Research Report on

Graduate-level comparative review of KV-store transactional/CRDT consistency models

Full search query: I want to find a graduate-level, theory-focused literature review on consistency models for key-value stores limited to transactional models and CRDTs—covering serializability, snapshot isolation, PSI and related transactional semantics, and CRDT/strong eventual consistency—presented using formal histories and order relations (real-time order, happens-before) and classic anomaly catalogs (e.g., Adya's phenomena) to compare models, articulate strictness/implication relations, and deliver a comparative analysis and gap analysis of how these models apply to key-value stores

Summary

A single, comprehensive graduate-level survey exactly matching your brief was not found, but the combination of transactional atomic-visibility foundations and equivalence results [2,3,10], operational KV semantics with model lattice (including WSI) and protocol proofs [1], axiomatic program-analysis results (e.g., PSI ' Prefix Consistency C SI) [7], CRDT/SEC theory [8] and its bridges to transactions [5,6,9], and availability/arbitration limits [4] together provide the ingredients to build the desired review.

Core foundations you can rely on

- Atomic-visibility, abstract-execution axioms for transactional histories:
  - Formal relations vis, ar, hb and dependency graphs (wr/ww/rw) characterize CC/PSI/SI/SR and tie guarantees to Adya phenomena (G0/G1/G2) [2,10].
  - These axioms provide snapshot closure (no fractured reads) and the anomaly lens needed for comparative analysis [2,10].
- State-based equivalence and compositional reasoning:
  - A unified, state-sequence framework proves equivalence to classical history-based definitions, clarifies anomaly characterizations (e.g., write skew), and composes session guarantees (CC a four transactional session guarantees) [3].

Transactional models (SR, SI, PSI) in KV terms

- Formal characterizations using histories/orders:
  - Serializability: cycle freedom over wr/ww/rw among committed transactions; strict serializability additionally preserves real-time order rt [2,3,7,10].
  - Snapshot Isolation: hb-closed snapshots plus prevention of concurrent ww conflicts; forbids G0/G1 but allows G2 rw-only cycles (write skew) [1,2,3,7,10].
  - Parallel Snapshot Isolation: causally consistent snapshots (hb-closed) with per-key ww conflict checks; forbids G0/G1 and per-key lost updates, admits write skew; avoids global snapshots [1,3,5,7,9].
- Operational KV semantics with axiomatic correspondence:
  - A centralized, multiversion KV semantics with partial client views is shown equivalent to declarative abstract executions, instantiating CC/PSI/SI/SR and verifying real KV protocols (COPS for CC, Clock-SI for SI) [1].
- Notable refinements and relations:
  - Weak Snapshot Isolation (WSI) is introduced and strictly positioned between PSI and SI within the KV semantics [1].
  - PSI composed with Consistent Prefix yields SI (PSI ' PC C SI), sharpening model-composition understanding for KV [7].

Strictness/implication lattice (KV transactional context)

- Ordering (informal but supported across the cited axioms and results):
  - Strict serializability > serializability > SI > WSI > PSI > Causal+ with atomic visibility > causal consistency [1,2,3,7,9,10].
- Key equivalences and separations:
  - CC equivalent to the four transactional session guarantees [3].
  - PSI weaker than SI by admitting rw-only cycles (write skew) [1,2,3,7].
  - WSI reduces anomalies permitted under PSI but remains weaker than SI [1].

Anomaly catalog (Adya phenomena) mapping

- Models with atomic visibility (PSI/SI/SR) forbid:
  - G0 (write cycles) and G1 (dirty and intermediate reads), as ensured by snapshot closure and commit rules [1,2,7,10].

- Distinguishing anomaly:
  - G2 rw cycles (write skew) are allowed under PSI and SI, disallowed under serializability [1,2,3,7].
- Additional signatures relevant to KV:
  - Lost update handled by ww-conflict prevention under SI/PSI and by CRDT merge semantics per key; cross-key invariants can still fail without stronger transactional constraints [1,3,5,7,8].
  - Fractured reads are excluded in the atomic-visibility family and explicitly in TCC+ [1,2,7,9,10].
  - Consistent-prefix violations are addressed in composition results (PSI ' PC C SI) [7].

CRDTs and strong eventual consistency (SEC), and how they compose with transactions

- CRDT/SEC baseline:
  - Formal definitions of so/vis/hb, causal consistency, and SEC for state- and op-based CRDTs; convergence without arbitration; per-object focus [8].
- Bridges to transactional KV:
  - Bounded-concurrency exploration detects invariant violations for replicated types, models multi-object atomicity, and formalizes PSI constraints; suggests selective strengthening to meet invariants [5].
  - Automated CRDT convergence verification parameterized by consistency policy (eventual, causal, PSI) shows stronger policies can relax per-operation commutativity requirements and ease composition [6].
  - Transactional Causal Plus (TCC+) extends Causal+ with transactional atomic visibility and causally/atomically sound snapshots; integrates CRDTs with convergence and rollback-freedom; positions as the strongest availability-compatible transactional model and uses local SI “zones” for stronger isolation where coordination is feasible [9].
- Orthogonality and composition insight:
  - SEC guarantees per-key convergence and is orthogonal to cross-key transactional anomalies; transactional atomic visibility is required to avoid fractured reads when composing multi-key CRDT operations [8,9].

Availability and arbitration: theory-guided limits

- Arbitration-Free Consistency theorem:
  - A specification admits an available implementation iff its visibility formulas do not require a total arbitration order; models relying on arbitration (examples include PC/PSI/CCC and, more strongly, SI/SR variants) necessitate coordination and are not fully available under partitions [4].
- Consequences for KV design:
  - Causality-centric transactional models (e.g., TCC+) can remain AP, while PSI/SI entail coordination costs; hybrid designs with availability-friendly baselines and local SI regions are theoretically motivated [4,9].

Applicability to key-value stores and what is concretely supported

- KV-centric formalisms and proofs:
  - Operational KV semantics and protocol verifications for CC and SI [1].
  - Axiomatic program encodings for replicated, weakly consistent transactional databases, including PSI/SI and serializability-by-cycle detection; compositional relations across models [7].
- Tooling and verification:
  - Automated anomaly detection for transactional programs under weak models (including PSI) [7].
  - CRDT verification and invariant repair under parametric policies spanning EC/CC/PSI [5,6].

Gaps and opportunities for the desired literature review

- What’s missing:
  - No single, integrative survey spans SR/SI/PSI together with CRDT/SEC and uses formal histories/orders plus Adya’s phenomena to deliver a KV-focused implication lattice and comprehensive gap analysis [1,2,3,4,5,6,7,8,9,10].
- What you can synthesize:
  - Use atomic-visibility axioms and state-based equivalences to ground transactions [2,3,10]; employ operational KV semantics and WSI to refine the lattice and connect to real protocols [1]; add compositional and anomaly-detection results (PSI ' PC C SI; cycle-based serializability) [7]; bring in CRDT/SEC theory and parametric verification to cover per-key convergence and transactional composition [5,6,8,9]; and frame availability trade-offs via the AFC theorem and TCC+ [4,9].
- Suggested structure for your review:

- Foundations (vis/ar/hb, dependency graphs, Adya phenomena) [2,3,10].
- Transactional models for KV (SR/SI/PSI, WSI, equivalences, anomalies) [1,2,3,7].
- CRDT/SEC and transactional composition (atomic visibility, fractured reads, invariants) [5,6,8,9].
- Availability vs arbitration and design patterns (hybrids with SI zones) [4,9].
- Comparative analysis and strictness lattice, ending with a gap analysis and research directions.

Categories

Comparative scope and positioning across the retrieved papers

The table below contrasts each paper along dimensions that matter for a theory-focused, graduate-level review of key-value transactional models (SR/SI/PSI) and CRDT/SEC, using formal histories/orders and anomaly catalogs, and aiming for strictness/implication analysis and gap identification.

Ref	Genre/role	Transactional models (SR/SI/PSI)	CRDT/SEC	Formal relations used (hb/rt/vis/ar, dep graphs)	Adya phenomena/anomaly catalog	Implication/strictness results	KV transactional scope	Availability/coordination insights	Notable for
[1]	Theory framework + model unification + protocol proofs	Yes (CC, PSI, SI, SR; introduces WSI between PSI and SI)	Per-key LWW; CRDTs per se not the focus	Yes: abstract executions, dependency graphs; vis/ar equivalence with operational semantics	Implicit via equivalence to classic definitions; anomaly classes discussed via model mappings	Yes: positions WSI strictly between PSI and SI; equivalence between operational and declarative specs	Yes: multiversion centralized KV with client views; verifies COPS (CC) and Clock-SI (SI)	Indirect: assumes last-writer-wins; coordination via “execution tests”	Clean bridge between axiomatic (vis/ar/dep) and operational KV semantics; new WSI result
[2]	Foundational axiomatic framework (atomic visibility)	Yes (family with atomic visibility including SI/PSI/SR)	Not CRDT-focused	Yes: vis/ar, hb closure; atomic visibility axioms	Yes (classic G0/G1/G2 grounding across models)	Yes (positions isolation levels with atomic visibility)	Yes: transactional KV abstract executions	Indirect (focus on axioms rather than availability)	Canonical reference for atomic-visibility transactional axioms
[3]	Unified state-based framework (consistency + isolation)	Yes (SR, SI, PSI; composition with session guarantees)	Discusses causal/SEC-style constraints via visibility; not CRDT-centric	Yes: visibility/arbitration over observable states; equivalence to traditional history-based defs	Discusses anomalies (e.g., write skew) within state-based lens	Yes: clarifies relations; shows CC a four session guarantees for transactions; positions PSI in “lazy/PL-2+” family	Yes: transactional workloads abstracted as state transitions	Indirect: argues PSI without per-site total orders	Bridging state- vs history-based formulations; compositional reasoning about guarantees
[4]	Theory (availability vs arbitration theorem)	Yes (includes SI, PSI, SR formalization)	Includes CRDT objects among others	Yes: explicit vis formulas and arbitration; hb vs ar contrasted	Not the focus	Yes (arbitration use implies coordination); impossibility examples for PC/PSI/CCC availability	Yes: multi-object (incl. KV ops, SQL txns)	Central: AFC theorem—available iff arbitration-free; PSI/PC/CCC shown non-available in examples	A principled link between model definitions and availability constraints
[5]	Program analysis/verification tool (Q9)	Yes (formalizes PSI; ATOMicity axiom for txns)	Yes (RDTs/CRDTs; invariant checking and repair)	Yes: vis/hb axioms; per-object order constraints	Indirect (through model axioms)	Shows selective strengthening to meet invariants; formal PSI constraints	Yes: multi-object transactions on eventually consistent store	Yes: explores strengthening trade-offs vs availability	Practical synthesis/verification against weak’strong policies incl. PSI
[6]	Automated CRDT	Partial (policies	Yes (SEC under	Yes: event-based		Observes stronger	Yes: multi-object	Indirect (policies	Cross-model view of

	convergence verification	parameterize proofs; includes PSI)	eventual/causal/PSI)	executions; policy axioms	Not the focus	policies can obviate commutativity needs; compares under EC/CC/PSI	settings possible; op-based CRDTs	reflect coordination)	CRDT convergence under EC/CC/PSI
[7]	Automated detection of serializability violations	Yes (formal FOL axioms for CC, PC, PSI, SI; serializability via dep cycles)	Not CRDT-focused	Yes: vis/ar; dependency graphs wr/ww/rw; program-to-axiom encoding	Yes (serializability ”cycle freedom)	Yes: shows PSI ' PC C SI; combines models compositionally	Yes: replicated, weakly-consistent transactional DBs (KV-like)	Indirect	Formal, tool-backed anomaly detection under PSI/SI; compositional model relations
[8]	CRDT survey	No transactional SI/PSI/SR	Yes (SEC; hb/vis, causal consistency)	Yes: so/vis/hb; SEC formalism	Not transactional anomalies	N/A within tx semantics	Per-object CRDTs; not multi-key tx	Discusses AP/CP landscape	Best formal CRDT/SEC tutorial; lacks cross-key transactional analysis
[9]	System + model (TCC+)	Yes (TCC+ vs CC+, PSI, SI)	Yes (CRDTs with TCC+)	Yes: vis/ar; atomic snapshots; transactional equivalence (a)	Indirect (focus on snapshot/atomicity)	Positions TCC+ as strongest availability-compatible transactional model; SI used in “zones”	Yes: transactional causal with atomic visibility; geo/edge KV	Yes: model aims for availability with convergence; SI zones require coordination	Realistic composition of TCC+ with local SI; atomic visibility emphasized
[10]	Robustness under atomic-visibility models	Yes (family with atomic visibility; relates to SI/PSI)	Not CRDT-focused	Yes: abstract executions/vis/ar	Indirect	Program robustness criteria across SI/PSI-like models	Yes: transactional KV abstractions	Indirect	Program-level guarantees under SI/PSI-family models

Key takeaways:

- The strongest theory unifiers for KV transactions with formal histories/orders are [1,2,3,7,10].
- The best CRDT/SEC formal survey is [8], but it does not cover multi-key transactional isolation; [5,6,9] are the main bridges between CRDTs and transactional semantics.
- Availability vs arbitration (and implications for PSI/SI) is uniquely addressed by [4] and practically instantiated in [9]’s TCC+ design.

PSI characterizations and their relationship to SI, SR, and availability

Aspect	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[9]
Snapshot definition	Client views + snapshot property (atomic visibility), LWW; PSI as causally closed snapshots + per-key ww checks	Axiomatic atomic visibility; PSI defined in the AV (atomic visibility) family	State-based snapshots with visibility/ar constraints; PSI within “lazy/PL-2+” style	PSI formalized via vis/conflict predicates; examples show non-availability (requires arbitration/coordination)	PSI axioms over vis/hb; “all writes to same object in same order everywhere”	PSI as a policy constraining delivery/visibility in event model	PSI axiomatized; combined with Prefix Consistency yields SI (PSI ' PC C SI)	PSI contrasted with TCC+; TCC+ aims for strongest availability-compatible model; PSI stronger than availability-compatible baseline
Anomalies prevented	No dirty/intermediate reads; no lost updates per key; allows rw-only cycles; atomic visibility by assumption	Same (AV axioms forbid G0/G1; allow some G2)	Same; clarifies write-skew under PSI	Shows why coordination is needed (arbitration) to prevent certain anomalies	Same; tool explores strengthening to remove anomalies	Not anomaly-focused	Same; serializability violations found under PSI; SI avoids them in cases	TCC+ avoids fractured reads and preserves atomic snapshots; still allows more than SI while staying availability-friendly
Relation to SI				PSI requires arbitration, hence	PSI weaker than SI; can	Policy lattice not explicit; PSI compared	PSI ' PC C SI; SI strictly	PSI stronger than causal/TCC+;

	WSI strictly between PSI and SI	PSI < SI within AV family	PSI in PL-2+ family; SI stronger	coordination; SI even more so	be strength-ened	to EC/CC for CRDT con-vergence	stronger than PSI	SI used in zones with stronger coordination
Availability stance	Not pri-mary; as-sumes coor-dination when needed	N/A	N/A	PSI not “arbitra-tion-free” ’ no fully available implementa-tion under partition	Explores trade-offs between availability and strengthening	Stronger policy (PSI) implies more coordination than EC/CC	N/A	TCC+ targets availability, SI used selec-tively

Citations: [1,2,3,4,5,6,7,9].

Formal machinery: histories, orders, and anomaly grounding

- Histories and relations:
  - Axiomatic vis/ar frameworks with hb closure are central in [1,2,3,7,10]; [3] rephrases over state sequences while proving equivalence to classic history-based definitions.
  - Dependency graphs (wr/ww/rw) and cycle-based serializability appear explicitly in [1] (via equivalence), [7] (as the core property), and are compatible with [2,10].
  - Per-key vs global orders and LWW arbitration are explicit assumptions in [1,5,7]; [4] separates causality (hb/vis) from arbitration (ar) and ties the latter to coordination.
- Anomaly catalogs:
  - Adya’s phenomena and classic anomalies are used as the comparison lens or implicitly encoded in the axioms in [1,2,3,7,10]; [7] concretely uses cycle detection (G2) to witness non-serializable behaviors under PSI; [1,2] ensure snapshot closure forbids G1 phenomena; [3] discusses write skew under SI/PSI.

Where CRDTs and transactions meet (and diverge)

- Strong eventual consistency:
  - [8] is the most complete formal survey of CRDTs and SEC, defining vis/hb-based SEC; however, it does not extend to multi-key transactional isolation or atomic visibility across keys.
- Composition with transactions:
  - [5] supports multi-object transactional atomicity atop an eventually consistent store and formalizes PSI; it explores invariant repair by strengthening consistency, making it a practical bridge between CRDT semantics and transactional isolation.
  - [6] parameterizes CRDT convergence proofs by policy (EC, CC, PSI), showing that stronger policies can relax commu-tativity requirements; this highlights that PSI-level constraints can simplify CRDT reasoning or composition.
  - [9] integrates transactional causal consistency (TCC+) with CRDTs, guaranteeing atomic visibility and convergence; it positions TCC+ as availability-compatible and augments with local SI zones for stronger semantics when coordination is feasible.

Citations: [5,6,8,9].

Applicability to key-value stores and protocols

- Direct KV focus with multi-key transactions:
  - [1] is explicitly a KV transactional semantics, parametric in “execution tests”; it proves COPS satisfies CC and Clock-SI satisfies SI, grounding the theory in KV protocols.
  - [7] targets replicated weakly-consistent transactional databases (KV-like), encoding programs into abstract executions over vis/ar and detecting anomalies under PSI/SI.
  - [2,10] provide KV-applicable axioms (atomic visibility) and program robustness reasoning, respectively.
- Availability/coordination implications:
  - [4] rigorously links arbitration to lack of availability; this informs the design trade-offs for KV systems targeting PSI/SI vs causal/TCC+.
  - [9] operationalizes this trade-off via TCC+ globally and SI within “SI zones.”

Citations: [1,2,4,7,9,10].

Unique contributions germane to the desired review

- New or clarified model relationships:
  - WSI located strictly between PSI and SI, with axiomatic/operational equivalence across definitions [1].
  - PSI ' Prefix Consistency C SI (composition result) [7].

  - “Arbitration-free Ô available” theorem; PSI/PC/CCC require arbitration in exemplars, hence are not fully available [4].
  - CC a the four transactional session guarantees (in the state-based framework) [3].

- End-to-end bridges:
  - Proof of protocol correctness under formal axioms (COPS for CC; Clock-SI for SI) [1].
  - Automated anomaly detection for transactional programs under PSI/SI [7].
  - CRDT convergence verification under EC/CC/PSI policies [6]; bounded-search invariant checking and selective strengthening [5].
  - Availability-compatible transactional model with atomic visibility (TCC+), plus “SI zones” [9].

Citations: [1,3,4,5,6,7,9].

Gaps relative to the target “graduate-level literature review” brief

- No single paper here is a dedicated, integrative literature review that jointly surveys SR/SI/PSI and CRDT/SEC for KV with a comprehensive anomaly-based comparison and implication lattice:
  - The closest unifiers for transactional KV are [1,2,3,7,10], but they are research contributions/frameworks, not survey articles.
  - The only true survey, [8], is CRDT-focused and does not cover multi-key transactional isolation or anomaly catalogs in the transactional sense.
  - Availability vs coordination is formalized in [4], but not presented as a survey of models; PSI/SI are included as exemplars of arbitration-using models.

- Limited explicit mapping between Adya phenomena and CRDT behaviors under transactional composition:
  - While [1,2,3,7,10] handle G0/G1/G2 for transactional models, and [8] formalizes SEC, a systematic cross-walk (e.g., fractured reads under transactional CRDT composition, or rw-cycles with CRDT-per-key merges) is only partially addressed in [5,6,9] and not surveyed comprehensively.

Citations: [1,2,3,4,5,6,7,8,9,10].

Consolidated comparison of model-to-anomaly coverage (transactional KV focus)

Model property	Serializability (SR)	Snapshot Isolation (SI)	Parallel SI (PSI)	TCC+/Causal+ with atomic visibility	CRDT SEC (per key)
Formalization references	[1,2,3,7,10]	[1,2,3,7,10]	[1,3,5,6,7,9]	[3,9]	[6,8]
Requires global arbitration	Typically yes (for strict serializability); SR itself can be axiomatized without rt but often uses ar [1,2,3,7]	No global rt; often uses ar or per-shard commit order [1,3,7]	Avoids global snapshot; per-key conflict and causality closure; may still rely on arbitration per key [1,3,5,7]	No global arbitration; aims for availability; orders concurrent txns for convergence [9]	No arbitration (state-based join or op-based causal delivery) [8]
Disallows G0/G1	Yes [2,7,10]	Yes [1,2,7]	Yes [1,2,7]	Yes (atomic snapshots, causal closure) [9]	N/A (per-object) [8]
Disallows G2 (rw cycles)	Yes [1,2,7]	No (write skew allowed) [1,2,3,7]	No (write skew allowed) [1,3,7]	No (more permissive; maintains atomic visibility) [9]	N/A (outside tx anomaly lens) [8]
Atomic visibility (no fractured reads)	Yes [1,2,7,10]	Yes [1,2,7,10]	Yes (by design in these axiomatizations) [1,2,7]	Yes (explicit) [9]	Per-object only; cross-key requires a transactional layer [8]
Availability under partitions	Not available in general (coordination needed) [4]	Generally coordination-heavy; not “arbitration-free” [4]	Not “arbitration-free” in exemplars [4]; availability-compatible variants exist only by weakening	Designed to be availability-compatible [9]	Yes (SEC) [8]

Citations: [1,2,3,4,7,8,9,10].

Practical signatures and model coverage across the papers

- Write skew (rw-only cycles): admitted by SI/PSI; discussed/analyzed in [1,2,3,7].
- Lost update (ww conflicts): prevented per key by SI/PSI and by CRDT merge semantics; still subtle at app-level across keys; covered in [1,3,5,7,8].
- Fractured reads (non-atomic visibility): excluded by the atomic-visibility family [1,2,7,9,10]; highlighted in TCC+ [9].
- Non-monotonic snapshots/consistent-prefix: analyzed compositionally (PSI ' PC C SI) in [7]; PSI’s per-key conflict checks vs global prefix noted in [1,3,7].

- Session guarantees vs causal consistency: equivalence results in [3]; embedded in causal/PSI axioms in [1,7,9].

Citations: [1,2,3,5,7,8,9,10].

Summary for expert readers

- For a theory-grounded synthesis of KV transactional models using vis/ar/hb and dependency graphs, start with [2] (atomic visibility axioms) and [1] (operational–axiomatic equivalence and WSI), complemented by [7] for program-level anomaly detection and compositional relations (e.g., PSI ' PC C SI). Use [3] to cross-check state-based definitions and composition with session guarantees.
- To situate availability trade-offs, employ [4]'s AFC theorem and [9]'s TCC+ as a constructive design point showing how to stay availability-compatible while providing atomic visibility and causal snapshots.
- For CRDTs and SEC, [8] is the formal baseline; to connect CRDTs with transactional guarantees in KV, leverage [5,6,9].
- Collectively, these papers provide the ingredients for the desired literature review, but no single reference delivers a complete, comparative survey across SR/SI/PSI and CRDT/SEC specifically for KV. Building that synthesis requires integrating [1,2,3,7,10] (transactional theory), [8] (CRDT theory), and [4,9] (availability implications) with the practical bridges in [5,6].

Timeline

Early axiomatic foundations: atomic visibility and abstract executions (2015–2016)

- Axiomatic, history-based foundations for transactional consistency with explicit visibility/arbitration and atomic visibility crystallize in work by Cerone and Gotsman, providing a general framework for models like CC, PSI, SI, and Serializability in terms of abstract executions and dependency graphs [2]. This sets the baseline vocabulary (vis, ar, hb; wr/ww/rw dependencies; Adya-style phenomena) used by later work.
- Follow-on robustness results under atomic visibility further consolidate the methodology, using the same abstract-execution style to reason about when programs are robust against weakening the consistency model [10]. Together, [2,10] anchor the "axiomatic with atomic visibility" line that later papers build upon.

State-based unification and equivalences across consistency/isolation (2016)

- Crooks et al. propose a state-based unified model that defines isolation and consistency as constraints over application-observable states, with explicit visibility and arbitration, and prove equivalences back to classical history-based definitions [3]. They:
  - Show how to compose guarantees and clarify distinctions among models (e.g., relating PSI to PL-2+) [3].
  - Offer anomaly-centric insights (e.g., write skew) in the state-based view while preserving ties to dependency-cycle formulations [3].
- This reframing establishes a parallel (state-sequence) formalism that is equivalent to abstract-execution/graph-based foundations, enabling multiple proof styles for the same models [3].

Robustness and program analysis under atomic visibility (2016–2019)

- The atomic-visibility framework of Cerone–Gotsman spurs two complementary theory directions:
  - Robustness/invariant reasoning for client programs over weak models with atomic visibility [10], later leveraged and generalized in centralized/operational KV semantics [1].
  - Automated static analysis, encoding transactional programs plus weak-model axioms (vis, ar) into SMT to detect serializability violations via dependency cycles; this includes formalizations of PSI, SI, prefix/causal models, and their combinations (e.g., PSI ' PC C SI) [7].
- Outcome: model-agnostic, axiomatic encodings became a practical substrate for tooling that reasons about anomalies across distributed KV-style systems [7].

Tool-supported exploration and verification for CRDTs and transactions (2018–2019)

- Two strands expand formal analysis from transactions to CRDT settings:
  - Bounded-concurrency exploration (Q9) searches for invariant violations over eventually consistent stores, modeling vis/hb and supporting multi-object transactions and a formal PSI axiom; it suggests strengthening consistency selectively when violations are found [5].
  - Automated parameterized verification proves SEC of CRDTs under parametric consistency policies (eventual, causal, PSI), capturing "commutativity modulo policy" and non-interference conditions; it shows how stronger policies can relax per-operation commutativity needs [6].
- Significance: CRDT reasoning is connected to transactional constraints through shared vis/hb/PSI axioms, foreshadowing the need to combine per-key SEC with transactional atomic visibility for multi-key invariants [5,6].

Operational KV semantics with execution-test parameterization; WSI and protocol proofs (2019)

- Xiong et al. introduce a centralized, multi-version KV-store semantics with per-client partial views and a parametric "execution test" that instantiates models including CC, PSI, SI, and Serializability; they prove equivalence to declarative specifications (abstract executions, dependency graphs) and verify real protocols (COPS for CC, Clock-SI for SI) within the framework [1].
- They identify Weak Snapshot Isolation (WSI), positioned strictly between PSI and SI, enriching the implication lattice in the KV context [1].
- This paper bridges operational and axiomatic traditions and grounds theory in protocol-level correctness proofs for distributed KV stores [1].

Transactional Causal+ and availability-friendly models for geo/edge settings (2021)

- Toumlilt et al. formalize Transactional Causal Plus (TCC+), extending Causal+ with transactional atomic visibility, snapshot properties, and convergence guarantees, and position it relative to CC, PSI, and SI; they deploy a hybrid model with SI "zones" and TCC+ globally in an edge-centric system [9].
- They explicitly incorporate CRDTs while preserving rollback-freedom and convergence, emphasizing the costs/benefits trade-off between causality-first transactional models and SI under partitioned/geo settings [9].
- This marks a pragmatic shift: target the strongest availability-compatible transactional model with atomic visibility, reserving SI for tightly connected regions [9].

Consolidation of CRDT theory and SEC (2023)

- Almeida's Computing Surveys article consolidates the formal CRDT landscape, with explicit so/vis/hb, SEC, and causal consistency, clarifying misconceptions and per-object synchronization semantics [8].
- While not transactional, it provides a comprehensive, formal foundation for per-key data types that can be composed with transactional mechanisms in KV systems—reinforcing the orthogonality of SEC to cross-key transactional guarantees [8].

Arbitration-free availability and its implications (2025)

- Attiya et al. formalize a general visibility/arbitration framework for objects and transactions and prove the Arbitration-Free Consistency (AFC) theorem: a spec admits an available implementation iff its visibility formulas avoid reliance on a total arbitration order [4].
- They instantiate models ranging from CC and Prefix Consistency to SI, PSI, and Serializability, and use litmus examples to show the unavailability of models requiring arbitration; this gives a sharp theory result explaining why SI/PSI variants often necessitate coordination, while causality-based models can be AP [4].
- The result synthesizes a decade of vis/ar/hb formalization to deliver a general availability trade-off principle for KV and beyond [4].

Trends and shifts across the decade

- From axioms to operational and back:
  - 2015–2016 establish axiomatic/abstract-execution baselines with atomic visibility [2,10] and a state-based equivalence viewpoint [3].
  - 2019 operationalizes KV semantics with a parametric execution-test interface, yet proves equivalence to axioms and validates protocols [1].
- Tooling maturation:
  - SMT-based anomaly finding and serializability checking for transactional programs under weak models appear by 2018 [7].
  - CRDT verification under parametric consistency and bounded exploration for invariants broaden applicability to mixed CRDT/transactional stacks [5,6].
- Granularity and composition:
  - Increasing emphasis on composing per-key CRDT SEC with cross-key transactional visibility and conflict checks (PSI/SI), highlighting fractured-reads and invariant preservation as first-class concerns [5,6,9].
- Availability-aware modeling:
  - Formal recognition of causality-centric models as availability-compatible, contrasted with arbitration-dependent isolation (SC/SI/PSI variants) culminating in the AFC theorem [4,9].
- Refining the isolation lattice for KV:
  - Clarifications of PSI's place relative to PL-2+ and SI [3,7], and the introduction of WSI between PSI and SI within a KV semantics [1].

Contributor clusters and sustained threads

- Cerone–Gotsman lineage (atomic visibility and abstract executions):
  - Foundations and robustness under atomic visibility; frameworks widely cited and reused in later axiomatic encodings and operational correspondences [2,10,1,7].
- Jagannathan's group (Nagar, Kaki):
  - Program analysis and verification across weak models, including formal PSI/SI constraints, serializability-violation detection, and CRDT convergence under parametric consistency [5,6,7]. This cluster operationalized axioms for automated reasoning at both program and data-type levels.
- State-based unification (Crooks et al.):
  - Alternative but equivalent formalism emphasizing application-visible states; clarified PSI/SI relations and anomaly characterizations [3].
- CRDT theory consolidation (Almeida):
  - Comprehensive survey formalizing SEC and causal consistency with vis/hb, shaping shared terminology for CRDT/KV composition discussions [8].
- Availability trade-offs (Attiya et al.):
  - General AFC theorem linking absence of arbitration to availability, with explicit transactional/KV instantiations including SI/PSI [4].
- Causality-first transactional systems (Shapiro/Toumlilt and collaborators):
  - TCC+ as the strongest availability-compatible transactional model, with explicit placement relative to PSI/SI and attention to CRDT composition in edge settings [9].

Significance and implications for future work

- Toward unified, KV-centered theory reviews: The field now has all key pieces—axiomatic and state-based definitions with atomic visibility; operational KV semantics with equivalence proofs; formal placements and refinements of PSI/SI (including WSI); automated tooling for anomaly discovery; CRDT SEC foundations; and availability limits for arbitration-based models [1,2,3,4,5,6,7,8,9,10]. Yet, a single, graduate-level literature review synthesizing these threads specifically for KV transactional models and CRDT composition remains scarce—an opportunity for the requested comparative and gap analysis.
- Practical theory for design:
  - The execution-test parameterization [1], program-analysis encodings [7], and AFC availability criterion [4] offer actionable guidance for KV store designers choosing between SI/PSI/TCC+ and CRDT compositions.
  - Verification frameworks suggest a path to invariant-preserving design under mixed CRDT/transactional semantics, possibly integrating escrow/reservation techniques atop PSI/TCC+ snapshots [5,6,9].
- Expected trajectory:
  - More precise characterizations of intermediate models (e.g., WSI-like variants) tailored to KV sharding/replication realities [1].
  - Increased emphasis on tool-supported synthesis of "just-enough" consistency for invariants under CRDT+transactions, driven by the parametric-policy verification line [5,6].
  - Continued clarification of the boundary between availability and arbitration as systems seek low-latency guarantees in geo/edge contexts [4,9].

Foundational Work

Which papers form the foundational references on this topic?

The below table shows the resources that are most often cited by the relevant papers on this topic. This is measured by the **reference rate**, which is the fraction of relevant papers that cite a resource. Use this table to determine the most important core papers to be familiar with if you want to deeply understand this topic. Some of these core papers may not be directly relevant to the topic, but provide important context.

Ref.	Reference Rate	Title	Cited By These Relevant Papers
[11]	0.65	Transactional storage for geo-replicated systems	[1, 3, 5, 6, 7, 9, 10]
[61]	0.54	Don't settle for eventual: scalable causal consistency for wide-area storage with COPS	[1, 3, 6, 7, 9, 10, 11]

[13]	0.50	'Cause I'm strong enough: Reasoning about consistency choices in distributed systems	[1, 5, 6, 7, 9, 10]
[24]	0.43	A critique of ANSI SQL isolation levels	[1, 3, 7, 10, 11, 12, 13]
[19]	0.39	Non-monotonic Snapshot Isolation: Scalable and Strong Consistency for Geo-replicated Transactional Systems	[1, 2, 3, 9, 10, 13, 14]
[7]	0.37	Automated Detection of Serializability Violations under Weak Consistency	[1, 6, 20]
[81]	0.33	Alone together: compositional reasoning and inference for weak isolation	[1, 5, 7]
[62]	0.29	Stronger Semantics for Low-Latency Geo-Replicated Storage	[1, 3, 19, 26]
[74]	0.27	Making Geo-Replicated Systems Fast as Possible, Consistent when Necessary	[1, 6, 9, 13]
[63]	0.26	Scalable atomic visibility with RAMP transactions	[1, 7, 10, 13, 26]
[141]	0.25	Seeing is Believing: A Client-Centric Specification of Database Isolation	[1, 7]
[142]	0.25	Generalized isolation level definitions	[1, 7, 12, 17, 20]
[143]	0.24	Algebraic Laws for Weak Consistency	[1, 7]
[144]	0.22	Eventually Consistent Transactions	[1, 7, 10, 13]
[41]	0.22	Wren: Nonblocking Reads in a Partitioned Transactional Causally Consistent Data Store	[1]
[58]	0.22	Consistency in Non-Transactional Distributed Storage Systems	[5, 8, 9, 16, 25]
[145]	0.21	From session causality to causal consistency	[1, 3]
[17]	0.20	On Parallel Snapshot Isolation and Release/Acquire Consistency	[1]
[72]	0.20	ROLA: A New Distributed Transaction Protocol and Its Formal Analysis	[1]
[36]	0.19	Conflict-Free Replicated Data Types	[5, 6, 9, 11, 13]

Adjacent Work

Which papers cite the same foundational papers as relevant papers?

Use this table to discover related papers on adjacent topics, to gain a broader understanding of the field and help generate ideas for useful new research directions.

Ref.	Adjacency score	Title	References These Foundational Papers
[26]	0.20	All in One: Design, Verification, and Implementation of SNOW-optimal Read Atomic Transactions	[11, 19, 61, 63, 68, 89, 108]
[28]	0.13	Distributed transactional reads: the strong, the quick, the fresh & the impossible	[11, 19, 24, 61, 63, 129]
[13]	0.12	'Cause I'm strong enough: Reasoning about consistency choices in distributed systems	[11, 19, 24, 61, 63, 108]
[58]	0.10	Consistency in Non-Transactional Distributed Storage Systems	[13, 61, 63, 108, 115, 129]
[46]	0.10	Parameterized and Runtime-Tunable Snapshot Isolation in Distributed Transactional Key-Value Stores	[11, 19, 24, 61]
[10]	0.10	Robustness against Consistency Models with Atomic Visibility	[11, 19, 24, 61, 63]
[35]	0.10	Totally-Ordered Prefix Parallel Snapshot Isolation	[11, 19, 24, 61]
[152]	0.09	TiQuE: Improving the Transactional Performance of Analytical Systems for True Hybrid Workloads	[11, 19, 24, 61]
[9]	0.09	Highly-available and consistent group collaboration at the edge with colony	[11, 13, 19, 61]
[120]	0.09	Quaestor: Query Web Caching for Database-as-a-Service Providers	[11, 61, 108, 123]
[7]	0.08	Automated Detection of Serializability Violations under Weak Consistency	[11, 13, 24, 61, 63, 108]
[153]	0.08	Viper: A Fast Snapshot Isolation Checker	[1, 11, 19, 24]

[52]	0.07	On Mixing Eventual and Strong Consistency: Acute Cloud Types	[11, 13, 42, 58, 61]
[154]	0.07	Decidable verification under a causally consistent shared memory	[7, 11, 17, 19]
[155]	0.07	MRVs: Enforcing Numeric Invariants in Parallel Updates to Hotspots with Randomized Splitting	[11, 24, 42, 61]
[156]	0.07	NOC-NOC: Towards Performance-optimal Distributed Transactions	[11, 24, 61]
[157]	0.07	Database Consistency Models	[11, 13, 19, 58]
[3]	0.07	Seeing is Believing: A Unified Model for Consistency and Isolation via States	[11, 19, 24, 61, 68, 108]
[66]	0.07	Regular Sequential Serializability and Regular Sequential Consistency	[11, 13, 58, 61]
[6]	0.07	Automated Parameterized Verification of CRDTs	[7, 11, 13, 61]

References

[1] Data Consistency in Transactional Storage Systems: a Centralised Approach
Shale Xiong, ..., and Philippa Gardner. ArXiv, 2019. 5 citations.
66% Topic Match

Proposes a centralised operational semantics for transactional KV stores.
Builds a parametric interleaving model (global multiversion KV store + partial client views) and an “execution test” to capture CC, PSI, SI, serializability, and a new WSI.
Provides formal correspondence proofs to declarative abstract-execution/dependency-graph definitions, proves implication/strictness relations, assumes atomic snapshots and LWW, and verifies protocols (COPS, Clock-SI); does not model non-snapshot isolation levels.

[2] A Framework for Transactional Consistency Models with Atomic Visibility
A. Cerone, ..., and Alexey Gotsman. Unknown journal, 2015. 100 citations.
56% Topic Match

No summary or abstract available

[3] Seeing is Believing: A Unified Model for Consistency and Isolation via States
Natacha Crooks, ..., and Allen Clement. ArXiv, 2016. 3 citations.
45% Topic Match

Proposes a state-based unified framework for consistency and isolation guarantees.
Replaces operation-history constraints with constraints on application-observable states using visibility and arbitration, and proves equivalence to classic history-based definitions.
Relevant: formalizes vis/ar and composes guarantees (shows causal Ô four session guarantees for transactions), characterizes anomalies (write-skew, monotonic reads), maps PSI/PSI-like models (PSI/Parallel SI H lazy/PL-2+), but does not model uncommitted/opacity or forking consistency.

[4] Arbitration-Free Consistency is Available (and Vice Versa)
H. Attiya, ..., and Enrique Rom'an-Calvo. ArXiv, 2025. 0 citations.
43% Topic Match

Shows that a storage spec admits an available implementation iff it is arbitration-free.
Proves an Arbitration-Free Consistency (AFC) theorem in a formal framework using visibility formulas and an explicit arbitration (total) order.
Framework covers key-value stores, CRDTs and transactional models (SI, PSI, SC, serializability), formalizes vis/ar/hb distinctions, gives litmus examples and impossibility results—highly relevant for theory-focused comparison of availability vs. arbitration in KV transactional/CRDT settings.

[5] Safe replication through bounded concurrency verification
Gowtham Kaki, ..., and S. Jagannathan. Proceedings of the ACM on Programming Languages, 2018. 39 citations.
37% Topic Match

Proposes a bounded-concurrency verification framework (Q9) to detect and repair weak-consistency anomalies in replicated data types.
Implements symbolic exploration of executions over an eventually-consistent store using formal vis/hb relations and axiomatic contracts (including a formal PSI and an ATOM atomicity axiom) to generate finite counterexamples and synthesized consistency-strengthening repairs.
Relevant: theory-focused, models vis/hb/ar-like relations and multi-object transactions, discusses CRDT/SEC and PSI formally—good fit for transactional/CRDT comparison and anomaly-based analysis, though emphasis is verification/repair rather than a pure literature review.

[6] Automated Parameterized Verification of CRDTs
Kartik Nagar and S. Jagannathan. ArXiv, 2019. 20 citations.
33% Topic Match

Presents an automated, parameterized verification framework for CRDT convergence (strong eventual consistency).
Encodes event-based execution axioms and commutativity/non interference constraints into SMT-style checks to prove convergence under various consistency policies (eventual, causal, PSI).
Relevant: theory-focused on CRDT/SEC and formal event/happens before models; discusses PSI as an environment assumption but does not cover multi-key transactional vis/ar/anomaly catalogs (Adya) or transactional serializability/SI comparisons.

[7] Automated Detection of Serializability Violations under Weak Consistency
Kartik Nagar and S. Jagannathan. Unknown journal, 2018. 34 citations.
30% Topic Match

Demonstrates an automated, theory-focused method to detect serializability violations under weak consistency.
Encodes transactional programs as abstract executions/dependency graphs and consistency models (vis, ar, causal closure) into FOL, then uses SMT to find dependency-cycle (non-serializable) executions.
Targets replicated weakly-consistent transactional stores (evaluates PSI, SI, etc.), uses Adya-style dependency-cycle characterization, gives bounded search plus inductive sufficient conditions for unbounded reasoning — highly relevant for formal, history/order-based comparisons of SI/PSI/serializability in KV systems.

[8] Approaches to Conflict-free Replicated Data Types
Paulo Sérgio Almeida. ACM Computing Surveys, 2023. 11 citations.
25% Topic Match

Presents an in-depth survey of CRDT approaches and strong eventual consistency (SEC).
Uses formal relations (so, vis, hb), defines causal consistency as vis=hb, and covers op/state/delta/pure-op CRDT designs and convergence properties.
Focused on per-object/replica semantics and SEC; lacks treatment of multi-key transactional models (SR/SI/PSI), Adya phenomena, dependency edges (wr/ww/rw), transactional snapshots/atomic visibility, and cross-key KV gap analysis—so only partly relevant.

[9] Highly-available and consistent group collaboration at the edge with colony
Ilyas Toumlilt, ..., and M. Shapiro. Proceedings of the 22nd International Middleware Conference, 2021. 5 citations.
22% Topic Match

Proposes a formally defined Transactional Causal+ (TCC+) model for edge KV transactions.
Formalizes TCC+ using visibility, arbitration, transactional equivalence (a), atomic visibility, causally+consistent snapshots, and eventual visibility liveness.
Notes relations to CC, Causal+, PSI and SI (SI enforced inside SI-zones), discusses CRDT composition, convergence, metadata costs, and edge/cloud topology — useful if you need formal vis/ar/hb treatment plus practical gap analysis for KV edge deployments.

[10] Robustness against Consistency Models with Atomic Visibility
G. Bernardi and Alexey Gotsman. Unknown journal, 2016. 41 citations.
21% Topic Match

No summary or abstract available

[11] Transactional storage for geo-replicated systems
Yair Sovran, ..., and Jinyang Li. Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, 2011. 412 citations.
18% Topic Match

No summary or abstract available

[12] A critique of snapshot isolation

M. Yabandeh and Daniel Gómez Ferro. ArXiv, 2012. 48 citations.
17% Topic Match

Abstract: The support for transactions is an essential part of a database management system (DBMS). Without this support, the developers are burdened with ensuring atomic execution of a transaction despite failures as well as concurrent accesses to the database by other transactions. Ideally, a transactional system provides serializability, which means that the outcome of concurrent transactions is equivalent to a serial execution of them. Based on experiences on lock-based implementations, nevertheless, serializability is known as an expensive feature that comes with high overhead and low concurrency. Commercial systems, hence, compromise serializability by implementing weaker guarantees such as snapshot isolation. The developers,...

[13] 'Cause I'm strong enough: Reasoning about consistency choices in distributed systems
Alexey Gotsman, ..., and M. Shapiro. Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, 2016. 137 citations.
16% Topic Match

No summary or abstract available

[14] Verifying Transactional Consistency of MongoDB
Hongrong Ouyang, ..., and Anqun Pan. ArXiv, 2021. 5 citations.
15% Topic Match

Abstract: MongoDB is a popular general-purpose, document-oriented, distributed NoSQL database. It supports transactions in three different deployments: single-document transactions utilizing the WiredTiger storage engine in a standalone node, multi-document transactions in a replica set which consists of a primary node and several secondary nodes, and distributed transactions in a sharded cluster which is a group of multiple replica sets, among which data is sharded. A natural and fundamental question about MongoDB transactions is: What transactional consistency guarantee do MongoDB Transactions in each deployment provide? However, it lacks both concise pseudocode of MongoDB transactions in each deployment and formal specification of the...

[15] Serializability for eventual consistency: criterion, analysis, and applications
Lucas Brutschy, ..., and Martin T. Vechev. Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, 2017. 52 citations.
14% Topic Match

No summary or abstract available

[16] A Generic Specification Framework for Weakly Consistent Replicated Data Types
Xue Jiang, ..., and Anqun Pan. IEEE Transactions on Parallel and Distributed Systems, 2025. 0 citations.
13% Topic Match

Abstract: Burckhardt et al. proposed a formal specification framework for eventually consistent replicated data types, denoted <inline-formula><tex-math notation="LaTeX">(vis, ar)</tex-math><alternatives><mml:math><mml:mrow><mml:mo>(</mml:mo><mml:mi>v</mml:mi><mml:mi>i</mml:mi><mml:mi>s</mml:mi><mml:mo>,</mml:mo><mml:mi>a</ xlink:href="wei-ieq1-3533546.gif"/></alternatives></inline-formula>, based on the notions of visibility and arbitration relations. However, being specific to eventually consistent systems, this framework has two limitations. First, it does not cover non-convergent consistency models since arbitration <inline-formula><tex-math notation="LaTeX">ar</tex-math><alternatives><mml:math><mml:mrow><mml:mi>a</mml:mi><mml:mi>r</mml:mi></mml:mrow></mml:math><inline-graphic xlink:href="wei-ieq2-3533546.gif"/></alternatives></inline-formula> is a total order over events. Second, it does not cover the consistency models in which each event is required to be aware of the return values of some events that are visible to it when justifying its return value. These limitations make the <inline-formula><tex-math notation="LaTeX">(vis, ar)</tex-math><alternatives><mml:math><mml:mrow><mml:mo>(</mml:mo><mml:mi>v</mml:mi><mml:mi>i</mml:mi><mml:mi>s</mml:mi><mml:mo>,</mml:mo><mml:mi>a</

[17] On Parallel Snapshot Isolation and Release/Acquire Consistency
Azalea Raad, ..., and Viktor Vafeiadis. Unknown journal, 2018. 18 citations.
13% Topic Match

Abstract: Parallel snapshot isolation (PSI) is a standard transactional consistency model used in databases and distributed systems. We argue that PSI is also a useful formal model for software transactional memory (STM) as it has certain advantages over other consistency models. However, the formal PSI definition is given declaratively by acyclicity axioms, which most programmers find hard to understand and reason about.

[18] On the complexity of checking transactional consistency
Ranadeep Biswas and C. Enea. Proceedings of the ACM on Programming Languages, 2019. 46 citations.
12% Topic Match

Abstract: Transactions simplify concurrent programming by enabling computations on shared data that are isolated from other concurrent computations and are resilient to failures. Modern databases provide different consistency models for transactions corresponding to different tradeoffs between consistency and availability. In this work, we investigate the problem of checking whether a given execution of a transactional database adheres to some consistency model. We show that consistency models like read committed, read atomic, and causal consistency are polynomial-time checkable while prefix consistency and snapshot isolation are NP-complete in general. These results complement a previous NP-completeness result concerning serializability. Moreover, in the context of...

[19] Non-monotonic Snapshot Isolation: Scalable and Strong Consistency for Geo-replicated Transactional Systems
Masoud Saeida Ardekani, ..., and M. Shapiro. 2013 IEEE 32nd International Symposium on Reliable Distributed Systems, 2013. 81 citations.
11% Topic Match

No summary or abstract available

[20] Dynamic Partial Order Reduction for Checking Correctness against Transaction Isolation Levels
A. Bouajjani, ..., and Enrique Rom'an-Calvo. Proceedings of the ACM on Programming Languages, 2023. 4 citations.
10% Topic Match

Abstract: Modern applications, such as social networking systems and e-commerce platforms are centered around using large-scale databases for storing and retrieving data. Accesses to the database are typically enclosed in transactions that allow computations on shared data to be isolated from other concurrent computations and resilient to failures. Modern databases trade isolation for performance. The weaker the isolation level is, the more behaviors a database is allowed to exhibit and it is up to the developer to ensure that their application can tolerate those behaviors. In this work, we propose stateless model checking algorithms for studying correctness of such applications that...

[21] Distributed snapshot isolation: global transactions pay globally, local transactions pay locally
Carsten Binnig, ..., and Norman May. The VLDB Journal, 2014. 37 citations.
10% Topic Match

No summary or abstract available

[22] Static serializability analysis for causal consistency
Lucas Brutschy, ..., and Martin T. Vechev. Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, 2018. 29 citations.
9% Topic Match

Abstract: Many distributed databases provide only weak consistency guarantees to reduce synchronization overhead and remain available under network partitions. However, this leads to behaviors not possible under stronger guarantees. Such behaviors can easily defy programmer intuition and lead to errors that are notoriously hard to detect. In this paper, we propose a static analysis for detecting non-serializable behaviors of applications running on top of causally-consistent databases. Our technique is based on a novel, local serializability criterion and combines a generalization of graph-based techniques from the database literature with another, complementary analysis technique that encodes our serializability criterion into first-order logic formulas...

[23] UniStore: A fault-tolerant marriage of causal and strong consistency (extended version)
Manuel Bravo, ..., and Hengfeng Wei. ArXiv, 2021. 3 citations.
9% Topic Match

Abstract: Modern online services rely on data stores that replicate their data across geographically distributed data centers. Providing strong consistency in such data stores results in high latencies and makes the system vulnerable to network partitions. The alternative of relaxing consistency violates crucial correctness properties. A compromise is to allow multiple consistency levels to coexist in the data store. In this paper we present UniStore, the first fault-tolerant and scalable data store that combines causal and strong consistency. The key challenge we address in UniStore is to maintain liveness despite data center failures: this could be compromised if a strong transaction...

[24] A critique of ANSI SQL isolation levels
Hal Berenson, ..., and P. O'Neil. Unknown journal, 1995. 1175 citations.
8% Topic Match

Abstract: ANSI SQL-92 [MS, ANSI] defines Isolation Levels in terms of phenomena: Dirty Reads, Non-Repeatable Reads, and Phantoms. This paper shows that these phenomena and the ANSI SQL definitions fail to properly characterize several popular isolation levels, including the standard locking implementations of the levels covered. Ambiguity in the statement of the phenomena is investigated and a more formal statement is arrived at; in addition new phenomena that better characterize isolation types are introduced. Finally, an important multiversion isolation type, called Snapshot Isolation, is defined.

[25] A Generic Specification Framework for Weakly Consistent Replicated Data Types
Xue Jiang, ..., and Yu Huang. 2020 International Symposium on Reliable Distributed Systems (SRDS), 2020. 4 citations.
8% Topic Match

Abstract: Recently Burckhardt et al. proposed a formal specification framework for eventually consistent replicated data types, denoted (vis, ar), based on the notions of visibility and arbitration relations. However, being specific to eventually consistent systems, this framework has two limitations. First, it does not cover non-convergent consistency models since arbitration ar is defined to be a total order over events in a computation. Second, it does not cover the consistency models in which each event is required to be aware of the return values of some or all events that are visible to it.In this paper, we extend the (vis, ar)...

[26] All in One: Design, Verification, and Implementation of SNOW-optimal Read Atomic Transactions
Si Liu. ACM Transactions on Software Engineering and Methodology (TOSEM), 2022. 6 citations.
8% Topic Match

Abstract: Distributed read atomic transactions are important building blocks of modern cloud databases that magnificently bridge the gap between data availability and strong data consistency. The performance of their transactional reads is particularly critical to the overall system performance, as many real-world database workloads are dominated by reads. Following the SNOW design principle for optimal reads, we develop LORA, a novel SNOW-optimal algorithm for distributed read atomic transactions. LORA completes its reads in exactly one round trip, even in the presence of conflicting writes, without imposing additional overhead to the communication, and it outperforms the state-of-the-art read atomic algorithms. To guide...

[27] Weak Consistency: A Generalized Theory and Optimistic Implementations for Distributed Transactions
A. Adya and B. Liskov. Unknown journal, 1999. 285 citations.
7% Topic Match

No summary or abstract available

[28] Distributed transactional reads: the strong, the quick, the fresh & the impossible
Alejandro Z. Tomsic, ..., and M. Shapiro. Proceedings of the 19th International Middleware Conference, 2018. 12 citations.
7% Topic Match

Abstract: This paper studies the costs and trade-offs of providing transactional consistent reads in a distributed storage system. We identify the following dimensions: read consistency, read delay (latency), and data freshness. We show that there is a three-way trade-off between them, which can be summarised as follows: (i) it is not possible to ensure at the same time order-preserving (e.g., causally-consistent) or atomic reads, Minimal Delay, and maximal freshness; thus, reading data that is the most fresh without delay is possible only in a weakly-isolated mode; (ii) to ensure atomic or order-preserving reads at Minimal Delay imposes to read data from...

[29] Conflict-free Replicated Data Types (CRDTs)
Nuno M. Preguiça, ..., and M. Shapiro. ArXiv, 2018. 52 citations.
6% Topic Match

Abstract: A conflict-free replicated data type (CRDT) is an abstract data type, with a well defined interface, designed to be replicated at multiple processes and exhibiting the following properties: (1) any replica can be modified without coordinating with another replicas; (2) when any two replicas have received the same set of updates, they reach the same state, deterministically, by adopting mathematically sound rules to guarantee state convergence.

[30] Automatic Analysis of Consistency Properties of Distributed Transaction Systems in Maude
Si Liu, ..., and J. Meseguer. Unknown journal, 2019. 17 citations.
6% Topic Match

Abstract: Many transaction systems distribute, partition, and replicate their data for scalability, availability, and fault tolerance. However, observing and maintaining strong consistency of distributed and partially replicated data leads to high transaction latencies. Since different applications require different consistency guarantees, there is a plethora of consistency properties—from weak ones such as read atomicity through various forms of snapshot isolation to stronger serializability properties—and distributed transaction systems (DTSs) guaranteeing such properties. This paper presents a general framework for formally specifying a DTS in Maude, and formalizes in Maude nine common consistency properties for DTSs so defined. Furthermore, we provide a fully automated...

[31] Making consistency more consistent: a unified model for coherence, consistency and isolation
Adriana Szekeres and Irene Zhang. Unknown journal, 2018. 9 citations.
6% Topic Match

No summary or abstract available

[32] Models for Storage in Database Backends
Edgard Schiebelbein, ..., and Marc Shapiro. Proceedings of the 11th Workshop on Principles and Practice of Consistency for Distributed Data, 2024. 0 citations.
5% Topic Match

Abstract: This paper describes ongoing work on developing a formal specification of a database backend. We present the formalisation of the expected behaviour of a basic transactional system that calls into a simple store API, and instantiate in two semantic models. The first is a map-based, classical versioned key-value store; the second, journal-based, appends individual transaction effects to a journal. We formalise a significant part of the specification in the Coq proof assistant. This work will be the basis for formalising a full-fledged backend store with features such as caching or write-ahead logging as variations on maps and journals.

[33] MonkeyDB: effectively testing correctness under weak isolation levels
Ranadeep Biswas, ..., and A. Lal. Proceedings of the ACM on Programming Languages, 2021. 17 citations.
5% Topic Match

Abstract: Modern applications, such as social networking systems and e-commerce platforms are centered around using large-scale storage systems for storing and retrieving data. In the presence of concurrent accesses, these storage systems trade off isolation for performance. The weaker the isolation level, the more behaviors a storage system is allowed to exhibit and it is up to the developer to ensure that their application can tolerate those behaviors. However, these weak behaviors only occur rarely in practice and outside the control of the application, making it difficult for developers to test the robustness of their code against weak isolation levels. This...

[34] A More Consistent Understanding of Consistency
Subhajit Sidhanta, ..., and R. Rodrigues. 2019 38th Symposium on Reliable Distributed Systems (SRDS), 2019. 0 citations.
5% Topic Match

Abstract: Recent storage systems trade strong consistency for performance, availability, and scalability. However, this makes it hard to understand the semantics that the storage system provides, and also makes the design and implementation of the storage system itself more error-prone. This paper proposes a comprehensive solution to these problems. In particular, we propose a specification language named ConSpec, which enables the formalization of different consistency semantics

that a storage system may provide, using a uniform syntax that is independent of the design and implementation of the target storage system. We use ConSpec to revisit several existing models in light of a...

[35] Totally-Ordered Prefix Parallel Snapshot Isolation
Nuno Faria and J. Pereira. Proceedings of the 8th Workshop on Principles and Practice of Consistency for Distributed Data, 2021. 3 citations.
5% Topic Match

Abstract: Distributed data management systems have increasingly been using variants of Snapshot Isolation (SI) as their transactional isolation criteria as it combines strong ACID guarantees with non-blocking reads and scalability. However, most existing proposals are limited by the performance of update propagation and stability detection, in particular, when execution and storage are disaggregated. In this paper, we propose TOPSI, an approach providing a restricted form of Parallel Snapshot Isolation (PSI) that allows partially ordering recent transactions to avoid waiting for remote updates or using a stale snapshot. Moreover, it has the interesting property of making a prefix of history in all...

[36] Conflict-Free Replicated Data Types
M. Shapiro, ..., and M. Zawirski. Unknown journal, 2011. 909 citations.
4% Topic Match

No summary or abstract available

[37] Verifying strong eventual consistency in distributed systems
Victor B. F. Gomes, ..., and A. Beresford. Proceedings of the ACM on Programming Languages, 2017. 79 citations.
4% Topic Match

Abstract: Data replication is used in distributed systems to maintain up-to-date copies of shared data across multiple computers in a network. However, despite decades of research, algorithms for achieving consistency in replicated systems are still poorly understood. Indeed, many published algorithms have later been shown to be incorrect, even some that were accompanied by supposed mechanised proofs of correctness. In this work, we focus on the correctness of Conflict-free Replicated Data Types (CRDTs), a class of algorithm that provides strong eventual consistency guarantees for replicated data. We develop a modular and reusable framework in the Isabelle/HOL interactive proof assistant for verifying...

[38] Scaling Out Acid Applications with Operation Partitioning
Habib Saissi, ..., and N. Suri. ArXiv, 2018. 0 citations.
4% Topic Match

Abstract: OLTP applications with high workloads that cannot be served by a single server need to scale out to multiple servers. Typically, scaling out entails assigning a different partition of the application state to each server. But data partitioning is at odds with preserving the strong consistency guarantees of ACID transactions, a fundamental building block of many OLTP applications. The more we scale out and spread data across multiple servers, the more frequent distributed transactions accessing data at different servers will be. With a large number of servers, the high cost of distributed transactions makes scaling out ineffective or even detrimental....

[39] Reasoning about Weak Isolation Levels in Separation Logic
Anders Alnor Mathiasen, ..., and Lars Birkedal. Proceedings of the ACM on Programming Languages, 2025. 0 citations.
4% Topic Match

Abstract: Consistency guarantees among concurrently executing transactions in local- and distributed systems, commonly referred to as isolation levels, have been formalized in a number of models. Thus far, no model can reason about executable implementations of databases or local transaction libraries providing weak isolation levels. Weak isolation levels are characterized by being highly concurrent and, unlike their stronger counterpart serializability, they are not equivalent to the consistency guarantees provided by a transaction library implemented using a global lock. Industrial-strength databases almost exclusively implement weak isolation levels as their default level. This calls for formalism as numerous bugs violating isolation have been...

[40] Consistency in Distributed Data Stores
Mohammad Roohitavaf. ArXiv, 2016. 0 citations.
3% Topic Match

Abstract: This paper focuses on the problem of consistency in distributed data stores.We define strong consistency model which provides a simple semantics for application programmers, but impossible to achieve with availability and partition-tolerance. We also define weaker consistency models including causal and eventual consistency. We review COPS and GentleRain as two causally consistent data stores as well as Dynamo as an eventually consistent data store. We provide insights about scenarios where each of these methods is suitable, and some future research directions.

[41] Wren: Nonblocking Reads in a Partitioned Transactional Causally Consistent Data Store
Kristina Spirovska, ..., and W. Zwaenepoel. 2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2018. 30 citations.
3% Topic Match

Abstract: Transactional Causal Consistency (TCC) extends causal consistency, the strongest consistency model compatible with availability, with interactive read-write transactions, and is therefore particularly appealing for geo-replicated platforms. This paper presents Wren, the first TCC system that at the same time i) implements nonblocking read operations, thereby achieving low latency, and ii) allows an application to efficiently scale out within a replication site by sharding. Wren introduces new protocols for transaction execution, dependency tracking and stabilization. The transaction protocol supports nonblocking reads by providing a transaction with a snapshot that is the union of a fresh causal snapshot S installed by every...

[42] A comprehensive study of Convergent and Commutative Replicated Data Types
M. Shapiro, ..., and M. Zawirski. Unknown journal, 2011. 429 citations.
3% Topic Match

No summary or abstract available

[43] Robustness Against Transactional Causal Consistency
Sidi Mohamed Beillahi, ..., and C. Enea. Unknown journal, 2019. 29 citations.
3% Topic Match

Abstract: Distributed storage systems and databases are widely used by various types of applications. Transactional access to these storage systems is an important abstraction allowing application programmers to consider blocks of actions (i.e., transactions) as executing atomically. For performance reasons, the consistency models implemented by modern databases are weaker than the standard serializability model, which corresponds to the atomicity abstraction of transactions executing over a sequentially consistent memory. Causal consistency for instance is one such model that is widely used in practice. In this paper, we investigate application-specific relationships between several variations of causal consistency and we address the issue of...

[44] Replication-aware linearizability
C. Enea, ..., and Chao Wang. Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, 2019. 19 citations.
3% Topic Match

Abstract: Distributed systems often replicate data at multiple locations to achieve availability despite network partitions. These systems accept updates at any replica and propagate them asynchronously to every other replica. Conflict-Free Replicated Data Types (CRDTs) provide a principled approach to the problem of ensuring that replicas are eventually consistent despite the asynchronous delivery of updates. We address the problem of specifying and verifying CRDTs, introducing a new correctness criterion called Replication-Aware Linearizability. This criterion is inspired by linearizability, the de-facto correctness criterion for (shared-memory) concurrent data structures. We argue that this criterion is both simple to understand, and it fits most...

[45] CAP Theorem: Revision of Its Related Consistency Models
F. D. Muñoz-Escoí, ..., and J. Bernabéu-Aubán. Comput. J., 2019. 12 citations.
3% Topic Match

Abstract: The CAP theorem states that only two of these properties can be simultaneously guaranteed in a distributed service: (i) consistency, (ii) availability, and (iii) network partition tolerance. This theorem was stated and proved assuming that “consistency” refers to atomic consistency. However, multiple consistency models exist and atomic consistency is located at the strongest edge of that spectrum. Many distributed services deployed in cloud platforms should be highly available

and scalable. Network partitions may arise in those deployments and should be tolerated. One way of dealing with CAP constraints consists in relaxing consistency. Therefore, it is interesting to explore the set...

[46] Parameterized and Runtime-Tunable Snapshot Isolation in Distributed Transactional Key-Value Stores
Hengfeng Wei, ..., and Jian Lu. 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS), 2017. 1 citations.
2% Topic Match

No summary or abstract available

[47] Causal Consistency and Latency Optimality: Friend or Foe?
Diego Didona, ..., and W. Zwaenepoel. ArXiv, 2018. 33 citations.
2% Topic Match

Abstract: Causal consistency is an attractive consistency model for geo-replicated data stores. It is provably the strongest model that tolerates network partitions. It avoids the long latencies associated with strong consistency, and, especially when using read-only transactions (ROTs), it prevents many of the anomalies of weaker consistency models. Recent work has shown that causal consistency allows "latency-optimal" ROTs, that are nonblocking, single-round and single-version in terms of communication. On the surface, this latency optimality is very appealing, as the vast majority of applications are assumed to have read-dominated workloads. In this paper, we show that such "latency-optimal" ROTs induce an extra...

[48] Designing a commutative replicated data type
M. Shapiro and Nuno M. Preguiça. ArXiv, 2007. 55 citations.
2% Topic Match

Abstract: Commuting operations greatly simplify consistency in distributed systems. This paper focuses on designing for commutativity, a topic neglected previously. We show that the replicas of any data type for which concurrent operations commute converges to a correct value, under some simple and standard assumptions. We also show that such a data type supports transactions with very low cost. We identify a number of approaches and techniques to ensure commutativity. We re-use some existing ideas (non-destructive updates coupled with invariant identification), but propose a much more efficient implementation. Furthermore, we propose a new technique, background consensus. We illustrate these ideas with...

[49] From causality to stability: understanding and reducing meta-data in CRDTs
Jim Bauwens and E. G. Boix. Proceedings of the 17th International Conference on Managed Programming Languages and Runtimes, 2020. 3 citations.
2% Topic Match

Abstract: Modern distributed applications increasingly replicate data to guarantee both high availability of systems and optimal user experience. Conflict-Free Replicated Data Types (CRDTs) are a family of data types specially designed for highly available systems that guarantee some form of eventual consistency. To ensure state convergence between replicas, CRDT implementations need to keep track of additional meta-data. This is not a scalable strategy, as a growing amount of meta-data has to be kept. In this paper, we show that existing solutions for this problem miss optimisation opportunities and may lead to less reactive CRDTs. For this, we analyse the relation between...

[50] MonkeyDB: Effectively Testing Correctness against Weak Isolation Levels
Ranadeep Biswas, ..., and A. Lal. ArXiv, 2021. 1 citations.
2% Topic Match

Abstract: Modern applications, such as social networking systems and e-commerce platforms are centered around using large-scale storage systems for storing and retrieving data. In the presence of concurrent accesses, these storage systems trade off isolation for performance. The weaker the isolation level, the more behaviors a storage system is allowed to exhibit and it is up to the developer to ensure that their application can tolerate those behaviors. However, these weak behaviors only occur rarely in practice, that too outside the control of the application, making it difficult for developers to test the robustness of their code against weak isolation levels....

[51] AIDA-DB: A Data Management Architecture for the Edge and Cloud Continuum
Nuno Faria, ..., and Fábio Coelho. 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), 2022. 3 citations.
2% Topic Match

Abstract: There is an increasing demand for stateful edge computing for both complex Virtual Network Functions (VNFs) and application services in emerging 5G networks. Managing a mutable persistent state in the edge does however bring new architectural, performance, and dependability challenges. Not only it has to be integrated with existing cloud-based systems, but also cope with both operational and analytical workloads and be compatible with a variety of SQL and NoSQL database management systems. We address these challenges with AIDA-DB, a polyglot data management architecture for the edge and cloud continuum. It leverages recent development in distributed transaction processing for a...

[52] On Mixing Eventual and Strong Consistency: Acute Cloud Types
Maciej KokociDski, ..., and PaweB T. Wojciechowski. IEEE Transactions on Parallel and Distributed Systems, 2021. 4 citations.
2% Topic Match

Abstract: In this article we study the properties of distributed systems that mix eventual and strong consistency. We formalize such systems through acute cloud types (ACTs), abstractions similar to conflict-free replicated data types (CRDTs), which by default work in a highly available, eventually consistent fashion, but which also feature strongly consistent operations for tasks which require global agreement. Unlike other mixed-consistency solutions, ACTs can rely on efficient quorum-based protocols, such as Paxos. Hence, ACTs gracefully tolerate machine and network failures also for the strongly consistent operations. We formally study ACTs and demonstrate phenomena which are neither present in purely eventually consistent...

[53] Optimistic Causal Consistency for Geo-Replicated Key-Value Stores
Kristina Spirovska, ..., and W. Zwaenepoel. IEEE Transactions on Parallel and Distributed Systems, 2021. 3 citations.
2% Topic Match

Abstract: Causal consistency (CC) is an attractive consistency model for geo-replicated data stores because it hits a sweet spot in the ease-of-programming versus performance trade-off. We present a new approach for implementing CC in geo-replicated data stores, which we call Optimistic Causal Consistency (OCC). OCC's main design goal is to maximize data freshness. The optimism in our approach lies in the fact that the updates replicated to a remote data center are made visible immediately, without checking if their causal dependencies have been received. Servers perform the dependency check needed to enforce CC only upon serving a client operation, rather than...

[54] Bolt-on causal consistency
Peter Bailis, ..., and Ion Stoica. Unknown journal, 2013. 238 citations.
1% Topic Match

No summary or abstract available

[55] Rethinking safe consistency in distributed object-oriented programming
M. Köhler, ..., and G. Salvaneschi. Proceedings of the ACM on Programming Languages, 2020. 13 citations.
1% Topic Match

Abstract: Large scale distributed systems require to embrace the trade off between consistency and availability, accepting lower levels of consistency to guarantee higher availability. Existing programming languages are, however, agnostic to this compromise, resulting in consistency guarantees that are the same for the whole application and are implicitly adopted from the middleware or hardcoded in configuration files. In this paper, we propose to integrate availability in the design of an object-oriented language, allowing developers to specify different consistency and isolation constraints in the same application at the granularity of single objects. We investigate how availability levels interact with object structure and...

[56] Robustness Against Read Committed: A Free Transactional Lunch
Brecht Vandevoort, ..., and F. Neven. Proceedings of the 41st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2022. 3 citations.
1% Topic Match

Abstract: Transaction processing is a central part of most database applications. While serializability remains the gold standard for desirable transactional semantics, many database systems offer improved transaction throughput at the expense of introducing potential anomalies through the choice of a lower isolation level. Transactions are often not arbitrary but are constrained by a set of transaction programs defined at the application level (as is the case for TPC-C for instance), implying that not every potential anomaly can effectively be realized. The question central to this paper is the following: when - within the context of specific transaction programs - do isolation...

[57] CLOTHO: directed test generation for weakly consistent database systems
Kia Rahmani, ..., and S. Jagannathan. Proceedings of the ACM on Programming Languages, 2019. 20 citations.
1% Topic Match

Abstract: Relational database applications are notoriously difficult to test and debug. Concurrent execution of database transactions may violate complex structural invariants that constraint how changes to the contents of one (shared) table affect the contents of another. Simplifying the underlying concurrency model is one way to ameliorate the difficulty of understanding how concurrent accesses and updates can affect database state with respect to these sophisticated properties. Enforcing serializable execution of all transactions achieves this simplification, but it comes at a significant price in performance, especially at scale, where database state is often replicated to improve latency and availability. To address these...

[58] Consistency in Non-Transactional Distributed Storage Systems
P. Viotti and M. Vukolic. ACM Computing Surveys (CSUR), 2015. 140 citations.
1% Topic Match

Abstract: Over the years, different meanings have been associated with the word consistency in the distributed systems community. While in the '80s "consistency" typically meant strong consistency, later defined also as linearizability, in recent years, with the advent of highly available and scalable systems, the notion of "consistency" has been at the same time both weakened and blurred. In this article, we aim to fill the void in the literature by providing a structured and comprehensive overview of different consistency notions that appeared in distributed systems, and in particular storage systems research, in the last four decades. We overview more than...

[59] Efficient Synchronization of State-Based CRDTs
Vitor Enes, ..., and J. Leitao. 2019 IEEE 35th International Conference on Data Engineering (ICDE), 2018. 39 citations.
1% Topic Match

Abstract: To ensure high availability in large scale distributed systems, Conflict-free Replicated Data Types (CRDTs) relax consistency by allowing immediate query and update operations at the local replica, with no need for remote synchronization. State-based CRDTs synchronize replicas by periodically sending their full state to other replicas, which can become extremely costly as the CRDT state grows. Delta-based CRDTs address this problem by producing small incremental states (deltas) to be used in synchronization instead of the full state. However, current synchronization algorithms for delta-based CRDTs induce redundant wasteful delta propagation, performing worse than expected, and surprisingly, no better than state-based. In...

[60] SCAR: Strong Consistency using Asynchronous Replication with Minimal Coordination
Yi Lu, ..., and S. Madden. ArXiv, 2019. 0 citations.
1% Topic Match

Abstract: Data replication is crucial in modern distributed systems as a means to provide high availability. Many techniques have been proposed to utilize replicas to improve a system's performance, often requiring expensive coordination or sacrificing consistency. In this paper, we present SCAR, a new distributed and replicated in-memory database that allows serializable transactions to read from backup replicas with minimal coordination. SCAR works by assigning logical timestamps to database records so that a transaction can safely read from a backup replica without coordinating with the primary replica, because the records cannot be changed up to a certain logical time. In addition,...

[61] Don't settle for eventual: scalable causal consistency for wide-area storage with COPS
Wyatt Lloyd, ..., and D. Andersen. Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, 2011. 690 citations.
1% Topic Match

No summary or abstract available

[62] Stronger Semantics for Low-Latency Geo-Replicated Storage
Wyatt Lloyd, ..., and D. Andersen. Unknown journal, 2013. 300 citations.
1% Topic Match

No summary or abstract available

[63] Scalable atomic visibility with RAMP transactions
Peter Bailis, ..., and Ion Stoica. Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, 2014. 138 citations.
1% Topic Match

No summary or abstract available

[64] Epoch-based Commit and Replication in Distributed OLTP Databases
Yi Lu, ..., and S. Madden. Proc. VLDB Endow., 2021. 49 citations.
1% Topic Match

Abstract: Many modern data-oriented applications are built on top of distributed OLTP databases for both scalability and high availability. Such distributed databases enforce atomicity, durability, and consistency through two-phase commit (2PC) and synchronous replication at the granularity of every single transaction. In this paper, we present COCO, a new distributed OLTP database that supports epoch-based commit and replication. The key idea behind COCO is that it separates transactions into epochs and treats a whole epoch of transactions as the commit unit. In this way, the overhead of 2PC and synchronous replication is significantly reduced. We support two variants of optimistic...

[65] Abstraction for conflict-free replicated data types
Hongjin Liang and Xinyu Feng. Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, 2021. 10 citations.
1% Topic Match

Abstract: Strong eventual consistency (SEC) has been used as a classic notion of correctness for Conflict-Free Replicated Data Types (CRDTs). However, it does not give proper abstractions of functionality, thus is not helpful for modular verification of client programs using CRDTs. We propose a new correctness formulation for CRDTs, called Abstract Converging Consistency (ACC), to specify both data consistency and functional correctness. ACC gives abstract atomic specifications (as an abstraction) to CRDT operations, and establishes consistency between the concrete execution traces and the execution using the abstract atomic operations. The abstraction allows us to verify the CRDT implementation and its client...

[66] Regular Sequential Serializability and Regular Sequential Consistency
J. Helt, ..., and Wyatt Lloyd. Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles, 2021. 12 citations.
0% Topic Match

Abstract: Strictly serializable (linearizable) services appear to execute transactions (operations) sequentially, in an order consistent with real time. This restricts a transaction's (operation's) possible return values and in turn, simplifies application programming. In exchange, strictly serializable (linearizable) services perform worse than those with weaker consistency. But switching to such services can break applications. This work introduces two new consistency models to ease this trade-off: regular sequential serializability (RSS) and regular sequential consistency (RSC). They are just as strong for applications: we prove any application invariant that holds when using a strictly serializable (linearizable) service also holds when using an RSS (RSC)...

[67] Clock-SI: Snapshot Isolation for Partitioned Data Stores Using Loosely Synchronized Clocks
Jiaqing Du, ..., and W. Zwaenepoel. 2013 IEEE 32nd International Symposium on Reliable Distributed Systems, 2013. 75 citations.
0% Topic Match

No summary or abstract available

[68] Building consistent transactions with inconsistent replication

Irene Zhang, ..., and Dan R. K. Ports. Proceedings of the 25th Symposium on Operating Systems Principles, 2015. 211 citations.
0% Topic Match

No summary or abstract available

[69] Karma: Cost-Effective Geo-Replicated Cloud Storage with Dynamic Enforcement of Causal Consistency
Tariq Mahmood, ..., and Mithuna Thottethodi. IEEE Transactions on Cloud Computing, 2021. 15 citations.
0% Topic Match

Abstract: Causal consistency has emerged as an attractive middle-ground to architecting cloud storage systems, as it allows for high availability and low latency, while supporting semantics stronger than eventual consistency. However, causally-consistent cloud storage systems have seen limited deployment in practice. A key factor is these systems employ full replication of all the data in all the data centers (DCs), incurring high cost. A simple extension of current causal systems to support partial replication by clustering DCs into rings incurs availability and latency problems. We propose Karma, the first system to enable causal consistency for partitioned data stores while achieving the...

[70] Tunable Causal Consistency: Specification and Implementation
Xue Jiang, ..., and Yu Huang. 2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS), 2022. 3 citations.
0% Topic Match

Abstract: To achieve high availability and low latency, dis-tributed data stores often geographically replicate data at multiple sites called replicas. However, this introduces the data consistency problem. Due to the fundamental tradeoffs among consistency, availability, and latency in the presence of network partition, no a one-size-fits-all consistency model exists. To meet the needs of different applications, many popular data stores provide tunable consistency, allowing clients to specify the consistency level per individual operation. In this paper, we propose tunable causal consistency (TCC). It allows clients to choose the desired session guarantee for each operation, from the well-known four session guarantees, i.e.,...

[71] A Unified, Practical, and Understandable Model of Non-transactional Consistency Levels in Distributed Replication
Guanzhou Hu, ..., and Remzi H. Arpaci-Dusseau. ArXiv, 2024. 1 citations.
0% Topic Match

Abstract: We present a practical model of non-transactional consistency levels in the context of distributed data replication. Unlike prior work, our simple Shared Object Pool (SOP) model defines common consistency levels in a unified framework centered around the single concept of ordering. This naturally reflects modern cloud object storage services and is thus easy to understand. We show that a consistency level can be intuitively defined by specifying two types of constraints on the validity of orderings allowed by the level: convergence, which bounds the lineage shape of the ordering, and relationship, which bounds the relative positions between operations. We give...

[72] ROLA: A New Distributed Transaction Protocol and Its Formal Analysis
Si Liu, ..., and J. Meseguer. Unknown journal, 2018. 12 citations.
0% Topic Match

Abstract: Designers of distributed database systems face the choice between stronger consistency guarantees and better performance. A number of applications only require read atomicity (RA) and prevention of lost updates (PLU). Existing distributed database systems that meet these requirements also provide additional stronger consistency guarantees (such as causal consistency), and therefore incur lower performance. In this paper we define a new distributed transaction protocol, ROLA, that targets applications where only RA and PLU are needed. We formally model ROLA in Maude. We then perform model checking to analyze both the correctness and the performance of ROLA. For correctness, we use standard...

[73] Causal consistency: beyond memory
Matthieu Perrin, ..., and C. Jard. Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, 2016. 41 citations.
0% Topic Match

Abstract: In distributed systems where strong consistency is costly when not impossible, causal consistency provides a valuable abstraction to represent program executions as partial orders. In addition to the sequential program order of each computing entity, causal order also contains the semantic links between the events that affect the shared objects -- messages emission and reception in a communication channel, reads and writes on a shared register. Usual approaches based on semantic links are very difficult to adapt to other data types such as queues or counters because they require a specific analysis of causal dependencies for each data type. This...

[74] Making Geo-Replicated Systems Fast as Possible, Consistent when Necessary
Cheng Li, ..., and R. Rodrigues. Unknown journal, 2012. 355 citations.
0% Topic Match

No summary or abstract available

[75] Performance-Optimal Read-Only Transactions
Haonan Lu, ..., and Wyatt Lloyd. Unknown journal, 2020. 28 citations.
0% Topic Match

No summary or abstract available

[76] Large-Scale Causal Data Replication for Stateful Edge Applications
Pedro Fouto, ..., and João Leitão. 2024 IEEE 44th International Conference on Distributed Computing Systems (ICDCS), 2024. 1 citations.
0% Topic Match

Abstract: Edge computing is becoming an increasingly popular paradigm, with modern Internet services leveraging hundreds of edge locations to serve their users. However, existing data replication solutions are not designed to operate in this environment, which restricts the edge components of Internet services to operate as read-only caches and entry points for accessing data centers, severely limiting the benefits extracted from the edge. This paper presents Arboreal, a novel distributed data management system for cloud and edge infrastructures that enables stateful edge applications to be deployed with full (read and write) local access to application data, overcoming the limitations of existing...

[77] A Performance Study of Epoch-based Commit Protocols in Distributed OLTP Databases
Jack Waudby, ..., and J. Webber. 2022 41st International Symposium on Reliable Distributed Systems (SRDS), 2022. 2 citations.
0% Topic Match

Abstract: Distributed OLTP systems execute the high-overhead, two-phase commit (2PC) protocol at the end of every distributed transaction. Epoch-based commit proposes that 2PC be executed only once for all transactions processed within a time interval called an epoch. Increasing epoch duration allows more transactions to be processed before the common 2PC. It thus reduces 2PC overhead per transaction, increases throughput but also increases average transaction latency. Therefore, required is the ability to choose the right epoch size that offers the desired trade-off between throughput and latency. To this end, we develop two analytical models to estimate throughput and average latency in...

[78] Keep CALM and CRDT On
Shadaj Laddad, ..., and Joseph M Hellerstein. Proc. VLDB Endow., 2022. 9 citations.
0% Topic Match

Abstract: Despite decades of research and practical experience, developers have few tools for programming reliable distributed applications without resorting to expensive coordination techniques. Conflict-free replicated datatypes (CRDTs) are a promising line of work that enable coordination-free replication and offer certain eventual consistency guarantees in a relatively simple object-oriented API. Yet CRDT guarantees extend only to data updates; observations of CRDT state are unconstrained and unsafe. We propose an agenda that embraces the simplicity of CRDTs, but provides richer, more uniform guarantees. We extend CRDTs with a query model that reasons about which queries are safe without coordination by applying monotonicity results...

[79] Detock: High Performance Multi-region Transactions at Scale
Cuong D. T. Nguyen, ..., and D. Abadi. Proceedings of the ACM on Management of Data, 2023. 16 citations.
0% Topic Match

Abstract: Many globally distributed data stores need to replicate data across large geographic distances. Since synchronously replicating data across such distances is slow, those systems with high consistency requirements often geo-partition data and direct all linearizable requests to the primary region of the accessed data. This significantly improves performance for workloads where most transactions access data close to where they originate from. However, supporting serializable multi-geo-partition transactions is a challenge, and they often degrade the performance of the whole system. This becomes even more challenging when they conflict with single-partition requests, where optimistic protocols lead to high numbers of aborts, and...

[80] Differentiated Consistency for Worldwide Gossips
David C Frey, ..., and François Taïani. IEEE Transactions on Parallel and Distributed Systems, 2023. 2 citations.
0% Topic Match

Abstract: Eventual consistency is a consistency model that favors liveness over safety. It is often used in large-scale distributed systems where models ensuring a stronger safety incur performance that are too low to be deemed practical. Eventual consistency tends to be uniformly applied within a system, but we argue a demand exists for differentiated eventual consistency, e.g. in blockchain systems. We propose update-query consistency with primaries and secondaries (UPS) to address this demand. UPS is a novel consistency mechanism that works in pair with our novel two-phase epidemic broadcast protocol gossip primary-secondary (GPS) to offer differentiated eventual consistency and delivery speed....

[81] Alone together: compositional reasoning and inference for weak isolation
Gowtham Kaki, ..., and S. Jagannathan. Proceedings of the ACM on Programming Languages, 2017. 26 citations.
0% Topic Match

Abstract: Serializability is a well-understood correctness criterion that simplifies reasoning about the behavior of concurrent transactions by ensuring they are isolated from each other while they execute. However, enforcing serializable isolation comes at a steep cost in performance because it necessarily restricts opportunities to exploit concurrency even when such opportunities would not violate application-specific invariants. As a result, database systems in practice support, and often encourage, developers to implement transactions using weaker alternatives. These alternatives break the strong isolation guarantees offered by serializable transactions to permit greater concurrency. Unfortunately, the semantics of weak isolation is poorly understood, and usually explained only...

[82] I Can't Believe It's Not Causal! Scalable Causal Consistency with No Slowdown Cascades
Syed Akbar Mehdi, ..., and Wyatt Lloyd. Unknown journal, 2017. 91 citations.
0% Topic Match

No summary or abstract available

[83] This Paper Is Included in the Proceedings of the 12th Usenix Symposium on Operating Systems Design and Implementation (osdi '16). the Snow Theorem and Latency-optimal Read-only Transactions the Snow Theorem and Latency-optimal Read-only Transactions
Haonan Lu, ..., and Wyatt Lloyd. Journal Not Provided, Unknown year. 66 citations.
0% Topic Match

No summary or abstract available

[84] Carousel: Low-Latency Transaction Processing for Globally-Distributed Data
Xinan Yan, ..., and Tim Brecht. Proceedings of the 2018 International Conference on Management of Data, 2018. 46 citations.
0% Topic Match

No summary or abstract available

[85] Fine-grained consistency for geo-replicated systems
Cheng Li, ..., and R. Rodrigues. Unknown journal, 2018. 20 citations.
0% Topic Match

No summary or abstract available

[86] ViSearch: Weak Consistency Measurement for Replicated Data Types
Lintian Shi, ..., and Xiaoxing Ma. ArXiv, 2022. 1 citations.
0% Topic Match

Abstract: Large-scale replicated data type stores often resort to eventual consistency to guarantee low latency and high availability. It is widely accepted that programming over eventually consistent data stores is challenging, since arbitrary divergence among replicas is allowed. Moreover, pragmatic protocols actually achieve consistency guarantees stronger than eventual consistency, which can be and need to be utilized to facilitate the reasoning of and programming over replicated data types. Toward the challenges above, we propose the ViSearch framework for precise measurement of eventual consistency semantics. ViSearch employs the visibility-arbitration specification methodology in concurrent programming, which extends the linearizability-based specification methodology with a...

[87] IsoDiff
Yifan Gan, ..., and Yang Wang. Proceedings of the VLDB Endowment, 2020. 19 citations.
0% Topic Match

Abstract: Weak isolation levels, such as Read Committed and Snapshot Isolation, are widely used by databases for their higher concurrency, but may introduce subtle correctness errors in applications that only experts can identify. This paper proposes IsoDiff, a tool to help a developer debug the anomalies caused by weak isolation for an application. To address the challenge that the number of anomalies can be non-polynomial with respect to the number of types of transactions, IsoDiff finds a representative subset of anomalies involving different transactions, operations, and problematic patterns. To reduce false positives, IsoDiff proposes two novel methods (correlation detection and timing...

[88] Hamsaz: replication coordination analysis and synthesis
Farzin Houshmand and M. Lesani. Proceedings of the ACM on Programming Languages, 2019. 50 citations.
0% Topic Match

Abstract: Distributed system replication is widely used as a means of fault-tolerance and scalability. However, it provides a spectrum of consistency choices that impose a dilemma for clients between correctness, responsiveness and availability. Given a sequential object and its integrity properties, we automatically synthesize a replicated object that guarantees state integrity and convergence and avoids unnecessary coordination. Our approach is based on a novel sufficient condition for integrity and convergence called well-coordination that requires certain orders between conflicting and dependent operations. We statically analyze the given sequential object to decide its conflicting and dependent methods and use this information to avoid...

[89] Read atomic transactions with prevention of lost updates: ROLA and its formal analysis
Si Liu, ..., and J. Meseguer. Formal Aspects of Computing, 2019. 13 citations.
0% Topic Match

No summary or abstract available

[90] Strong and Efficient Consistency with Consistency-aware Durability
Aishwarya Ganesan, ..., and Remzi H. Arpaci-Dusseau. ACM Transactions on Storage (TOS), 2021. 18 citations.
0% Topic Match

Abstract: We introduce consistency-aware durability or Cad, a new approach to durability in distributed storage that enables strong consistency while delivering high performance. We demonstrate the efficacy of this approach by designing cross-client monotonic reads, a novel and strong consistency property that provides monotonic reads across failures and sessions in leader-based systems; such a property can be particularly beneficial in geo-distributed and edge-computing scenarios. We build Orca, a modified version of ZooKeeper that implements Cad and cross-client monotonic reads. We experimentally show that Orca provides strong consistency while closely matching the performance of weakly consistent ZooKeeper. Compared to strongly consistent ZooKeeper,...

[91] Building Consistent Transactions with Inconsistent Replication
Irene Zhang, ..., and Dan R. K. Ports. ACM Transactions on Computer Systems (TOCS), 2018. 22 citations.
0% Topic Match

Abstract: Application programmers increasingly prefer distributed storage systems with strong consistency and distributed transactions (e.g., Google's Spanner) for their strong guarantees and ease of use. Unfortunately, existing transactional storage systems are expensive to use—in part, because they require costly

replication protocols, like Paxos, for fault tolerance. In this article, we present a new approach that makes transactional storage systems more affordable: We eliminate consistency from the replication protocol, while still providing distributed transactions with strong consistency to applications. We present the Transactional Application Protocol for Inconsistent Replication (TAPIR), the first transaction protocol to use a novel replication protocol, called inconsistent replication,...

[92] G-DUR: a middleware for assembling, analyzing, and improving transactional protocols
Masoud Saeida Ardekani, ..., and M. Shapiro. Unknown journal, 2014. 15 citations.
0% Topic Match

No summary or abstract available

[93] Mutating etcd Towards Edge Suitability
Andrew Jeffery, ..., and Richard Mortier. ArXiv, 2023. 1 citations.
0% Topic Match

Abstract: In the edge environment servers are no longer being co-located away from clients, instead they are being co-located with clients away from other servers, focusing on reliable and performant operation. Orchestration platforms, such as Kubernetes, are a key system being transitioned to the edge but they remain unsuited to the environment, stemming primarily from their critical key-value stores. In this work we derive requirements from the edge environment showing that, fundamentally, the design of distributed key-value datastores, such as etcd, is unsuited to meet them. Using these requirements, we explore the design space for distributed key-value datastores and implement two...

[94] Transaction processing in consistency-aware user's applications deployed on NoSQL databases
M. T. González-Aparicio, ..., and Ruben Casado. Human-centric Computing and Information Sciences, 2017. 12 citations.
0% Topic Match

Abstract: NoSQL databases are capable of storing and processing big data which is characterized by various properties such as volume, variety and velocity. Such databases are used in a variety of user applications that need large volume of data which is highly available and efficiently accessible. But they do not enforce or require strong data consistency nor do they support transactions. This paper investigates into the transaction processing in consistency-aware applications hosted on MongoDB and Riak which are two representatives of Document and Key-Value NoSQL databases, respectively. It develops new transaction schemes in order to provide NoSQL databases with transactional facilities...

[95] Quantitative Analysis of Consistency in NoSQL Key-Value Stores
Si Liu, ..., and J. Meseguer. Leibniz Trans. Embed. Syst., 2015. 31 citations.
0% Topic Match

Abstract: The promise of high scalability and availability has prompted many companies to replace traditional relational database management systems RDBMS with NoSQL key-value stores. This comes at the cost of relaxed consistency guarantees: key-value stores only guarantee eventual consistency in principle. In practice, however, many key-value stores seem to offer stronger consistency. Quantifying how well consistency properties are met is a non-trivial problem. We address this problem by formally modeling key-value stores as probabilistic systems and quantitatively analyzing their consistency properties by statistical model checking. We present for the first time a formal probabilistic model of Apache Cassandra, a popular NoSQL...

[96] Efficiently Supporting Multi-Level Serializability in Decentralized Database Systems
Zhanhao Zhao, ..., and Xiaoyong Du. IEEE Transactions on Knowledge and Data Engineering, 2023. 4 citations.
0% Topic Match

Abstract: In decentralized database systems, it is reported that serializability could still produce unexpected transaction orderings, leading to the stale read anomaly. To eliminate this anomaly, strict serializability imposes an additional ordering constraint, called the real-time order, which is required to be preserved among serializable transactions. Yet, preserving the real-time order in strict serializability often causes the performance to drop significantly. Because a weaker data consistency often yields better performance, in this paper, we model serializability from different consistency perspectives to properly leverage the performance and consistency. To do this, we first define a group of orderings, based on which we...

[97] SLOG: Serializable, Low-latency, Geo-replicated Transactions
Kun Ren, ..., and Daniel J. Abadi. Proc. VLDB Endow., 2019. 70 citations.
0% Topic Match

Abstract: For decades, applications deployed on a world-wide scale have been forced to give up at least one of (1) strict serializability (2) low latency writes (3) high transactional throughput. In this paper we discuss SLOG: a system that avoids this tradeoff for workloads which contain physical region locality in data access. SLOG achieves high-throughput, strictly serializable ACID transactions at geo-replicated distance and scale for all transactions submitted across the world, all the while achieving low latency for transactions that initiate from a location close to the home region for data they access. Experiments find that SLOG can reduce latency by...

[98] Consistency without borders
P. Alvaro, ..., and Joseph M Hellerstein. Proceedings of the 4th annual Symposium on Cloud Computing, 2013. 37 citations.
0% Topic Match

No summary or abstract available

[99] MINOS: Distributed Consistency and Persistency Protocol Implementation & Offloading to SmartNICs
Antonis Psistakis, ..., and Josep Torrellas. 2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2024. 1 citations.
0% Topic Match

Abstract: To enable high-performance, programmable, and resilient distributed systems, Distributed Data Persistency (DDP) models provide specific data consistency and persistency guarantees. Since these models target leaderless systems (i.e., systems where any node can initiate requests), they deliver high performance and are scalable. However, they are also more complex. In this paper, we develop detailed distributed algorithms for DDP models. They support Linearizable consistency with five different types of persistency. We call these algorithms MINOSBaseline (MINOS-B) and evaluate them on a 5-node distributed machine. Additionally, to improve performance, we also redesign the algorithms to offload them to a new SmartNIC architecture. The...

[100] Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data
Carlos Baquero and M. Serafini. Proceedings of the 7th Workshop on Principles and Practice of Consistency for Distributed Data, 2015. 0 citations.
0% Topic Match

Abstract: Consistency is one of the fundamental issues of distributed computing. There are many competing consistency models, with subtly different power in principle. In practice, the well-known Consistency-Availability-Partition Tolerance trade-off translates to difficult choices between fault tolerance, performance, and programmability. The issues and trade-offs are particularly vexing at scale, with a large number of processes or a large shared database, and in the presence of high latency and failure-prone networks. It is clear that there is no one universally best solution. Possible approaches cover the whole spectrum between strong and eventual consistency. Strong consistency (total ordering via, for example, linearizability or...

[101] Design and Validation of Cloud Storage Systems Using Rewriting Logic
P. C. Ölveczky. 2019 21st International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2019. 0 citations.
0% Topic Match

Abstract: Most large modern-day applications (Facebook, Gmail, eBay, etc.) rely on widely distributed and replicated storage of data for scalability, availability, and disaster tolerance. Since maintaining high degrees of data consistency requires costly communication across distant sites, applications over such distributed and partially replicated data are complex artifacts that must carefully balance the required degrees of consistency and performance. In this paper I summarize work at the University of Illinois Assured Cloud Computing center on using rewriting logic and its associated Maude tool environment to formally model and analyze both the correctness and the performance of state-of-the-art distributed transaction systems designs,...

[102] Scalable and Robust Snapshot Isolation for High-Performance Storage Engines
Adnan Alhomssi and Viktor Leis. Proc. VLDB Endow., 2023. 9 citations.
0% Topic Match

Abstract: MVCC-based snapshot isolation promises that read queries can proceed without interfering with concurrent writes. However, as we show experimentally, in existing implementations a single long-running query can easily cause transactional throughput to collapse. Moreover, existing out-of-memory commit protocols fail to meet the scalability needs of modern multi-core systems. In this paper, we present three complementary techniques for robust and scalable snapshot isolation in out-of-memory systems. First, we propose a commit protocol that minimizes cross-thread communication for better scalability, avoids touching the write set on commit, and enables efficient fine-granular garbage collection. Second, we introduce the Graveyard Index, an auxiliary data...

[103] Formal Modeling and Analysis of Cassandra in Maude
Si Liu, ..., and J. Meseguer. Unknown journal, 2014. 28 citations.
0% Topic Match
No summary or abstract available

[104] Achieving Probabilistic Atomicity With Well-Bounded Staleness and Low Read Latency in Distributed Datastores
Lingzhi Ouyang, ..., and Jian Lu. IEEE Transactions on Parallel and Distributed Systems, 2021. 3 citations.
0% Topic Match
Abstract: Although it has been commercially successful to deploy weakly consistent but highly-responsive distributed datastores, the tension between developing complex applications and obtaining only weak consistency guarantees becomes more and more severe. The almost strong consistency tradeoff aims at achieving both strong consistency and low latency in the common case. In distributed storage systems, we investigate the generic notion of almost strong consistency in terms of designing fast read algorithms while guaranteeing Probabilistic Atomicity with well-Bounded staleness (PAB). This problem has been explored in the case where only one client can write the data. However, the more general case where multiple...

[105] Causal memory: definitions, implementation, and programming
M. Ahamad, ..., and P. Hutto. Distributed Computing, 1995. 453 citations.
0% Topic Match
No summary or abstract available

[106] A Survey on NoSQL Stores
A. Davoudian, ..., and Mengchi Liu. ACM Computing Surveys (CSUR), 2018. 210 citations.
0% Topic Match
No summary or abstract available

[107] Consistency models in distributed systems: A survey on definitions, disciplines, challenges and applications
Hesam Nejati Sharif Aldin, ..., and Mostafa Razavi Ghods. ArXiv, 2019. 8 citations.
0% Topic Match
Abstract: The replication mechanism resolves some challenges with big data such as data durability, data access, and fault tolerance. Yet, replication itself gives birth to another challenge known as the consistency in distributed systems. Scalability and availability are the challenging criteria on which the replication is based upon in distributed systems which themselves require the consistency. Consistency in distributed computing systems has been employed in three different applicable fields, such as system architecture, distributed database, and distributed systems. Consistency models based on their applicability could be sorted from strong to weak. Our goal is to propose a novel viewpoint to different...

[108] Consistency-based service level agreements for cloud storage
D. Terry, ..., and Hussam Abu-Libdeh. Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, 2013. 271 citations.
0% Topic Match
No summary or abstract available

[109] Reconciling Earlier Snapshot Time With Local Cache for Optimal Performance Under Transactional Causal Consistency
Tieqiang Mo, ..., and Shan Duan. IEEE Transactions on Services Computing, 2021. 0 citations.
0% Topic Match
Abstract: Currently, Transactional Causal Consistency (TCC) has succeeded in removing some anomalies discovered in eventual consistency, diminishing the synchronization overhead consumed by strong consistency, and advocating interactive read-write transactions. However, the popular design in maximizing snapshot time for data freshness weighs against read-write interaction especially under read heavy overloads. Moreover, lower end-user latency and higher throughput in reading becomes paramount desiderata in deploying web services such as social networks among many data-centers. To this purpose, this article proposes earlier snapshot time and client-side local data-center cache for read-heavy transaction to boost lower response time and higher throughput. By the former, our...

[110] GOC-Ledger: State-based Conflict-Free Replicated Ledger from Grow-Only Counters
Erick Lavoie. ArXiv, 2023. 3 citations.
0% Topic Match
Abstract: Conventional blockchains use consensus algorithms that totally order updates across all accounts, which is stronger than necessary to implement a replicated ledger. This makes updates slower and more expensive than necessary. More recent consensus-free replicated ledgers forego consensus algorithms, with significant increase in performance and decrease in infrastructure costs. However, current designs are based around reliable broadcast of update operations to all replicas which require reliable message delivery and reasoning over operation histories to establish convergence and safety. In this paper, we present a replicated ledger as a state-based conflict-free replicated data type (CRDT) based on grow-only counters. This design...

[111] Byzantine-tolerant causal broadcast
Alex Auvolat, ..., and François Taïani. Theor. Comput. Sci., 2021. 26 citations.
0% Topic Match
No summary or abstract available

[112] Update Serializability in Locking
R. Hansdah and L. Patnaik. Unknown journal, 1986. 28 citations.
0% Topic Match
No summary or abstract available

[113] Hermes: A Fast, Fault-Tolerant and Linearizable Replication Protocol
Antonios Katsarakis, ..., and V. Nagarajan. Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems, 2020. 69 citations.
0% Topic Match
Abstract: Today's datacenter applications are underpinned by datastores that are responsible for providing availability, consistency, and performance. For high availability in the presence of failures, these datastores replicate data across several nodes. This is accomplished with the help of a reliable replication protocol that is responsible for maintaining the replicas strongly-consistent even when faults occur. Strong consistency is preferred to weaker consistency models that cannot guarantee an intuitive behavior for the clients. Furthermore, to accommodate high demand at real-time latencies, datastores must deliver high throughput and low latency. This work introduces Hermes, a broadcast-based reliable replication protocol for in-memory datastores that...

[114] SSS: Scalable Key-Value Store with External Consistent and Abort-free Read-only Transactions
Masoomeh Javidi Kishi, ..., and R. Palmieri. 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 2019. 9 citations.
0% Topic Match

Abstract: We present SSS, a scalable transactional key-value store deploying a novel distributed concurrency control that provides external consistency for all transactions, never aborts read-only transactions due to concurrency, all without specialized hardware. SSS ensures the above properties without any centralized source of synchronization. SSS's concurrency control uses a combination of vector clocks and a new technique, called snapshot-queuing, to establish a single serialization order where transactions are guaranteed to read from the latest non-concurrent transaction externally visible to clients. We compare SSS against high performance key-value stores, Walter, ROCOCO, and a two-phase commit baseline. SSS outperforms 2PC-baseline by as much...

[115] Implementing linearizability at large scale and low latency
Collin Lee, ..., and J. Ousterhout. Proceedings of the 25th Symposium on Operating Systems Principles, 2015. 75 citations.
0% Topic Match

No summary or abstract available

[116] Squirrel: an extensible distributed key-value store
Kevin De Porre and E. G. Boix. Proceedings of the 4th ACM SIGPLAN International Workshop on Meta-Programming Techniques and Reflection, 2019. 2 citations.
0% Topic Match

Abstract: Distributed key-value (KV) stores are a rising alternative to traditional relational databases since they provide a flexible yet simple data model. Recent KV stores use eventual consistency to ensure fast reads and writes as well as high availability. Support for eventual consistency is however still very limited as typically only a handful of replicated data types are provided. Moreover, modern applications maintain various types of data, some of which require strong consistency whereas other require high availability. Implementing such applications remains cumbersome due to the lack of support for data consistency in today's KV stores. In this paper we propose...

[117] Fast in-memory transaction processing using RDMA and HTM
Xingda Wei, ..., and Haibo Chen. Proceedings of the 25th Symposium on Operating Systems Principles, 2015. 329 citations.
0% Topic Match

No summary or abstract available

[118] Optimistic Execution in Key-Value Store
Duong N. Nguyen, ..., and M. Demirbas. ArXiv, 2018. 2 citations.
0% Topic Match

Abstract: Limitations of CAP theorem imply that if availability is desired in the presence of network partitions, one must sacrifice sequential consistency, a consistency model that is more natural for system design. We focus on the problem of what a designer should do if she has an algorithm that works correctly with sequential consistency but is faced with an underlying key-value store that provides a weaker (e.g., eventual or causal) consistency. We propose a detect-rollback based approach: The designer identifies a correctness predicate, say P , and continue to run the protocol, as our system monitors P . If P is...

[119] Omega: A Secure Event Ordering Service for the Edge
Cláudio Correia, ..., and Luís Rodrigues. IEEE Transactions on Dependable and Secure Computing, 2022. 1 citations.
0% Topic Match

No summary or abstract available

[120] Quaestor: Query Web Caching for Database-as-a-Service Providers
Felix Gessert, ..., and N. Ritter. Proc. VLDB Endow., 2017. 29 citations.
0% Topic Match

No summary or abstract available

[121] The Logical Timestamp Skew Anomaly in Event-Replicated Transaction Schedulers
Emil Koutanov. IEEE Access, 2021. 1 citations.
0% Topic Match

Abstract: To sidestep reasoning about the complex effects of concurrent execution, many system designers have conveniently embraced strict serializability on the strength of its claims, support from commercial and open-source database communities and ubiquitous levels of industry adoption. Crucially, distributed components are built on this model; multiple schedulers are composed in an event-driven architecture to form larger, ostensibly correct systems. This paper examines the oft-misconstrued position of strict serializability as a composable correctness criterion in the design of such systems. An anomaly is presented wherein a strict serializable scheduler in one system produces a history that cannot be serially applied to...

[122] Consistency in Partitioned Networks
Susan B. Davidson, ..., and Dale Skeen. ACM Comput. Surv., 1985. 580 citations.
0% Topic Match

No summary or abstract available

[123] Spanner: Google's globally-distributed database
Brian F. Cooper. Unknown journal, 2013. 503 citations.
0% Topic Match

No summary or abstract available

[124] Delta-State-Based Synchronization of CRDTs in Opportunistic Networks
F. Guidec, ..., and Camille Noûs. 2021 IEEE 46th Conference on Local Computer Networks (LCN), 2021. 1 citations.
0% Topic Match

Abstract: Conflict-Free Replicated Data Types (CRDTs) are distributed data types that support optimistic replication: replicas can be updated locally, and updates propagate asynchronously among replicas, so consistency is eventually obtained. This ability to tolerate asynchronous communication makes them ideal candidates to serve as software building blocks in opportunistic networks (OppNets), that is, mobile networks in which the dissemination of information can only depend on unpredicted transient radio contacts between pairs of nodes. In this paper we investigate the problem of implementing CRDTs in an Opp-Net, and we propose a delta-state-based algorithm to solve this problem. Experimental results confirm that this algorithm...

[125] Technical Report: Optimistic Execution in Key-Value Store
Duong N. Nguyen, ..., and M. Demirbas. ArXiv, 2018. 3 citations.
0% Topic Match

Abstract: Limitations of the CAP theorem imply that if availability is desired in the presence of network partitions, one must sacrifice sequential consistency, a consistency model that is more natural for system design. We focus on the problem of what a designer should do if he/she has an algorithm that works correctly with sequential consistency but is faced with an underlying key-value store that provides a weaker (e.g., eventual or causal) consistency. We propose a detect-rollback based approach: The designer identifies a correctness predicate, say P, and continues to run the protocol, as our system monitors P. If P is violated...

[126] Stateful Serverless Computing with Crucial
Daniel Barcelona-Pons, ..., and P. López. ACM Transactions on Software Engineering and Methodology (TOSEM), 2022. 45 citations.
0% Topic Match

Abstract: Serverless computing greatly simplifies the use of cloud resources. In particular, Function-as-a-Service (FaaS) platforms enable programmers to develop applications as individual functions that can run and scale independently. Unfortunately, applications that require fine-grained support for mutable state and synchronization, such as machine learning (ML) and scientific computing, are notoriously hard to build with this new paradigm. In this work, we aim at bridging this

gap. We present Crucial, a system to program highly-parallel stateful serverless applications. Crucial retains the simplicity of serverless computing. It is built upon the key insight that FaaS resembles to concurrent programming at the scale of...

[127] Building reliable distributed systems at a worldwide scale demands trade-offs between consistency and availability.
Werner Vogels. Journal Not Provided, Unknown year. 19 citations.
0% Topic Match
No summary or abstract available

[128] Contract-based return-value commutativity: safely exploiting contract-based commutativity for faster serializable transactions
T. Soethout, ..., and J. Vinju. Proceedings of the 11th ACM SIGPLAN International Workshop on Programming Based on Actors, Agents, and Decentralized Control, 2021. 0 citations.
0% Topic Match
Abstract: A key challenge of designing distributed software systems is maintaining data consistency. We can define data consistency and data isolation guarantees --e.g. serializability-- in terms of schedules of atomic reads and writes, but this excludes schedules that would be semantically consistent. Others use manually provided information on "non-conflicting operations" to define guarantees that work for more applications allowing more parallel schedules. To be safe, an engineer might avoid marking operations as non-conflicting, with detrimental effects to efficiency. To be fast, they might mark more non-conflicting operations than is strictly safe. Our goal is to help engineers by automatically deriving commutative...

[129] TAO: Facebook's Distributed Data Store for the Social Graph
N. Bronson, ..., and Venkateshwaran Venkataramani. Unknown journal, 2013. 550 citations.
0% Topic Match
No summary or abstract available

[130] Model checking driven explorative testing of CRDT designs and implementations
Yuqi Zhang, ..., and Xiaoxing Ma. Journal of Software: Evolution and Process, 2022. 8 citations.
0% Topic Match
Abstract: Internet scale distributed systems often replicate data at multiple geographic locations to provide low latency and high availability, despite node and network failures. According to the CAP theorem, low latency and high availability can only be achieved at the cost of accepting weak consistency. The conflict free replicated data type (CRDT) is a framework that provides a principled approach to maintaining eventual consistency among data replicas. CRDTs have been notoriously difficult to design and implement correctly. Subtle deep bugs lie in the complex and tedious handling of all possible cases of conflicting data updates. We argue that the CRDT design should be...

[131] Optimal Reads-From Consistency Checking for C11-Style Memory Models
P. Abdulla, ..., and Hunkar Can Tuncc. Proceedings of the ACM on Programming Languages, 2023. 11 citations.
0% Topic Match
Abstract: Over the years, several memory models have been proposed to capture the subtle concurrency semantics of C/C++. One of the most fundamental problems associated with a memory model M is consistency checking: given an execution X, is X consistent with M? This problem lies at the heart of numerous applications, including specification testing and litmus tests, stateless model checking, and dynamic analyses. As such, it has been explored extensively and its complexity is well-understood for traditional models like SC and TSO. However, less is known for the numerous model variants of C/C++, for which the problem becomes challenging due to...

[132] Sequential consistency versus linearizability
H. Attiya and J. Welch. ACM Transactions on Computer Systems (TOCS), 1994. 284 citations.
0% Topic Match
No summary or abstract available

[133] Linearizability of Persistent Memory Objects Under a Full-System-Crash Failure Model
Joseph Izraelevitz, ..., and M. Scott. Unknown journal, 2016. 183 citations.
0% Topic Match
No summary or abstract available

[134] Robustness against release/acquire semantics
O. Lahav and R. Margalit. Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, 2019. 31 citations.
0% Topic Match
Abstract: We present an algorithm for automatically checking robustness of concurrent programs against C/C++11 release/acquire semantics, namely verifying that all program behaviors under release/acquire are allowed by sequential consistency. Our approach reduces robustness verification to a reachability problem under (instrumented) sequential consistency. We have implemented our algorithm in a prototype tool called Rocker and applied it to several challenging concurrent algorithms. To the best of our knowledge, this is the first precise method for verifying robustness against a high-level programming language weak memory semantics.

[135] Semantically Correct and Intent Defined Commutativity in Distributed Systems
Courtney Robinson. 2020 19th International Symposium on Parallel and Distributed Computing (ISPDC), 2020. 1 citations.
0% Topic Match
Abstract: This paper reviews the problem of correctness in the context of scaleable distributed systems. It takes the concept of a strongly eventually consistent (SEC) data structure based on the work done on conflict free replicated data types and examines the theoretical basis of a practical application to generalised database management systems.It reasons that the condition required for a database management system to implement a monotonic semilattice and remain generic is not possible. It further proposes a framework which extends the SEC condition to capture enough information to make such a system practical to implement. Finally, it evaluates such an implementation...

[136] Efficient Renaming in Sequence CRDTs
Matthieu Nicolas, ..., and Olivier Perrin. IEEE Transactions on Parallel and Distributed Systems, 2020. 2 citations.
0% Topic Match
Abstract: To achieve high availability, large-scale distributed systems have to replicate data and to minimise coordination between nodes. For these purposes, literature and industry increasingly adopt Conflict-free Replicated Data Types (CRDTs) to design such systems. CRDTs are new specifications of existing data types, e.g. Set or Sequence. While CRDTs have the same behaviour as previous specifications in sequential executions, they actually shine in distributed settings as they natively support concurrent updates.To this end, CRDTs embed in their specification conflict resolution mechanisms. These mechanisms usually rely on identifiers attached to elements of the data structure to resolve conflicts in a deterministic and...

[137] This Paper Is Included in the Proceedings of the 11th Usenix Symposium on Networked Systems Design and Implementation (nsdi '14). Farm: Fast Remote Memory Farm: Fast Remote Memory
A. Dragojevic, ..., and Aleksandar Dragojevi´c. Journal Not Provided, Unknown year. 871 citations.
0% Topic Match
No summary or abstract available

[138] Loom: A Closed-Box Disaggregated Database System
Fábio Coelho, ..., and Rui Oliveira. Proceedings of the 12th Latin-American Symposium on Dependable and Secure Computing, 2023. 0 citations.
0% Topic Match
Abstract: Cloud native database systems provide highly available and scalable services as part of cloud platforms by transparently replicating and partitioning data across automatically managed resources. Some systems, such as Google Spanner, are designed and implemented from scratch. Others, such as Amazon Aurora, derive from traditional database systems for better compatibility but disaggregate storage to cloud services. Unfortunately, because they follow an open-box approach

and fork the original code base, they are difficult to implement and maintain. We address this problem with Loom, a replicated and partitioned database system built on top of PostgreSQL that delegates durable storage to a distributed...

[139] A Surfing Concurrence Transaction Model for Key-Value NoSQL Databases
Changqing Li and Jianhua Gu. Journal of Software Engineering and Applications, 2018. 1 citations.
0% Topic Match

Abstract: As more and more application systems related to big data were developed, NoSQL (Not Only SQL) database systems are becoming more and more popular. In order to add transaction features for some NoSQL database systems, many scholars have tried different techniques. Unfortunately, there is a lack of research on Redis's transaction in the existing literatures. This paper proposes a transaction model for key-value NoSQL databases including Redis to make possible allowing users to access data in the ACID (Atomicity, Consistency, Isolation and Durability) way, and this model is vividly called the surfing concurrence transaction model. The architecture, important features and...

[140] Exploring the design space of highly-available distributed transactions
Alejandro Z. Tomsic. Unknown journal, 2018. 1 citations.
0% Topic Match

No summary or abstract available

[141] Seeing is Believing: A Client-Centric Specification of Database Isolation
Natacha Crooks, ..., and Allen Clement. Proceedings of the ACM Symposium on Principles of Distributed Computing, 2017. 60 citations.
Not measured Topic Match

No summary or abstract available

[142] Generalized isolation level definitions
A. Adya, ..., and P. O'Neil. Proceedings of 16th International Conference on Data Engineering (Cat. No.00CB37073), 2000. 230 citations.
Not measured Topic Match

No summary or abstract available

[143] Algebraic Laws for Weak Consistency
A. Cerone, ..., and Hongseok Yang. Unknown journal, 2017. 32 citations.
Not measured Topic Match

No summary or abstract available

[144] Eventually Consistent Transactions
S. Burckhardt, ..., and Shmuel Sagiv. Unknown journal, 2012. 94 citations.
Not measured Topic Match

No summary or abstract available

[145] From session causality to causal consistency
J. BrzeziDski, ..., and D. Wawrzyniak12th Euromicro Conference on Parallel, Distributed and Network-Based Processing, 2004. Proceedings., 2004. 58 citations.
Not measured Topic Match

No summary or abstract available

[146] Session guarantees for weakly consistent replicated data
D. Terry, ..., and B. Welch. Proceedings of 3rd International Conference on Parallel and Distributed Information Systems, 1994. 483 citations.
Not measured Topic Match

No summary or abstract available

[147] A promising semantics for relaxed-memory concurrency
Jeehoon Kang, ..., and Derek Dreyer. Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, 2017. 194 citations.
Not measured Topic Match

No summary or abstract available

[148] Principles of Eventual Consistency
S. Burckhardt. Found. Trends Program. Lang., 2014. 140 citations.
Not measured Topic Match

No summary or abstract available

[149] Putting consistency back into eventual consistency
Valter Balegas, ..., and M. Shapiro. Proceedings of the Tenth European Conference on Computer Systems, 2015. 143 citations.
Not measured Topic Match

No summary or abstract available

[150] The serializability of concurrent database updates
C. Papadimitriou. J. ACM, 1979. 1047 citations.
Not measured Topic Match

No summary or abstract available

[151] Limitations of Highly-Available Eventually-Consistent Data Stores
H. Attiya, ..., and Adam Morrison. IEEE Transactions on Parallel and Distributed Systems, 2015. 74 citations.
Not measured Topic Match

No summary or abstract available

[152] TiQuE: Improving the Transactional Performance of Analytical Systems for True Hybrid Workloads
Nuno Faria, ..., and N. Nes. Proc. VLDB Endow., 2023. 4 citations.
Not measured Topic Match

Abstract: Transactions have been a key issue in database management for a long time and there are a plethora of architectures and algorithms to support and implement them. The current state-of-the-art is focused on storage management and is tightly coupled with its design, leading, for instance, to the need for completely new engines to support new features such as Hybrid Transactional Analytical Processing (HTAP). We address this challenge with a proposal to implement transactional logic in a query language such as SQL. This means that our approach can be layered on existing analytical systems but that the retrieval of a transactional...

[153] Viper: A Fast Snapshot Isolation Checker
Jian Zhang, ..., and Cheng Tan. Proceedings of the Eighteenth European Conference on Computer Systems, 2023. 13 citations.
Not measured Topic Match

Abstract: Snapshot isolation (SI) is supported by most commercial databases and is widely used by applications. However, checking SI today---given a set of transactions, checking if they obey SI---is either slow or gives up soundness. We present viper, an SI checker that is sound, complete, and fast. Viper checks black-box databases and hence is transparent to both users and databases. To be fast, viper introduces BC-polygraphs, a new representation of transaction dependencies.

A BC-polygraph is acyclic iff transactions are SI, a theorem that we prove. Viper also introduces heuristic pruning, an optimization to accelerate checking SI by leveraging common knowledge of...

[154] Decidable verification under a causally consistent shared memory
O. Lahav and Udi Boker. Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation, 2020. 21 citations.
Not measured Topic Match

Abstract: Causal consistency is one of the most fundamental and widely used consistency models weaker than sequential consistency. In this paper, we study the verification of safety properties for finite-state concurrent programs running under a causally consistent shared memory model. We establish the decidability of this problem for a standard model of causal consistency (called also "Causal Convergence" and "Strong-Release-Acquire"). Our proof proceeds by developing an alternative operational semantics, based on the notion of a thread potential, that is equivalent to the existing declarative semantics and constitutes a well-structured transition system. In particular, our result allows for the verification of a...

[155] MRVs: Enforcing Numeric Invariants in Parallel Updates to Hotspots with Randomized Splitting
Nuno Faria and José Pereira. Proceedings of the ACM on Management of Data, 2023. 2 citations.
Not measured Topic Match

No summary or abstract available

[156] NOC-NOC: Towards Performance-optimal Distributed Transactions
Si Liu, ..., and David A. Basin. Proceedings of the ACM on Management of Data, 2024. 7 citations.
Not measured Topic Match

Abstract: Substantial research efforts have been devoted to studying the performance optimality problem for distributed database transactions. However, they focus just on optimizing transactional reads, and thus overlook crucial factors, such as the efficiency of writes, which also impact the overall system performance. Motivated by a recent study on Twitter's workloads showing the prominence of write-heavy workloads in practice, we make a substantial step towards performance-optimal distributed transactions by also aiming to optimize writes, a fundamentally new dimension to this problem. We propose a new design objective and establish impossibility results with respect to the achievable isolation levels. Guided by these...

[157] Database Consistency Models
M. Shapiro and P. Sutra. ArXiv, 2018. 11 citations.
Not measured Topic Match

Abstract: A data store allows application processes to put and get data from a shared memory. In general, a data store cannot be modelled as a strictly sequential process. Applications observe non-sequential behaviours, called anomalies. The set of pos- sible behaviours, and conversely of possible anomalies, constitutes the consistency model of the data store.

[158] PaRiS: Causally Consistent Transactions with Non-blocking Reads and Partial Replication
Kristina Spirovska, ..., and W. Zwaenepoel. 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 2019. 22 citations.
Not measured Topic Match

Abstract: Geo-replicated data platforms are the backbone of several large-scale online services. Transactional Causal Consistency (TCC) is an attractive consistency level for building such platforms. TCC avoids many anomalies of eventual consistency, eschews the synchronization costs of strong consistency, and supports interactive read-write transactions. Partial replication is another attractive design choice for building geo-replicated platforms, as it reduces storage requirements and update propagation costs. This paper presents PaRiS, the first TCC system that supports partial replication and implements non-blocking parallel read operations. The latter reduce read latency which is of paramount importance for the performance of read-intensive applications. PaRiS relies on...

[159] Bonspiel: Low Tail Latency Transactions in Geo-Distributed Databases
Fan Cui, ..., and Ziliang Lai. Proc. VLDB Endow., 2025. 0 citations.
Not measured Topic Match

Abstract: Tail latency is crucial as it impacts user satisfaction and service-level objectives (SLOs). However, geo-distributed databases have long struggled with this issue due to wide-area network access, resulting in tail latencies of several or even exceeding ten seconds. In this paper, we highlight that further optimizing atomic commit protocols does not help but hit a tail latency wall. Instead, making concurrency control and access method selection geo-aware can mitigate this issue. To this end, we present Bonspiel, a new geo-distributed database equipped with geo-aware concurrency control and access method selection. In our experiments, Bonspiel successfully caps the tail latency of...

[160] Occam's Razor for Distributed Protocols
Ziliang Lai, ..., and Feifei Li. Proceedings of the 2024 ACM Symposium on Cloud Computing, 2024. 1 citations.
Not measured Topic Match

Abstract: Optimizing distributed protocols has traditionally been a real pain, requiring experts to figure out where improvements can be made, along with rigorous correctness proofs and meticulous implementation. This paper presents a theory to systematize this process. The proposed theory can optimize any existing distributed protocols while preserving all their original quality attributes (e.g., generality, correctness). Crucially, applying the optimizations derived from this theory does not necessitate touching the original implementation --- all you need is just to bolt on a few new message adapters. Case studies demonstrate the effectiveness of this approach. For instance, applying the theory to optimize Spanner's...

[161] Peepco: Batch-Based Consistency Optimization
I. Kuraj, ..., and Armando Solar-Lezama. Proceedings of the ACM on Programming Languages, 2025. 0 citations.
Not measured Topic Match

Abstract: We present batch-based consistency, a new approach for consistency optimization that allows programmers to specialize consistency with application-level integrity properties. We implement the approach with a two-step process: we statically infer optimal consistency requirements for executions of bounded sets of operations, and then, use the inferred requirements to parameterize a new distributed protocol to relax operation reordering at run time when it is safe to do so. Our approach supports standard notions of consistency. We implement batch-based consistency in Peepco, demonstrate its expressiveness for partial data replication, and examine Peepco’s run-time performance impact in different settings.

[162] Practical client-side replication
A. Linde, ..., and Nuno M. Preguiça. Proceedings of the VLDB Endowment, 2020. 12 citations.
Not measured Topic Match

Abstract: Client-side replication and direct client-to-client synchronization can be used to create highly available, low-latency interactive applications. Causal consistency, the strongest available consistency model under network partitions, is an attractive consistency model for these applications. This paper focuses on how client misbehaviour impacts causal consistency. We analyze the possible attacks to causal consistency and derive secure consistency models that preclude different types of misbehaviour. We propose a set of techniques for implementing such secure consistency models, which exhibit different trade-offs between the application guarantees, and the latency and communication overhead. Our evaluation shows that secure consistency models impose low overhead when...

[163] FW-KV: improving read guarantees in PSI
Masoomeh Javidi Kishi and R. Palmieri. Proceedings of the 22nd International Middleware Conference, 2021. 0 citations.
Not measured Topic Match

Abstract: We present FW-KV, a novel distributed transactional in-memory key-value store that guarantees the Parallel Snapshot Isolation (PSI) correctness level. FW-KV's primary goal is to allow its read-only transactions to access more up-to-date (fresher) versions of objects than Walter, the state-of-the-art implementation of PSI. FW-KV achieves that without assuming synchrony or a synchronized clock service. The improved level of freshness comes at no significant performance degradation, especially in low contention workloads, as assessed by our evaluation study including two standard OLTP benchmarks, YCSB and TPC-C. The performance gap between FW-KV and Walter is less than 5% in low contention scenarios, and...

[164] On Reading Fresher Snapshots in Parallel Snapshot Isolation
Masoomeh Javidi Kishi and R. Palmieri. 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), 2020. 1 citations.

Not measured Topic Match

Abstract: In this paper we briefly present FPSI, a distributed transactional in-memory key-value store whose primary goal is to enable transactions to read more up-to-date (fresher) versions of shared objects than existing implementations of the well-known Parallel Snapshot Isolation (PSI) correctness level, in the absence of a synchronized clock service among nodes. FPSI builds upon Walter, an implementation of PSI well suited for social applications. The novel concurrency control at the core of FPSI allows its abort-free read-only transactions to access the latest version of objects upon their first contact to a node.