

Gap Analysis:

| Theme (What's the question ?) | What we know (strengths/innova- tions) | What's missing or limited | Why it matters | Concrete next steps | Open hypothesis to test |
|--|---|--|--|--|--|
| The exact boundary of PSI and its coordination needs | PSI formalized operationally; WSI strictly between PSI and SI 1 . PSI as PL-2+; PSI \wedge PC = SI 3 7 . . | Unified taxonomy aligning PSI variants (per-key/global arbitration; with/without PC) with availability status is missing; AFC provides only examples for PSI unavailability 1 3 4 7 . . | Prevents under/over-constraining systems; clarifies portability of “PSI” claims. | Produce a machine-checked lattice of PSI variants, their equivalences, and availability under AFC; design and prove an “arbitration-free PSI-like subset.” | There exists a PSI variant (strictly stronger than CC but weaker than WSI) that is arbitration-free in AFC’s sense while preserving atomic visibility. 1 3 4 7 |
| Availability with atomic visibility | AFC theorem gives necessary/sufficient condition; TCC+ aims to be strongest AP model with atomic snapshots 4 9 . . | Formal placement of TCC+ in AFC terms; minimal coordination needed for atomic snapshots not fully characterized. | Guides geo/edge system design and when to deploy “SI zones.” | Formalize TCC+ in AFC’s framework; prove whether it is arbitration-free. Derive lower bounds on coordination for atomic snapshots under causality. | TCC+ is arbitration-free and thus fully available; any strictly stronger atomic-visibility model requires arbitration. 4 9 |

| | | | | | |
|---|---|--|--|--|---|
| CRDTs + multi-key transactional invariant s | SEC foundations 8 ; parametric CRDT verification under EC/CC/PSI 6 ; bounded exploration and selective strengthening 5 ; TCC+ composes CRDTs with atomic txns 9 | General, implementable rules (e.g., escrow/credis) that guarantee invariant-preserving composition across keys under PSI/WSI/TCC+; scaling verification beyond bounded search 5 6 8 9 . | Needed for practical, safe-by-construction apps on KV+CRDT stacks. | Develop “invariant-preserving PSI/TCC+” protocols (escrow/reservations) with proofs and tool support; integrate with 5 6 . | Under PSI/TCC+, per-key escrow plus atomic snapshots suffices to preserve common linear constraints without global coordination. 5 6 9 |
| End-to-end equivalence across lenses | Operational↔axiomatic equivalence with protocol proofs 1 ; state-based↔history-based equivalence and composition 3 ; AV axioms robust for program reasoning 2 10 . . | Equivalences covering WSI generally and TCC+ explicitly; opacity for uncommitted effects is outside 3 . . | Ensures definitions match implementable systems and program-level reasoning. | Mechanize equivalence for WSI and TCC+ across abstract executions, state sequences, and operational KV; extend to opacity/virtual-world consistency. | WSI and TCC+ admit canonical abstract-execution axioms equivalent to operational semantics used by implementers. 1 2 3 9 10 |

| | | | | | |
|---|--|---|--|---|---|
| Costs, metadata, and anomaly incidence | Composition result $\text{PSI} \wedge \text{PC} \approx \text{SI}$ informs strengthening 7 ; TCC+ discusses metadata qualitatively vs SI 9 ; AFC links arbitration to coordination in principle 4 . | Quantitative models predicting latency/metadata vs anomaly rate across topologies; shared benchmarks for anomaly frequency under PSI/WSI/TCC+/SI. | Lets practitioners pick the “just-enough” model. | Build an anomaly benchmark suite and telemetry method; derive analytic cost models calibrated with experiments. | In typical OLTP+social workloads, WSI delivers SI-level anomaly rates at significantly lower coordination cost than SI. 1 4 7 9 |
| Non-snapshot models and alternative arbitration | AV family and KV semantics focus on atomic snapshots; LWW is the default arbitration 1 2 3 10 . | Unified treatment of RC/RC+ and custom merges (beyond LWW) in KV with formal anomaly catalogs is missing. | Many production systems expose RC variants and custom CRDT merges. | Extend AV/operational frameworks to parameterize arbitration/merge (incl. CRDT joins); map to Adya phenomena. | Some non-LWW CRDT merges can simulate SI’s no-lost-update per key without arbitration. 1 2 3 8 10 |
| Prefix constraints and their enforcement | $\text{PSI} \wedge \text{PC} \approx \text{SI}$ clarifies theory 7 . | Practical, scalable protocols for PC at scale; how PC interacts with sharding/relication in KV. | A path to SI via lightweight strengthening. | Design and evaluate PC protocols under partitioning; prove composition with PSI in deployed architectures. | Enforcing PC with per-shard metadata yields near-SI behavior with sub-SI cost for most workloads. 7 |

| | | | | | |
|--------------------------------|--|--|--|--|---|
| Tooling scale and completeness | SMT-based anomaly detection for tx programs 7 ; bounded search and repair for replicated types 5 ; parametric CRDT verification 6 . | Scaling beyond bounded concurrency ; integrating tools across models (PSI/WSI/TCC+/SI) and languages; soundness vs completeness envelopes. | Needed for CI pipelines and regression testing of consistency. | Create a common intermediate representation for vis/ar/hb models and compose analyzers; incremental/under-approximate search with proof obligations. | For real applications, a hybrid (bounded search + inductive summaries) catches >95% of harmful patterns under PSI/TCC+. 5 6 7 |
| Liveness and failure semantics | TCC+ states Eventual Visibility; convergence emphasized 9 . | Precise liveness assumptions (fairness, partitions, churn) and their effect on guarantees are under-specified comparatively. | Impacts user-visible staleness and progress. | Formalize liveness in a shared model; derive staleness bounds under TCC+ vs PSI/SI; test on edge topologies. | Under realistic edge churn, TCC+ bounds snapshot staleness without coordination, whereas PSI variants need arbitration to match. 4 9 |

| Term | Plain definition | Why it matters / what it controls | Quick example | Refs |
|-----------|---|-----------------------------------|-------------------|-------|
| Operation | A single read or write to a key/object. | Building block of transactions. | GET(x), PUT(x=5). | 1 2 3 |

| | | | | |
|----------------------------|--|---|---|------------------------|
| Transaction | A group of operations that should appear to happen atomically. | The unit of isolation and atomic visibility. | Transfer: read a, read b, write a-1, write b+1. | 1 2 3 7 10 |
| so (session/program order) | The order in which one client issues operations/transactions. | Encodes “my actions happen in the order I did them.” | T1 then T2 from the same client ⇒ T1 so→ T2. | 1 2 3 8 |
| rt (real-time order) | Wall-clock order of completed transactions. | Required for strict serializability; not required by SI/PSI. | If T1 finishes before T2 starts, then T1 rt→ T2. | 2 3 7 |
| vis (visibility) | Which committed writes a transaction can see (its snapshot). | Drives reads: a read must see some writer in vis. | If T2 sees T1's write, then T1 vis→ T2. | 1 2 3 7 10 |
| ar (arbitration) | A total order used to break ties among concurrent writes. | Introduces coordination; many models rely on it. | If T1 and T2 write x concurrently, ar decides “winner.” | 3 4 |
| hb (happens-before) | Causality: transitive closure of so and vis (and messages). | Snapshot must be closed under hb in causal/PSI models. | If T1 vis→ T2 and T2 so→ T3, then T1 hb→ T3. | 2 3 8 |
| wr / ww / rw edges | Dependencies in a graph of transactions: write→read, write→write, read→write. | Cycles over these edges characterize anomalies and (non-)serializability. | T1 writes x, T2 reads x ⇒ wr edge T1→T2. | 1 2 3 7 10 |
| Dependency graph | Graph whose nodes are transactions and edges are wr/ww/rw. | Serializability = “no cycle” in this graph. | A triangle of wr/rw edges forming a cycle breaks SR. | 2 3 7 |
| Abstract execution | Axiomatic description of one run using vis/ar (and hb). | Clean, model-agnostic way to define consistency levels. | A set of txns + relations satisfying model axioms. | 2 10 1 7 |

| | | | | |
|-------------------|---|--|---|---|
| State-based model | Describes systems as sequences of observable states with constraints. | Equivalent to abstract-execution definitions but often easier to reason about composition. | Reasoning over state snapshots rather than histories. | 3 |
|-------------------|---|--|---|---|

Isolation/consistency models

| Term | Plain definition | What it guarantees (and allows) | Quick example | Refs |
|-----------------------------------|--|--|---|------------------------|
| Serializability (SR / PL-3) | The outcome is as if all transactions ran one-by-one in some order. | Forbids all wr/ww/rw cycles (no write skew). | There exists a single total order matching the outcome. | 2 3 7 10 |
| Strict serializability | Serializability that also preserves real-time order (rt). | Strongest transactional model considered here. | If T1 finished before T2 began, T1 appears before T2. | 2 3 7 |
| Snapshot Isolation (SI) | Each txn reads a consistent snapshot and concurrent same-key writes can't both commit. | No dirty/intermediate reads; no per-key lost updates; allows write skew (rw-only cycle). | Two doctors update disjoint rows → possible write skew. | 1 2 3 7 10 |
| Parallel Snapshot Isolation (PSI) | Causally consistent snapshots plus per-key conflict checks; no single global snapshot. | No dirty/intermediate reads; per-key no lost update; can allow write skew. | Sharded/geo system: check conflicts per key only. | 1 3 5 7 9 |
| Weak Snapshot Isolation (WSI) | A model strictly between PSI and SI (in the KV semantics). | Stronger than PSI, weaker than SI; reduces some anomalies admitted by PSI. | Blocks some PSI-allowed patterns without full SI cost. | 1 |
| Causal Consistency (CC) | Every txn sees effects of all its causal predecessors (hb-closed). | Preserves causality; may allow concurrent conflicts. | Read-your-writes and observed-writes preserved. | 3 8 9 |
| Causal+ (CC+) | Causal consistency plus session guarantees like RYW, MW, MR, WFR. | Strengthens CC for user sessions. | Your later ops see your earlier writes and reads. | 3 9 |

| | | | | |
|---|--|--|---|---|
| Transactional Causal+ (TCC+) | Causal+ extended to multi-key transactions with atomic snapshots and convergence. | Availability-friendly atomic visibility; weaker than SI. | Edge system with atomic causal snapshots. | 9 |
| Consistent Prefix (PC) | Reads reflect a prefix of a per-shard or global commit order (no “future without past”). | Avoids non-monotonic snapshots across keys. | If you see version 5, you must see ≤ 5 on that log. | 7 |
| $\text{PSI} \wedge \text{PC} \approx \text{SI}$ | Composition result: PSI plus prefix consistency equals SI (under the formalization in 7). | Design route to reach SI by adding PC to PSI. | Take PSI system, add prefix order enforcement \rightarrow SI. | 7 |

Common anomalies (Adya phenomena and named patterns)

| Term | Plain definition | Which models forbid/allow it | Quick example | Refs |
|---------------------------------|---|--|--|------------------|
| G0 (write cycle) | Cycle of writes that require conflicting orders. | Forbid: SI/PSI/SR. | T1: $x=1$; T2: $x=2$, both “win” inconsistently. | 2 7 10 |
| G1a (dirty read) | Read sees an uncommitted write. | Forbid: SI/PSI/SR. | T2 reads T1’s write before T1 commits. | 2 7 10 |
| G1b (intermediate read) | Read sees a non-final value written inside a txn. | Forbid: SI/PSI/SR (atomic visibility). | Read an intermediate step of a multi-write txn. | 2 7 10 |
| G1c (circular information flow) | Circular wr dependencies via reads that violate snapshot closure. | Forbid: SI/PSI/SR. | T1 reads T2, T2 reads T1 in a way violating hb-closure. | 2 7 10 |
| G2 (anti-dependency cycle) | Cycle containing rw edges (write skew). | Allow: SI/PSI; Forbid: SR. | Two txns read each other’s old values and write disjoint keys. | 1 2 3 7 |

| | | | | |
|-----------------|--|--|---|------------------|
| Lost update | Two concurrent writes on the same key where one overwrites the other unintentionally. | Blocked per key by SI/PSI; also avoided by many CRDTs via merge. | Both increment x; only one effect remains. | 1 3 7 8 |
| Write skew | Each txn reads values the other later changes, and both commit since no same-key conflict. | Allowed by SI/PSI; disallowed by SR. | Two doctors clear each other's on-call flags. | 1 2 3 7 |
| Fractured reads | See some but not all writes of a committed multi-key txn. | Forbid in atomic-visibility family (SI/PSI/SR/TCC+). | Read a=1 (new) but b=old from same txn T. | 1 2 7 9 |

CRDTs and convergence

| Term | Plain definition | Why it matters | Quick example | Refs |
|---|--|--|---|------|
| CRDT (Conflict-free Replicated Data Type) | A data type whose concurrent updates converge automatically across replicas. | Per-key eventual convergence without global order. | Grow-only set, PN-counter. | 8 |
| State-based CRDT | Each replica periodically merges states using a least upper bound (join). | Merge is associative/commutative/identity/dempotent \Rightarrow convergence. | Two sets merge by union. | 8 |
| Op-based CRDT | Replicas send operations, delivered causally; concurrent ops are designed to commute. | Requires causal delivery for SEC. | Add/Remove operations with causal tags. | 8 |
| SEC (Strong Eventual Consistency) | If replicas apply the same set of updates (possibly in different orders), they reach the same state. | Guarantees convergence without arbitration. | Offline edits reconcile to same result. | 8 |

Arbitration, availability, and costs

| Term | Plain definition | Why it matters | Quick example | Refs |
|--|--|---|---|--------|
| Arbitration (total order) | A single total order used to decide winners among concurrent writes/txns. | Implies coordination; can hurt availability/latency. | “Last writer wins” according to a global timestamp. | 3 4 |
| LWW (Last-Writer-Wins) | A common arbitration rule: the write with the greatest timestamp wins. | Simple conflict resolution; may discard concurrent updates. | Two writes to x; larger ts value is kept. | 1 3 |
| AFC theorem (Arbitration-Free Consistency) | A spec has an available implementation iff it doesn't require a total arbitration order in its formulas. | Formal boundary between availability and coordination. | Shows why SC/SI/PSI-like constraints often need coordination. | 4 |
| Availability (AP sense) | A replica can answer requests without waiting on others (even under partitions). | Desired for geo/edge; limited by need for arbitration. | Local write completes despite network partition. | 4 9 |
| “SI zones” | Regions in a system where stronger SI is enforced while the rest uses a more available model. | Hybrid design to balance latency vs anomalies. | Edge nodes run TCC+, datacenter enforces SI. | 9 |

Session-level guarantees

| Term | Plain definition | Why it matters | Quick example | Refs |
|------------------------|---|---|---|--------|
| Read-Your-Writes (RYW) | After you write, your later reads see that write. | Improves user experience and reasoning. | Post a comment; refresh shows it. | 3 9 |
| Monotonic Reads (MR) | Your reads never go backward in time. | Avoids “time travel” across pages. | Don't see an older profile after a newer one. | 3 9 |

| | | | | |
|---|---|---|---|--------|
| Monotonic Writes (MW) | Your writes are applied in your issue order. | Preserves user intent. | Edit A then B; system doesn't apply B then A. | 3 9 |
| Writes-Follow-Reads (WFR) | Your writes are ordered after the data you just read. | Keeps causality between your read context and next write. | Like/reply after reading a post keeps that order. | 3 9 |
| CC ≡ four session guarantees (for txns) | Causal consistency is equivalent to RYW+MR+MW+WFR (in the state-based framework). | Lets you implement CC by providing session guarantees. | Providing all four yields CC behavior. | 3 |

Snapshot/atomicity extras

| Term | Plain definition | Why it matters | Quick example | Refs |
|-------------------------------------|--|--|---|------------------|
| Snapshot | The set of committed effects a txn reads. | Must be consistent (e.g., hb-closed) under many models. | A read sees a picture of the DB at some logical time. | 1 2 3 7 |
| Atomic visibility | If any write of a txn is visible, then all its writes are visible. | Prevents fractured reads. | Read both new a and new b from the same txn. | 2 9 10 |
| Opacity / Virtual-world consistency | Ensures even uncommitted/aborted txns only see consistent states. | Stronger safety for long-running txns; not covered by some frameworks. | A txn never reads partial/invalid states while it runs. | 3 |

| Term | Plain definition | Why it matters | Quick example | Refs |
|------------------------|---|---|---|------|
| Execution test (in 1) | A parametric check that decides whether a txn can commit under a model. | Lets one operational semantics instantiate many models. | “Commit if no per-key ww conflict with visible txns.” | 1 |

| Open question (plain words) | Why it matters | What we know (key refs) | What's missing (gap) | Good next steps | Example hypothesis to test |
|--|---|---|---|---|---|
| What exactly is PSI, and when does it need coordination? | Teams use “PSI” differently; wrong assumptions cause bugs or extra latency. | PSI = causal snapshots + per-key conflict checks; WSI sits between PSI and SI 1 . PSI is in the PL-2+ “lazy” family; PSI \wedge Prefix Consistency \approx SI 3 7 . | A single, agreed map of PSI variants (with/without Prefix Consistency, per-key vs global arbitration) and their availability status is missing 1 3 4 7 . | Build a machine-checked “PSI family” taxonomy; prove which variants are equivalent and which require total order (arbitration) per AFC 4 | There exists a PSI-like variant (stronger than Causal, weaker than WSI) that needs no total order and stays available under partitions. 1 3 4 7 |

| | | | | | |
|--|--|--|--|---|---|
| <p>How strong can atomic visibility be while staying highly available?</p> | <p>Edge/geo systems want strong safety without blocking on the network.</p> | <p>AFC: a model is implementable in an always-available way iff it avoids total arbitration</p> <ul style="list-style-type: none"> 4 . TCC+ aims to be the strongest availability-friendly transactional model with atomic snapshots 9 . | <p>Formal placement of TCC+ under AFC is not shown; minimal coordination needed for atomic snapshots under causality not characterized</p> <ul style="list-style-type: none"> 4 9 . | <p>Formalize TCC+ in AFC's logic; prove whether it is “arbitration-free.” Derive lower bounds on coordination for atomic snapshots.</p> | <p>TCC+ is arbitration-free; any strictly stronger atomic-snapshot model needs some total order and thus is not fully available.</p> <ul style="list-style-type: none"> 4 9 |
| <p>How to keep multi-key app invariants with CRDTs + transactions?</p> | <p>Real apps need cross-key rules (e.g., totals, limits) and also want CRDT convergence.</p> | <p>Strong eventual consistency for per-key CRDTs is well-understood</p> <ul style="list-style-type: none"> 8 . Tools explore invariants and strengthen consistency selectively 5 ; CRDT convergence can be reasoned under EC/CC/PSI policies 6 . TCC+ composes CRDTs with atomic snapshots 9 . | <p>A general recipe (e.g., escrow/credits) for safe cross-key invariants under PSI/WSI/TCC+, with proofs and tooling, is missing</p> <ul style="list-style-type: none"> 5 6 8 9 . | <p>Design invariant-preserving “escrow CRDT transactions” for PSI/TCC+; integrate into analyzers from</p> <ul style="list-style-type: none"> 5 6 ; evaluate on standard workloads. | <p>With per-key escrow and atomic snapshots, many linear constraints ($\text{sum} \leq K$) hold without global coordination.</p> <ul style="list-style-type: none"> 5 6 9 |

| | | | | | |
|--|--|---|--|---|--|
| Do our different formalisms truly agree for all models used in practice? | Designers and verifiers need the same meaning across papers and tools. | <p>Equivalence: operational KV \leftrightarrow axiomatic for CC/PSI/SI/SR; protocol proofs (COPS, Clock-SI)</p> <ul style="list-style-type: none"> 1 . State-based \leftrightarrow classical histories; composes session guarantees 3 . Atomic-visibility axioms power program robustness 2 10 . | <p>No end-to-end equivalence yet for WSI (all variants) and TCC+; opacity (safe views for running txns) not covered in state-based model</p> <ul style="list-style-type: none"> 1 2 3 9 10 . | <p>Mechanize equivalences for WSI and TCC+ across abstract executions, state sequences, and operational semantics; extend state-based model with opacity.</p> | <p>WSI and TCC+ have axioms equivalent to their operational definitions, enabling sound program analysis.</p> <ul style="list-style-type: none"> 1 2 3 9 10 |
| What are the real costs vs benefits (latency, metadata, anomaly rates)? | Choosing SI/WSI/PSI/TCC+ needs data, not guesses. | <p>Theory links arbitration to coordination in principle</p> <ul style="list-style-type: none"> 4 ; PSI \wedge Prefix Consistency \approx SI suggests a strengthening path 7 ; TCC+ discusses metadata qualitatively 9 . | <p>Few quantitative studies of end-to-end cost and anomaly frequency under real topologies/work loads</p> <ul style="list-style-type: none"> 4 7 9 . | <p>Create an open benchmark that logs anomalies and costs across models; publish latency/throughput/metadata vs observed anomaly rates.</p> | <p>In typical OLTP/social workloads, WSI achieves SI-like low anomaly rates at lower latency than SI.</p> <ul style="list-style-type: none"> 1 4 7 9 |

| | | | | | |
|---|--|--|--|---|---|
| Can we support non-snapshot isolation and non-LWW merges in one clean theory? | Many systems use Read Committed variants or custom merges (not just last-writer-wins). | Current frameworks focus on atomic snapshots; LWW is often assumed 1 2 3 10 . | Unified treatment of RC/RC+ and custom CRDT merges with a clear anomaly map is missing. | Extend axioms/semantics to parameterize merge/arbitration (incl. CRDT joins); relate each choice to Adya anomalies. | Certain CRDT merges can give “no lost update per key” without a global order, reducing coordination vs LWW-based PSI. 1 2 3 8 10 |
| How to enforce Consistent Prefix at scale and use it to reach SI? | PC prevents “seeing the future” and, with PSI, can yield SI. | Formal result: PSI \wedge PC \approx SI 7 . | Scalable protocols for PC in sharded/geo settings and empirical evaluation of PSI \rightarrow SI via PC are lacking. | Design shard-friendly PC protocols; measure cost/benefit; verify composition with PSI in real deployments. | Shard-local PC metadata plus PSI delivers near-SI behavior with much lower coordination than full SI. 7 |
| How can tools scale and still be trustworthy? | Developers need CI-friendly checks for anomalies/invariants. | SMT-based serializability violation detection under weak models 7 ; bounded search and repair for replicated types 5 ; parametric CRDT | Scaling beyond small bounds; combining tools/models (PSI/WSI/TCC +/SI) with soundness guarantees is open. | Create a shared IR for vis/ar/hb; compose analyzers; use incremental and under-approximate search with proof obligations. | A hybrid (bounded search + inductive summaries) catches >95% of harmful patterns under PSI/TCC+ on real apps. 5 6 |

| | | | | | |
|--|--|---|--|--|--|
| | | convergence proofs 6 . | | | 7 |
| What are the liveness and staleness guarantees under churn/partitions? | Users care how fresh data is and whether txns make progress. | TCC+: Eventual Visibility; convergence emphasized 9 . AFC draws the availability boundary 4 . | Precise, comparable liveness/staleness bounds across models are not formalized or measured. | Define shared liveness models; derive and test staleness bounds for TCC+/PSI/SI on edge/geo topologies. | Under realistic edge churn, TCC+ bounds snapshot staleness without coordination; PSI needs arbitration to match. 4 9 |
| How should WSI be used in practice? | WSI might give “just-enough” safety for lower cost. | WSI is strictly between PSI and SI in a KV semantics 1 . | Which anomalies WSI still admits, and when it pays off vs SI/PSI, are not empirically charted. | Catalog WSI’s allowed/blocked patterns; implement WSI in a testbed; compare cost/anomaly trade-offs to SI/PSI. | WSI blocks most PSI anomalies that matter to apps, at notably lower cost than SI. 1 |