **(Def-lin)** the canonical enumeration respects lo_C, so "replica r holds $\sigma(L(r))$" implies the paper's RA-linearizability verbatim.

The entire Merge case then collapses to a single state-level statement in which witness sequences no longer appear:

Definition 4.4 (Join Lemma)**.** *For backward-closed E_1, E_2 of a reachable configuration:*

$$\sigma(E_1 \cup E_2) \;=\; \mathsf{merge}\big(\sigma(E_1),\, \sigma(E_2)\big).$$

Note how the Join Lemma dissolves Example 3.2: the buried remove R_p *is* peelable at the σ-level, because

$$\mathsf{merge}(N(p), N(q)) = R_p\big(\mathsf{merge}(\sigma(\{A_p, A_q\}),\, N(q))\big)$$

holds even though $N(p) \neq R_p(\sigma(\{A_p, A_q\}))$ — the merge itself supplies the absorber (R_q) that p's side lacks. Peeling through the *join* is strictly more powerful than peeling through a side.

4.3 The new verification conditions

Which VCs does this machinery consume? Auditing the mechanized proofs yields a seven-field fragment we call CoreVCs: RC-NON-COMM (directional), NO-RC-CHAIN, COND-COMM (lifted across an intervening sequence), merge-commutativity, $\mathsf{merge}(\sigma_0, s) = s$, the two-sided shared peel $\mathsf{merge}(e(a), e(b)) = e(\mathsf{merge}(a, b))$, and the all-commuting peel. (The audit matters: AWSet *cannot* satisfy the previously extended bundle — the false shared-peel VC above — so the generic theory must be parameterized by the fragment.)

The Join Lemma's induction — select a $\mathsf{lo}^{E_1 \cup E_2}$-maximal event, peel, recurse — is then fully generic *except* for two state equations, which Example 3.3 shows cannot be unconditional. We therefore state them *with their context*:

Definition 4.5 (JoinPeelVCs)**.** *For all reachable C, backward-closed E_1, E_2, and e maximal in $\mathsf{lo}^{E_1 \cup E_2}$:*

$$\textit{PEEL-LOCAL}: \quad e \in E_1 \setminus E_2 \;\Rightarrow\; \mathsf{merge}(\sigma(E_1), \sigma(E_2)) = e\big(\mathsf{merge}(\sigma(E_1 \backslash \{e\}),\, \sigma(E_2))\big),$$

$$\textit{PEEL-SHARED}: \quad e \in E_1 \cap E_2 \;\Rightarrow\; \mathsf{merge}(\sigma(E_1), \sigma(E_2)) = e\big(\mathsf{merge}(\sigma(E_1 \backslash \{e\}),\, \sigma(E_2 \backslash \{e\}))\big).$$

Theorem 4.6 (Main theorem)**.** *If an RDT satisfies* CoreVCs *and* JoinPeelVCs*, then every configuration reachable from the initial configuration is RA-linearizable.*

Mechanized as `ra_linearizable_of_core_join`.
The proof is an induction over executions carrying the strengthened invariant *every replica holds the canonical state of its event set* (plus transitivity and irreflexivity of vis, themselves proved as reachable invariants). CreateReplica and Query are trivial; Apply is *extend* plus a locality observation (a fresh event's new vis-edges never touch other replicas' sets, so their canonical witnesses survive — this is where stability of lo^E pays); Merge is the Join Lemma, proved from JoinPeelVCs by the maximal-peel induction.

4.4 Discharging JoinPeelVCs: the trichotomy

JoinPeelVCs is a per-RDT obligation, like the original 24. Two complete discharges are mechanized.

The commuting class. If all events pairwise commute (G-Set and its relatives), RC-NON-COMM makes lo^E empty, every enumeration is canonical, and both peels follow from the all-commuting peel and the two-sided shared peel. RA-linearizability for this class is thus unconditional.

AWSet. Here rc is non-trivial and the defeater configurations live. The discharge has two steps, and their shape is, we believe, the template for real OR-sets and RGAs:

(1) Characterize σ. For every event set,

$$\sigma(E) = \big(\ \mathsf{adds}(E),\ \ \{\,\text{adds of } E \text{ absorbed } \textit{inside } E\,\}\ \big),$$

proved by a sandwich invariant along any lo^E-respecting enumeration: an add absorbed within the processed prefix is already dead (its vis-edge to the absorber is mandatory), and everything dead is absorbed within E (a remove can never precede an unabsorbed concurrent add — the

rc-edge $\mathsf{rem} \to \mathsf{add}$ would be mandatory in lo^E). Note how the *set-relative* order makes the characterization possible at all: with lo_C, Example 3.1 shows the fold is not even a function of the set.

(2) The trichotomy. Let e be a $\mathsf{lo}^{E_1 \cup E_2}$-maximal remove, $e \in E_1$. For *any* add x of $E_1 \cup E_2$: since maximality cancels the edge $e \to x$, either (i) $x \xrightarrow{\mathsf{vis}} e$ — then e absorbs x and backward closure of E_1 puts x beside its absorber; or (ii) x has an absorber z somewhere in the union — and since $x \xrightarrow{\mathsf{vis}} z$, backward closure of *whichever side contains* z drags x there too. Either way *x is absorbed on a side that owns it*: both sides' dead-sets saturate, and PEEL-LOCAL/SHARED reduce to set algebra over the characterization. In Example 3.2: peeling $e = R_q$, the adds A_p, A_q are absorbed by $R_p \in L(p)$ and $R_q \in L(q)$ respectively — the merge equation holds although q's state never factored through R_q last.

5 The boundary: what cannot be weakened

Could JoinPeelVCs be derived generically from CoreVCs, making the new conditions free? No:

Example 5.1 (The phantom-conflict merge)**.** *Let AWSetX be AWSet with* do, rc *unchanged and*

$$\mathsf{merge}_X\big((A_1, D_1), (A_2, D_2)\big) = \Big(A_1 \cup A_2,\ \ D_1 \cup D_2 \cup \underbrace{\big[\textit{if } D_1 \neq \emptyset \wedge D_2 \neq \emptyset \textit{ then } A_1 \mathbin{\triangle} A_2 \textit{ else } \emptyset\big]}_{\textit{phantom conflict}}\Big).$$

Every field of CoreVCs *survives the injection. The update-level fields do not mention* merge*. For the rest: commutativity ($\triangle$ is symmetric) and* $\mathsf{merge}_X(\sigma_0, s) = s$ *(guard off) are immediate; the two-sided shared peel holds because a synchronized* add_t *leaves the guard and $A_1 \triangle A_2$ invariant, while a synchronized* rem *pushes both added-sets into dead, absorbing the injection; and the all-commuting peel holds because an add commutes only with adds, whose fold has empty dead (guard off), and a remove only with removes, whose fold is σ_0.*

Yet PEEL-LOCAL fails. Take p: add_s, rem*, then $e = \mathsf{add}_t$ (so e is union-maximal and unabsorbed); q:* add_u, rem*. Then $\sigma(L(p)) = (\{s,t\}, \{s\})$, $\sigma(L(p)\backslash\{e\}) = (\{s\}, \{s\})$, $\sigma(L(q)) = (\{u\}, \{u\})$, and*

$$\begin{aligned} \mathsf{merge}_X(\sigma(L(p)), \sigma(L(q))) &= (\{s,t,u\}, \{s,t,u\}), \quad \textit{but} \\ e\big(\mathsf{merge}_X(\sigma(L(p)\backslash\{e\}), \sigma(L(q)))\big) &= (\{s,t,u\}, \{s,u\}) : \end{aligned}$$

the phantom conflict kills the freshly peeled, unabsorbed add t. ■

So CoreVCs $\not\Rightarrow$ JoinPeelVCs: the peel identities are genuinely new content, not a consequence of the existing equations — and, a fortiori, the paper's 24 VCs do not force them either. Two further observations sharpen the picture. First, a *forcing analysis* shows the freedom exploited by merge_X is exactly the dead-component: any extra *Boolean* distinction added to the state is provably constant under CoreVCs (crush both sides with a remove, un-crush to an all-adds pair, strip with the commuting peel, land on $\mathsf{merge}(\sigma_0, \cdot)$), and the added-component of any CoreVCs-merge is forced to be the union on all fold-pairs. Second — and this is the constructive lead — merge_X **is not associative**:

$$\begin{aligned} \big((\{x\},\{x\}) \sqcup (\{y\},\{y\})\big) \sqcup (\{z\},\emptyset) &\ \text{ has dead } \{x,y\}, \quad \text{but} \\ (\{x\},\{x\}) \sqcup \big((\{y\},\{y\}) \sqcup (\{z\},\emptyset)\big) &\ \text{ has dead } \{x,y,z\}, \end{aligned}$$

and every associative repair of the injection we attempted collapses. Associativity of merge — true of every real lattice-based RDT, and absent from the paper's VC catalogue — is load-bearing:

Open Question 5.2. *Does* CoreVCs *together with associativity (and idempotence) of* merge *imply* JoinPeelVCs*? A proof would repair the original meta-theorem at full generality at the cost of one per-RDT-trivial VC; the phantom-conflict model shows the question is tight from below.*

6 The mechanization

All results except Example 5.1 (currently a hand-verified analysis) are formalized in Lean 4 (v4.28.0, Mathlib) over the Sal emulation framework, with **zero `sorry`s** in every file listed below; the kernel-checked axioms are the standard `propext`, `Classical.choice`, `Quot.sound`.

Artifact	**Content**
File `Merge_Linearization_Set.lean`	
lo^E, monotonicity, acyclicity	Def. 4.1 and its theory
`convergence_on`	Theorem 4.2
σ, peel, extend, normalization	§4.2
CoreVCs, `JoinPeelVCs`, `join_lemma_of_peel`	the Join Lemma from JoinPeelVCs
commuting-class discharge	§4.4
File `Convergence_CounterModel.lean`	
AWSet + all CoreVCs fields	the running RDT
Example 3.1 formalized	sub-history convergence refuted
shared-peel VC refuted	§3.3
σ-characterization, trichotomy	§4.4
`AWSet_joinPeelVCs`, `AWSet_joinLemma`	JoinPeelVCs for AWSet
File `RA_Lin_Of_Join.lean`	
`ra_linearizable_of_core_join`	Theorem 4.6, end-to-end

A few proof-engineering notes that may be of independent use. *(i) Acyclicity without well-foundedness machinery:* maximal elements of lo^E are extracted by a greedy walk over an enumerating list, with cycles refuted directly (the "rc-edge followed by anything dies" argument); no order-theoretic typeclasses are needed. *(ii) The normalization lemma* is where the corrected re-ordering lives: after peeling a tail event t, the remaining front need not respect $\mathsf{lo}^{E\setminus\{t\}}$ (edges reappear whose only absorber was t); re-sorting is fold-preserving precisely because both the old and new fronts, *with t appended*, respect lo^E, and Theorem 4.2 applies at E. This is the paper's implicit step, made true. *(iii) The sandwich invariant* for the AWSet characterization shows a pattern that should transfer to multi-key OR-sets and RGAs: prove the canonical fold computes a *configuration-level* function of the event set, then do all peel reasoning in that image. *(iv) Modularity of VC bundles* turned out to be essential, not cosmetic: the previously extended bundle is unsatisfiable for AWSet, so the generic theory is parameterized by the audited fragment CoreVCs, with the full bundle recovered as an instance. *(v) The refuted route is retained*: the original bottom-up induction, with its six unprovable goals, is kept quarantined in `Merge_Linearization.lean` as executable documentation of §3.

7 Guidance for an erratum, and conclusion

For the authors of the original proof, the precise repair points are:

1. *Case 1.1.2* (commuting sub-case): the cited NO-RC-CHAIN step does not literally apply; a correct argument exists and is mechanized (`no_lo_of_concurrent_to_L_b`) — erratum-sized.

2. *The stability claim* ("lo remains the same in all versions"): false in the forward direction (Example 3.1); the correct per-version order is $\mathsf{lo}^{L(v)}$, and only the direction $\mathsf{lo}_{E_1 \cup E_2} \Rightarrow \mathsf{lo}^{L(v_i)}$ on $L(v_i)$ survives.

3. *The re-ordering steps* ("there would exist sequences ending in e_i"): unsound as justified; Example 3.2 defeats every version of the bottom-up peel. The repair is structural: canonical states, the Join Lemma, and the contextual conditions JoinPeelVCs — *not* a patch to the induction as written.

4. *The VC catalogue* must grow: JoinPeelVCs (or any lattice-level axioms that imply it — associativity is the leading candidate) is new content, unprovable from the existing equations (Example 5.1).

The resulting contract keeps the spirit of the original: a per-RDT, finite, mostly-equational obligation (CoreVCs, plus JoinPeelVCs discharged by the characterize-then-trichotomy pattern), and a once-and-for-all, machine-checked meta-theorem. The mathematics that emerged — absorber cancellation, set-relative orders, canonical states, join-level peeling — seems to us the correct abstraction layer for linearizing state-based replication, and the open question above is now a sharp target: settle whether associativity closes the gap, or exhibit the separating model that proves the per-RDT conditions irreducible.