DOI: 10.15514/ISPRAS-2026-38(5)-3

An Epistemic Approach to Conflict-Free Replicated Data Types

G.V. Semenov, ORCID: 0000-0003-4725-7666 <georgii.v.semenov@gmail.com>
ITMO University,
49, bldg. A, Kronverksky pr., Saint Petersburg, 197101, Russia.

Abstract. Optimistic replication enables highly available geo-distributed systems by allowing replicas to diverge temporarily and reconcile asynchronously. While celebrated conflict-free replicated data types (CRDTs) guarantee strong eventual convergence, their built-in conflict resolution mechanisms are often implicit and inflexible, limiting their applicability in scenarios requiring semantic transparency and user involvement. In this paper, we investigate the epistemic conflict model, a framework that enables explicit, flexible, and coordination-free conflict resolution while preserving convergence guarantees. Firstly, we show that state-based, operation-based and delta-CRDTs are naturally expressed in the epistemic model and provide a framework for their conflict-free representation. Secondly, we consider sets as a fundamental replicated data type that empowers more advanced replicated objects such as lists, graphs and maps, and define the respective conflict resolution procedures in the epistemic model, which are known as add-wins and remove-wins semantics in CRDTs. Thus, we demonstrate that the epistemic conflict model is sufficiently expressive to capture semantics of existing CRDT designs, while preserving interpretability.

Keywords: conflict-free replicated data types; semantic conflicts; epistemic conflict model; optimistic replication.

For citation: Semenov G.V. An Epistemic Approach to Conflict-Free Replicated Data Types. Trudy ISP RAN/Proc. ISP RAS, 2026, vol. 38, issue 5, pp. 37-50. DOI: 10.15514/ISPRAS-2026-38(5)-3.

Эпистемический подход к бесконфликтно реплицируемым типам

Г.В. Семенов, ORCID: 0000-0003-4725-7666 <georgii.v.semenov@gmail.com>
Университет ИТМО,
Россия, 197101, г. Санкт-Петербург, Кронверкский пр., д. 49, к. А.

Аннотация. Оптимистическая репликация является одним из ключевых подходов к обеспечению высокой доступности географически распределённых систем, позволяя репликам временно расходиться в состоянии с последующим асинхронным согласованием. Несмотря на то, что широко распространённые бесконфликтные реплицируемые типы данных (Conflict-free Replicated Data Types, CRDT) гарантируют сильную сходимость, встроенные в них механизмы разрешения конфликтов, как правило, являются неявными и недостаточно гибкими, что ограничивает их применение в сценариях, требующих семантической прозрачности и участия пользователя в процессе разрешения конфликтов. В работе исследуется эпистемическая модель конфликтов – формальная модель, обеспечивающая явное, гибкое и не требующее координации разрешение конфликтов при сохранении гарантий сходимости. Во-первых, показано, что CRDT, основанные на состоянии, на операциях и на дельта-состояниях (delta-CRDT), естественным образом формализуются в рамках эпистемической модели, которая предоставляет единый аппарат для их бесконфликтного представления. Во-вторых, множества рассматриваются как фундаментальный реплицируемый тип данных, лежащий в основе более сложных реплицируемых структур, таких как списки, графы и отображения. Для них в рамках эпистемической модели определяются соответствующие процедуры разрешения конфликтов, реализующие семантики, известные в теории CRDT. Тем самым демонстрируется, что эпистемическая модель конфликтов обладает достаточной выразительной мощностью для формализации семантики существующих реализаций CRDT, одновременно сохраняя их интерпретируемость и семантическую прозрачность.

Ключевые слова: бесконфликтно реплицируемые типы данных; семантические конфликты; эпистемическая модель конфликта; оптимистическая репликация.

Для цитирования: Семенов Г.В. Эпистемический подход к бесконфликтно реплицируемым типам. Труды ИСП РАН, 2026, том 38, вып. 5, стр. 37–50 (на английском языке). DOI: 10.15514/ISPRAS-2026-38(5)-3.

1. Introduction

Optimistic replication [1] is widely employed in asynchronous geo-distributed systems to improve availability at the expense of strict or sequential consistency. In such systems, multiple versions of the same data may coexist and, after concurrent modifications, are synchronized a posteriori. Consequently, local replicas are allowed to remain temporarily inconsistent or globally outdated. Because updates are performed without coordination, concurrent operations may conflict, giving rise to ambiguity in how changes should be merged into a shared history. To resolve such ambiguities, a *conflict model* is required to guide the synchronization process, ensuring eventual consistency while preserving data integrity. A conflict model can be defined by specifying which concurrent modifications are *identified* as conflicting and how such conflicts are eventually *resolved*.

The strictest interpretation of conflicting operations is captured by the *concurrent* conflict model, based on the happens-before relation ($\prec$) [2]. Under this model, two operations are considered conflicting if they are causally unrelated; consequently, an explicit ordering must be established between them. Classical mechanisms for identifying such conflicts rely on vector timestamps, including vector clocks [3-4], plausible clocks [5], and prime clocks [6]. However, conflict resolution under this model is inherently "pessimistic", as it requires nodes to agree on a global ordering of concurrent operations during the *scheduling* phase [1]. In asynchronous environments subject to failures and unbounded network delays, reaching consensus is fundamentally impossible in the general case, as established by the FLP theorem [7].

To address the challenge of conflict-free synchronization, the notion of logical monotonicity has emerged as a foundational principle. Informally, monotonic programs admit consistent,

coordination-free distributed implementations; this insight is formalized by the CALM theorem [8-9]. Within this framework, semilattices, which are monotonic algebraic structures equipped with a least upper bound (LUB), form the basis of *state-based* conflict-free replicated data types (convergent CRDTs, or CvRDTs) [10]. In such systems, replicas periodically exchange and merge their states using an idempotent, commutative, and associative binary operator, ensuring convergence without coordination due to the totality of the LUB binary operator.

Complementarily, the concept of operational commutativity [11] underpins *operation-based* CRDTs [10]. In this setting, conflicts are restricted to concurrent operations that do not commute, defining a *non-commutative* conflict model. It is well established that state-based and operation-based CRDTs are expressively equivalent, in the sense that each model can simulate the other [10]. Moreover, δ-CRDTs [12] further optimize state-based CRDTs by transmitting only the incremental changes (deltas) instead of the entire state, bridging the gap between state-based and operation-based approaches.

CRDTs have thus become a standard abstraction for coordination-free distributed systems, as they guarantee strong eventual consistency (SEC). Formally, an object is strongly eventually consistent if (i) it is eventually consistent, and (ii) any two correct replicas that have delivered the same set of updates reach equivalent states [10]. However, despite these strong convergence guarantees, the conflict resolution mechanisms embedded in CRDTs are typically implicit and opaque to users. They are difficult to supervise or adapt, and often misaligned with application-specific semantics, which limits their suitability for human-centric tasks such as collaborative decision-making, reasoning, auditing, and explainability. In other words, the ideas of monotonicity and commutativity are too restrictive for representation of arbitrary replicated objects.

To address the challenges above, the *epistemic* conflict model [13] aims to provide a semantic framework for representation of replicated objects with support for local-first [14], user-driven conflict resolution. The model uses the ideas of monotonicity and strong eventual convergence, while allowing conflicts to arise and be explicitly resolved. To identify conflicts, operations are assigned an immutable set of causal premises, and two operations are in conflict if a common premise *entails* ($\vdash$) both of them, but there is an operation which *discards* ($\ll$) the visibility of the premise. To resolve conflicts, operations are allowed to be concurrently *rebased* ($\hat{\vdash}$), which allows coordination-free conflict resolution.

In this paper, we aim to show that conflict-free replicated data types can be modelled using the epistemic conflict model. Firstly, we show that state-based, operation-based and delta-CRDTs can be trivially expressed in the epistemic model, where no conflict identification is needed and hence there is no need for a resolution procedure. Secondly, we consider sets as a family of fundamental replicated data types that empowers more advanced objects such as lists, graphs and maps, and provide its definitions in the epistemic model, considering the expressibility of grow-only, add-wins and remove-wins semantics.

The rest of the paper is organized as follows. In Section 2, we recall the system model and the requirements for objects to be represented in the epistemic conflict model. In Section 3, we show how various kinds of CRDTs can be modelled in the epistemic framework. In Section 4, we provide definitions in the epistemic model for the Grow-Only Set (G-Set), add-only Observed-Remove Set (OR-Set), and build an original construction for remove-wins set. In Section 5, we discuss related work and conclude in Section 6.

2. Epistemic conflict model

The epistemic conflict model [13] is a semantic framework for representation of replicated objects with support for local-first [14] conflict resolution. That is, conflicts are allowed to arise and be explicitly resolved without a central coordinator, while preserving convergence guarantees. In this section, we outline the system model and recall the requirements for objects to be represented in the epistemic conflict model.

2.1 System model

Let us consider a system of multiple nodes, which are the *replicas* $\mathcal{R} = \{r_1, r_2, \cdots, r_n\}$ of a shared namespace M. A *namespace* M is modelled as a set of pre-initialized replicated objects $\{M_1, M_2, \cdots, M_n\}$. Each object M_i is associated with its basis of *actions* A_i, which is formally a set of all allowed atomic operations on the object.

Definition 1 (Operation). An *operation* o is a contiguous sequence of action instances $\{a_1, a_2, \cdots, a_k\}$ with transactional semantics, i.e., executing an operation o is equivalent to a sequential uninterruptible execution of all its actions $a_1 \circ a_2 \circ \cdots \circ a_k, a_i \in \cup^n A_i$.

Definition 2 (History). *Local history* $\mathcal{H}_r$ is a sequence of operations, currently applied by replica r.

Definition 3 (Projection). *Local history projection* $\mathcal{H}_r(x)$ is a subsequence of actions, corresponding to object $x = M_i$, in the local history $\mathcal{H}_r$ of replica r.

Operations issued on the replicas are *applied* locally and eventually *synchronized* between nodes, which are connected with an unreliable network. Initially, each object $x = M_i$ is initialized with a *constructor* operation c_x. When an operation is *applied* by a replica, it is just appended to the local history $\mathcal{H}_r$. Eventually, the current replica history $\mathcal{H}_r$ is *published*, so that it is available for pull on the other replicas. When ready to reconcile with foreign changes, replica r *synchronizes-with* another replica r' by integrating operations from another replica's published history $\mathcal{H}_{r'}$ into the local history, i.e., $\mathcal{H}_r \leftarrow \mathcal{H}_r \sqcup \mathcal{H}_{r'}$. Complete synchronization is achieved through integrating the histories of all replicas in an arbitrary order, until they converge to the same *state* once all the conflicts are resolved.

2.2 Conflict definition

In the epistemic conflict model, conflicts are defined in terms of two relations over operations: *entails* ($\vdash$) and *is discarded by* ($\ll$).

Entailment ($\vdash$) captures *epistemic preconditions* [15] of an operation: issuing a new operation relies on the presence of effects of its premises, under the assumption that their effects remain visible.

Definition 4 (Visibility). An action instance a corresponding to object x is said to be *visible* in the history $\mathcal{H}_r(x)$, if a monotonic non-increasing predicate associated with the action holds, i.e., $vis(a, \mathcal{H}_r(x)) = \top$.

To capture the dependency on visibility of other operations, an operation o is assigned an immutable set of *premises* $\Gamma = \{o_1, o_2, \cdots, o_n\}$ that entail o, which is denoted as $\Gamma \vdash o$.

Definition 5 (Entailment). An operation o_1 is said to *entail* another operation o_2, denoted as $o_1 \vdash o_2$, if the effect of o_1 is a necessary precondition for the effect of o_2, to be visible in the history, i.e.:

1. $o_1 \in \Gamma(o_2)$, where $\Gamma(o_2)$ is the set of premises of o_2;
2. $o_1 \prec o_2$, i.e., o_1 happens-before o_2, in any history $\mathcal{H}_r$;
3. $vis(o_1, \mathcal{H}_r \setminus \{o_2\}) = \top$ for any history $\mathcal{H}_r$.

The relationship of discarding ($\ll$) captures *epistemic revision* [16]: an operation may invalidate the effect of another operation, thereby rendering it no longer a valid premise for subsequent operations.

Definition 6 (Discarding). An operation o_1 is said to *be discarded by* another operation o_2, denoted as $o_1 \ll o_2$, if:

1. $o_1 \vdash o_2$;
2. $vis(o_1, \mathcal{H}_r) = \bot$, where $o_2 \in \mathcal{H}_r$.

Finally, the conflict identification procedure is defined as follows: two operations are in conflict if they have a common premise that is discarded by one of them.

Definition 7 (Conflict Identification). Operations o_1 and o_2 are said to be in *conflict* if there exists an operation o' such that $o' \vdash o_1$ and $o' \ll o_2$.

2.3 Object definition

An essential criterion for convergence lies in the notion of discard-completeness, which ensures that all operations that may discard the visibility of a common premise are connected through the entailment relation.

Definition 8 (Discard-completeness). An object x with an action basis A is *discard-complete* if for every pair of its action instances a_1, a_2 in any history $\mathcal{H}_r(x)$ of x the following holds:

$$vis(a_1, \mathcal{H}_r(x) \setminus \{a_2\}) \wedge \neg vis\big(a_1, \mathcal{H}_r(x)\big) \Rightarrow a_1 \ll a_2.$$

Then, to represent a replicated object $x = M_i$ in the epistemic conflict model, the state of the object must be reducible from the local history projection $\mathcal{H}_r(x)$, the following must hold:

1. *Entailment* relation $\vdash$ is defined to link an action $a \in A_i$ of an operation o to its premises $\Gamma(o)$ upon creation;
2. *Visibility* predicate $vis\big(a, \mathcal{H}_r(x)\big)$ is defined for $a \in A_i$;
3. *Discard-completeness* is satisfied.

2.4 Conflict resolution

Conflict resolution in the epistemic model is based on the idea of *rebasing* operations to a merge operation that reconciles them.

Definition 9 (Rebase). An operation o is well-defined *rebased* to $\hat{o}$, denoted as $\hat{o} \;\hat{\vdash}\; o$, when

1. o is interpreted as an empty sequence of actions;
2. $\Gamma(o) \subseteq \Gamma(\hat{o})$;
3. $\forall o' \colon (o' \vdash \hat{o}) \wedge (\hat{o} \;\hat{\vdash}\; o) \Rightarrow (o' \vdash o)$.

Thus, conflict resolution is achieved through identifying an operation component containing mutually conflicting operations and rebasing them to a merge operation $\hat{o}$ that reconciles them, as shown in Fig. 1 and 2. The contents of $\hat{o}$ are defined by an external $reconcile$ procedure, which generally may be interactive and involve user participation. In this paper, we consider data types with a deterministic resolution procedure, which is a function of the conflicting operations and their premises.

Fig. 1. Conflict resolution of o_1 and o_2 with one conflicting premise o'.

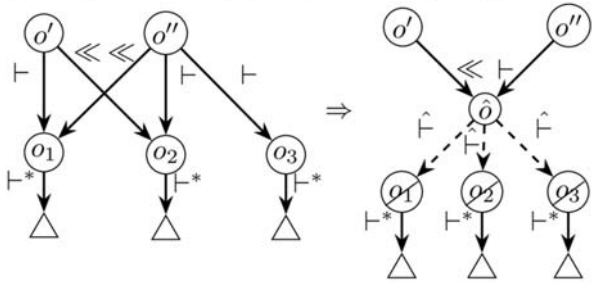

Fig. 2. Conflict resolution of o_2 against o_1 and o_3 with two conflicting premises o' and o'', and the merge operation $\hat{o}$ discarding premise o'.

3. CRDTs in the Epistemic model

Conflict-free replicated data types (CRDTs) [10] are a class of data structures that guarantee strong eventual consistency (SEC) by their definition, i.e., any two correct replicas that have delivered the same set of updates reach equivalent states. Despite their name, CRDTs do not necessarily avoid conflicts; rather, they are designed to ensure that conflicts are resolved in a deterministic and coordination-free manner.

However, the conflict resolution policies embedded in CRDTs are restricted by the data type definition and may appear to be not a full-fledged conflict resolution, but a "hack" to provide strong convergence at the cost of versatility. For example, in the last-writer-wins (LWW) register, the conflict resolution policy is based on a total ordering of operations, provided by the wall-clock timestamps, which may not align with the intuitive semantics of a register, as older updates are silently discarded.

In this section, we recall the definitions of state-based and operation-based CRDTs and provide a generic framework to model them in the epistemic conflict model, extending it for the case of δ-CRDTs.

3.1 State-based CRDTs

Definition 10 (State-based CRDT). A state-based CRDT (convergent CRDT, CvRDT) is a state-transfer replicated data type empowered by the merge operation $\sqcup$ (or the least-upper bound over the semi-lattice) defined on the state domain of the object S, such that $\forall s_1, s_2, s_3 \in S$:

1. $s_1 \sqcup s_2 = s_2 \sqcup s_1$ (commutativity);
2. $s_1 \sqcup (s_2 \sqcup s_3) = (s_1 \sqcup s_2) \sqcup s_3$ (associativity);
3. $s_1 \sqcup s_1 = s_1$ (idempotency);
4. $s_1 \sqcup s_2 \in S$ (totality, or conflict-free criterion).

Let us consider a state-based CRDT x with a state domain S and a merge operation $\sqcup$. Then, we can model x in the epistemic conflict model as an object with the following operations:

- constructor operation c_x that initializes the object to the least element of the semi-lattice, i.e., $s_{c_x} = \bot$;
- modification operation $m(s)$ that updates the replica state s' of the object to $s \in S$, such that $s' \sqcup s = s$;
- merge operation $\sqcup$ that takes two states $s_1, s_2 \in S$ and produces a new state $s_1 \sqcup s_2$.

Now, we can define the necessary model properties for representing the object x as follows:

1. **Entailment**:
   - constructor operation c_x has no premises, i.e., $\Gamma(c_x) = \emptyset$;
   - modification operation $m(s)$ is entailed by the operation o that produced the current state s', i.e., $\Gamma\big(m(s)\big) = \{o\}$.
   - merge operation $\sqcup$ is entailed by the previous operations o_1 and o_2 that produced the states s_1 and s_2 being merged, i.e., $\Gamma(\sqcup) = \{o_1, o_2\}$.
2. **Visibility**: due to the totality of the merge operation $\sqcup$, there are no conflicting operations in the history, and thus all operations are visible, i.e., $\forall o \in \mathcal{H}_r(x) \colon vis\big(o, \mathcal{H}_r(x)\big) = \top$.
3. **Discard-completeness**: due to the totality of the merge operation $\sqcup$, there are no conflicting operations in the history, and thus no operation can discard the visibility of another operation, i.e., $\neg\exists o_1, o_2 \in \mathcal{H}_r(x) \colon o_1 \ll o_2$.

3.2 Operation-based CRDTs

Definition 11 (Operation-based CRDT). An operation-based CRDT is an operation-transfer replicated data type empowered by the operation broadcasting, such that:

1. an operation is applied exactly-once (reliable delivery);
2. operations are applied in causal order with respect to the happens-before ($\prec$) relation (causal delivery);
3. $\neg(o_1 \prec o_2) \wedge \neg(o_2 \prec o_1) \Rightarrow o_1 \circ o_2 = o_2 \circ o_1$ (commutativity, or conflict-free criterion).

Similarly, let us consider an operation-based CRDT x with an action basis A within the epistemic model:

- constructor operation c_x that initializes the object with a sequence of actions $a_1, a_2, \dots, a_k$ that produce the initial state of the object;
- modification operation o that atomically transforms the object state with a sequence of actions $a_1, a_2, \dots, a_k$.

Now, we can define the necessary model properties for representing the object x as follows:

1. **Entailment**:
   - constructor operation c_x has no premises, i.e., $\Gamma(c_x) = \emptyset$;
   - modification operation o is transitively entailed by all the previous operations on the replica, i.e., $\forall o' \prec o: o' \vdash^* o$.
2. **Visibility**: without further assumptions, all operations may be treated as visible, due to their commutativity.
3. **Discard-completeness**: trivially satisfied, since all operations are visible.

3.3 Delta-CRDTs

δ-CRDTs [12] refine state-based CRDTs by replacing full-state transmission with transmission of *delta-states* (or *deltas*), i.e., fragments $d \in S$ such that $d \sqsubseteq s$ and $d \sqcup s = s'$, where s' is the updated state of the object. Formally, every update operation u on a δ-CRDT produces a delta-state $\delta_u \in S$ satisfying $s \sqcup \delta_u = u(s)$, and replicas exchange deltas instead of full states, reducing communication overhead while preserving the semilattice structure.

We model a δ-CRDT x in the epistemic conflict model as follows. The action basis is extended with a *delta* action $\delta(d)$ for each delta-state $d \in S$, which replaces the monolithic modification action $m(s)$ of the state-based construction. Constructor operation c_x initializes the object to the bottom element $\perp \in S$. A delta action $\delta(d)$ represents an incremental update, so that the replica state is updated as $s' \leftarrow s \sqcup d$ upon applying $\delta(d)$.

The epistemic model properties are then defined as follows:

1. **Entailment**:
   - constructor operation c_x has no premises, i.e., $\Gamma(c_x) = \emptyset$;
   - a delta action $\delta(d)$ is entailed by the operation o that produced the state s from which d was derived, i.e., $\Gamma\big(\delta(d)\big) = \{o\}$;
   - a merge of an accumulated delta $d = d_1 \sqcup d_2 \sqcup \cdots \sqcup d_k$ is entailed by all operations o_i that produced the constituent deltas d_i, i.e., $\Gamma\big(\delta(d)\big) = \{o_1, o_2, \dots, o_k\}$.
2. **Visibility**: as in the state-based case, the totality of $\sqcup$ ensures that any accumulated delta d is compatible with the current replica state s, so $\forall o \in \mathcal{H}_r(x)$: $vis\big(o, \mathcal{H}_r(x)\big) = \top$.

3. **Discard-completeness**: trivially satisfied, since no delta action can make another action invisible – the semilattice order guarantees $s \sqcup d \geq s$, and thus visibility is monotonically preserved.

4. Set semantics in the Epistemic model

The key characterization of removal-supported sets is that concurrent adds and removes on the same element are allowed to conflict; the conflict is resolved by an explicitly chosen *conflict policy*: either *add-wins* (concurrent adds take precedence, and the element remains in the set) or *remove-wins* (concurrent removes take precedence, and the element is removed).

In this section, we instantiate the epistemic framework for classical conflict-free type definitions: grow-only, add-wins, and remove-wins sets. This is of fundamental importance, as sets are the most basic data type, and their semantics are the basis for more complex data types, such as maps, graphs, and sequences. For example, Last-Writer-Wins Map (LWW-Map) is a map of keys to LWW-Registers, and it may be modeled as a set of key-value pairs with add-wins semantics, where the value is the timestamped register state.

For each type we provide its action basis A, the entailment premises $\Gamma(o)$ assigned at operation creation, and the visibility predicate $vis(a, \mathcal{H}_r)$, satisfying the discard-completeness property. Recall that discarding and conflict identification are derived from visibility alone (see Definition 6), hence we do not need to declare conflicts separately.

4.1 Grow-only semantics

A *grow-only set* (G-Set) is a CRDT that supports only insertion operations on elements from a universe $\mathcal{U}$. Elements can be added but never removed, and the merge operation is simply set union, which is commutative, associative, and idempotent over the semilattice of all finite subsets of a universe $\mathcal{U}$. Since no action ever makes another invisible, there are no discards and hence no conflicts. The G-Set is a degenerate, trivially convergent object in the epistemic model.

Let us model the G-Set in the epistemic conflict model as a delta-transfer object x, following the approach for delta-based CRDTs in Section 3.3:

- **Action basis.** $A = \{add(e) \mid e \in \mathcal{U}\}$.
- **Entailment.**
  - $\Gamma(c) = \emptyset$ for the constructor c.
  - $\Gamma\big(add(e)\big) =$ last in $\mathcal{H}_r(x)$.
- **Visibility.**
  - $vis\big(c, \mathcal{H}_r(x)\big) = \top$.
  - $vis\big(add(e), \mathcal{H}_r(x)\big) = \top$.
- **Discard-completeness.** Trivially satisfied.
- **State.** $S = \{e \mid add(e) \in \mathcal{H}_r(x)\}$.

An example for the definition above is shown in Fig. 3. The figure illustrates two concurrent insertions $add_1(e_1)$ and $add_2(e_2)$ issued on different replicas. In the diagram, $\sqcup_1$ and $\sqcup_2$ represent replica-local merges, while the later $add_3(e_2)$ shows that an additional insertion may be performed after the first merge, because the insertion δ-mutator is idempotent by definition.

4.2 Add-wins semantics

An *observed-remove set* (OR-Set) is a CRDT that supports both insertion and removal operations. OR-Set implements the *add-wins* policy, as remove works only with the elements which are visible at the time of the remove operation (Fig. 4).

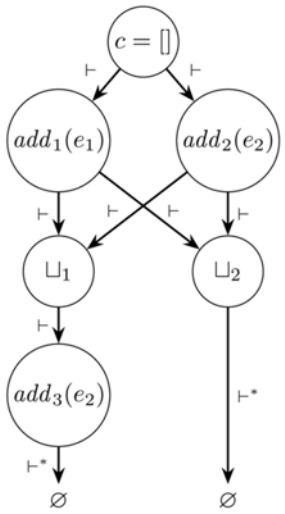

Fig. 3. G-Set: concurrent delta-mutators $\mathtt{add}_1(e_1)$ and $\mathtt{add}_2(e_2)$ are merged on the replicas on $\sqcup_1$ and $\sqcup_2$, and an additional insertion $\mathtt{add}_3(e_2)$ is performed after the first merge.

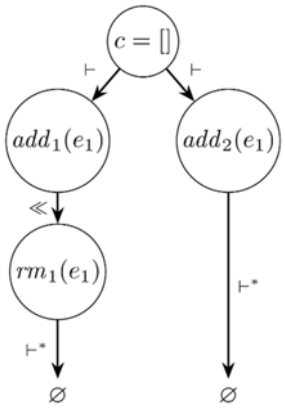

Fig. 4. OR-Set: there is no conflict between $\mathtt{rm}_1(e_1)$ and $\mathtt{add}_2(e_1)$, because the semantics of $\mathtt{rm}_1(e_1)$ is defined in terms of the visible adds of e_1 at the time of its creation.

In the epistemic model, this can be achieved by making the remove operation depend on the set of all currently visible adds of e.

- **Action basis.** $A = \{add(e),\ rmv(e) \mid e \in \mathcal{U}\}$.
- **Entailment.**
  - $\Gamma(c) = \emptyset$ for the constructor c.
  - $\Gamma\big(add(e)\big) = \{\, c \,\}$.
  - $\Gamma\big(rmv(e)\big) = \{\ add_i(e) \in \mathcal{H}_r(x) \mid vis\big(add_i(e), \mathcal{H}_r(x)\big)\ \}$.
- **Visibility.**

  - $vis\big(c, \mathcal{H}_r(x)\big) = \top$.
  - $vis\big(add(e), \mathcal{H}_r(x)\big) = \top \iff \nexists\, rmv_j(e) \in \mathcal{H}_r(x) \colon add(e) \in \Gamma\left(rmv_j(e)\right)$
  - $\mathrm{vis}\big(\mathrm{rmv(e)}, \mathcal{H}_r(\mathrm{x})\big) = \top$
- **Discard-completeness.** If $add_i(e)$ is invisible, then it is entailed by $rmv_j(e)$. Thus, the object is discard-complete.
- **State.** $S = \{e \mid \exists\, add_i(e) \in \mathcal{H}_r(x) \colon vis\big(add_i(e), \mathcal{H}_r(x)\big)\}$.

4.3 Remove-wins semantics

Remove-wins semantics inverts the priority of the add-wins policy: when an $add(e)$ and a $rmv(e)$ are issued concurrently, the remove takes precedence and the element is absent from the merged state. Although less widely known than OR-Set, a remove-wins observed-remove set has been noted in the delta-enabled CRDT literature [17].

We construct a remove-wins set in the epistemic model by maintaining two internal components: a positive set P and a tombstone set N. The positive set P follows the G-Set construction of Section 4, recording every insertion ever performed. The tombstone set N follows the add-wins OR-Set construction, recording every removal. An element e belongs to the logical state of the object if and only if it has been added to P and has not been removed into N. The action basis, entailment, and visibility predicate are defined as follows.

- **Action basis.** $A = \{\, add(e),\ rmv(e) \mid e \in \mathcal{U}\,\}$:
  - $add(e) = [add_P(e), rmv_N(e)]$;
  - $rmv(e) = [add_N(e)]$.
- **Entailment.**
  - $\Gamma(c) = \emptyset$ for the constructor c.
  - $\Gamma\big(add(e)\big) = \{c_P\} \cup \Gamma\big(rmv_N(e)\big)$.
  - $\Gamma\big(rmv(e)\big) = \{c_N\}$.
- **Visibility.**
  - $vis\big(c, \mathcal{H}_r(x)\big) = \top$.
  - $vis\big(add(e), \mathcal{H}_r(x)\big) = \top$.
  - $vis\big(rmv(e), \mathcal{H}_r(x)\big) = \nexists add_i(e) \in \mathcal{H}_r(x) \colon rmv(e) \vdash add_i(e)$.
- **Discard-completeness.** If $rmv(e)$ is invisible, then there is $add_i(e)$ such that $rmv(e) \vdash add_i(e)$, so the object is discard-complete.
- **State.** $S = S_P \setminus S_N$.

Let us consider the scenario in Fig. 5. Operation $rmv_1(e_1)$ has seen the existing element e_1, because $add_1(e_1)$ was already present in the local history at that moment. Any concurrent insertion will have no effect on the merged state, because $rmv_1(e_1)$ enforces the remove-wins policy. If $add_2(e_1)$ is issued after $rmv_1(e_1)$, it may restore e_1, so the element is visible in the set again.

4.4 Conflict definitions

Despite the conflict-free nature of the set constructions above, conflicts can still arise when concurrent operations are issued on the same element in terms of epistemic model. These conflicts are remove-remove conflicts in OR-Set, which occur when two concurrent remove operations target the same element, and add-add conflicts in remove-wins set, which occur when two concurrent add operations target the same element. In both cases, the conflict is resolved by a local rebase operation

$\hat{\digamma}$, which produces a merge operation that is visible on both replicas and converges to the same state without coordination, as shown in Fig. 6.

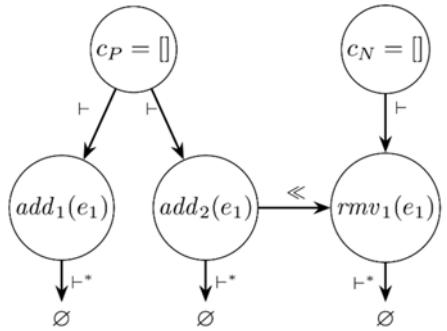

Fig. 5. Remove-wins set: an example entailment graph with $add_1(e_1)$ and $rmv_1(e_1)$ issued concurrently.

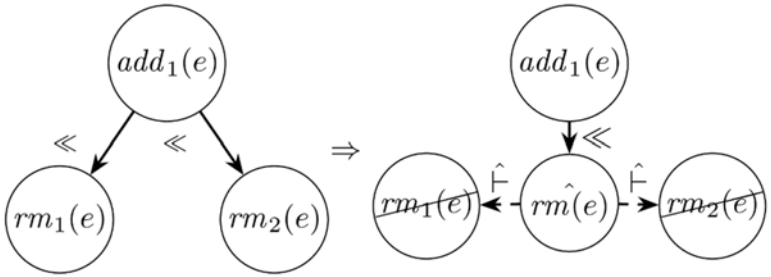

Fig. 6. Concurrent remove conflict resolution in epistemic OR-Set.

5. Discussion

Relation to semantic reconciliation frameworks. The epistemic conflict model shares its motivation with earlier work on semantics-based reconciliation of optimistically replicated data. IceCube [18] and the constraint-based formalism of Shapiro et al. [19] represent a shared object as a set of actions connected by semantic constraints such as conflict, dependence, and atomicity. Reconciliation is then cast as a scheduling problem: find an ordering of concurrent actions that satisfies all constraints and is as close as possible to each replica's local preference. The decentralized commitment protocol of Sutra et al. [20] extends this line of work by running an asynchronous election among candidate schedules without a central coordinator. The epistemic model differs from these approaches in two important ways. First, it does not require a global search over schedules; conflicts are identified and resolved purely through the entailment and visibility predicates that are attached to individual operations at creation time. Second, resolution is expressed as a local rebase operation $\hat{\digamma}$, which produces a merge operation convergently and without coordination, directly in the spirit of local-first computing [14]. This makes the epistemic model more suitable for settings with high latency or frequent disconnection, where schedule-based approaches incur unacceptable overhead.

Relation to classical set CRDTs. The set constructions presented in Section 4 are related to, but distinct from, the classical two-phase set (2P-Set) of Shapiro et al [10]. A 2P-Set also maintains a positive set P and a tombstone set N, with the rule that an element is present iff $e \in P \setminus N$. However, 2P-Set enforces a monotonicity invariant that prevents any element from being re-added once removed, and it resolves concurrent add/remove conflicts by simply treating all tombstoned

elements as permanently deleted – effectively a remove-wins policy without explicit conflict identification. In contrast, the epistemic model makes the conflict policy explicit and derives it from the visibility predicate: the add-wins OR-Set of Section 4 and the remove-wins construction of the same section are both instances of the same framework, distinguished solely by the definition of $\Gamma(rmv)$ and the visibility rule. This uniformity is precisely the expressiveness gain that the epistemic model provides over the implicit, per-type conflict resolution mechanisms of classical CRDTs.

6. Conclusion

In this paper, we have shown that conflict-free replicated data types fit naturally into the epistemic conflict model as a degenerate case in which the conflict identification procedure is vacuous: the totality of the semilattice merge (for state-based and delta-CRDTs) and the commutativity of concurrent operations (for operation-based CRDTs) ensure that visibility is always $\top$ and discard completeness is trivially satisfied. This observation establishes the epistemic model as a genuine generalization of the CRDT family rather than an orthogonal approach.

Beyond the CRDT embedding, we have defined three set semantics directly in the epistemic framework. The G-Set is a straightforward application of the delta-transfer construction. The add wins OR-Set is obtained by restricting remove operations to the set of currently visible insertions, so that concurrent adds are never invalidated. The remove-wins set is a novel epistemic construction in which the premise subsumption rule gives removes priority over concurrent adds whose causal context they cover. All three constructions are proved discard-complete, guaranteeing strong eventual convergence without coordination.

Because sets are the foundation for more complex replicated data types such as maps, graphs, and sequences, these definitions provide a reusable building block for richer objects. Future work includes formalizing the epistemic model for collaborative text editing and tree-structured documents, investigating the compositionality of epistemic object definitions, and exploring interactive resolution procedures in which the $reconcile$ function is supplied by the user rather than fixed at design time.

References

[1]. Saito Y., Shapiro M. Optimistic replication. ACM Comput. Surv, Mar. 2005, vol. 37, no. 1, pp. 42-81. DOI: 10.1145/1057977.1057980.

[2]. Lamport L. Time, clocks, and the ordering of events in a distributed system. Commun. ACM, Jul. 1978, vol. 21, no. 7, pp. 558-565. DOI: 10.1145/359545.359563.

[3]. Fidge C.J. Partial orders for parallel debugging. In Proceedings of the 1988 ACM SIGPLAN and SIGOPS Workshop on Parallel and Distributed Debugging, ser. PADD '88. New York, NY, USA: Association for Computing Machinery, 1988, pp. 183-194. DOI: 10.1145/68210.69233.

[4]. Mattern F. et al. Virtual time and global states of distributed systems. Univ., Department of Computer Science Saarbrücken, Germany, 1988.

[5]. Torres-Rojas F.J., Ahamad M. Plausible clocks: constant size logical clocks for distributed systems. Distributed Computing, 1999, vol. 12, no. 4, pp. 179-195. DOI: 10.1007/s004460050065.

[6]. Kshemkalyani A.D., Shen M., Voleti B. Prime clock: Encoded vector clock to characterize causality in distributed systems. Journal of Parallel and Distributed Computing, 2020, vol. 140, pp. 37-51. Available at: https://www.sciencedirect.com/science/article/pii/S0743731519304939, accessed 05.08.2026.

[7]. Fischer M.J., Lynch N.A., Paterson M.S. Impossibility of distributed consensus with one faulty process. J. ACM, Apr. 1985, vol. 32, no. 2, pp. 374-382. DOI: 10.1145/3149.214121.

[8]. Hellerstein J.M., Alvaro P. Keeping CALM: when distributed consistency is easy. CoRR, 2019. DOI: 10.48550/arXiv.1901.01930.

[9]. Hellerstein J.M., Alvaro P. Keeping calm: when distributed consistency is easy. Commun. ACM, Aug. 2020, vol. 63, no. 9, pp. 72-81. DOI: 10.1145/3369736.

[10]. Shapiro M., Preguiça N., Baquero C., Zawirski M. A comprehensive study of Convergent and Commutative Replicated Data Types. Inria – Centre Paris-Rocquencourt; INRIA, Research Report RR-7506, Jan. 2011. Available at: https://inria.hal.science/inria-00555588, accessed 05.08.2026.

[11]. Weihl W. Commutativity-based concurrency control for abstract data types. IEEE Transactions on Computers, 1988, vol. 37, no. 12, pp. 1488-1505.
[12]. Almeida P.S., Shoker A., Baquero C. Delta state replicated data types. Journal of Parallel and Distributed Computing, 2018, vol. 111, pp. 162-173. Available at: https://www.sciencedirect.com/science/article/pii/S0743731517302332, accessed 05.08.2026.
[13]. Semenov G., Aksenov V. Semantic conflict model for collaborative data structures. In Proceedings of the 13th International Workshop on Principles and Practice of Consistency for Distributed Data, ser. PaPoC '26. New York, NY, USA: Association for Computing Machinery, 2026, pp. 25-31. DOI: 10.1145/3806077.3806700.
[14]. Kleppmann M., Wiggins A., van Hardenberg P., McGranaghan M. Local-first software: you own your data, in spite of the cloud. In Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, ser. Onward! 2019. New York, NY, USA: Association for Computing Machinery, 2019, pp. 154-178. DOI: 10.1145/3359591.3359737.
[15]. Van Ditmarsch H., van Der Hoek W., Kooi B. Dynamic epistemic logic. Springer, 2008.
[16]. Baltag A., Smets S. A qualitative theory of dynamic interactive belief revision. Logic and the foundations of game and decision theory (LOFT 7), 2008, vol. 3, pp. 9-58.
[17]. Baquero C. delta-enabled-crdts, 2014. Available at: https://github.com/CBaquero/delta-enabled-crdts, accessed 14.04.2026.
[18]. Preguiça N., Shapiro M., Matheson C. Semantics-based reconciliation for collaborative and mobile environments. In On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE, Meersman R., Tari, Z., Schmidt D.C. Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 38-55.
[19]. Shapiro M., Bhargavan K., Krishna N. A constraint-based formalism for consistency in replicated systems. In Principles of Distributed Systems, Higashino T. Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 331-345.
[20]. Sutra P., Barreto J., Shapiro M. Decentralised commitment for optimistic semantic replication. In Proceedings of the 2007 OTM Confederated International Conference on On the Move to Meaningful Internet Systems: CoopIS, DOA, ODBASE, GADA, and IS – Volume Part I, ser. OTM'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 318-335. DOI: 10.1007/978-3-540-76848-7_21.

Information about authors

Георгий Витальевич Семенов – аспирант университета ИТМО. Его научные интересы включают распределенные системы, протоколы синхронизации и согласования, бесконфликтные реплицируемые типы данных, компьютерная поддержка совместной работы.

Georgii Vitalyevich SEMENOV – postgraduate student at ITMO University (Saint Petersburg, Russia). His research interests include distributed systems, synchronization and reconciliation protocols, conflict-free replicated data types (CRDT), computer-supported cooperative work (CSCW).