Non-uniform Replication

Gonçalo Cabrita and Nuno Preguiça
NOVA LINCS

OPODIS'17, Lisbon, Portugal,
18-20 December 2017

Context

- Increase in user activity has forced services to find new ways to scale
- Several services store their data in geo-replicated key-value stores
- These data stores sacrifice strong consistency for high availability

Problem

- Information stored in these data stores increases rapidly
- It is typically impossible to maintain all the data in all replicas
- Some systems adopt a partial replication model

Example


BACK
Leaderboard
Total Score
Position 1
Global
National
Local
Me and my Friends
1 dsf 17760680
Global Standings
1 dsf 17760680
2 uaefalcon 17746010
3 Vartap 17609190
4 jojomi2ndlife 17606310
5 birdybum 17389580
6 apptt 17183650
7 MadMoa 17148930
8 AngryMP 17147650
BIRDS
PLAY

Example: Top-1 (partial replication)

$$\{ \quad \} \; \{ \quad \} \; \{ \quad \}$$

Example: Top-1 (partial replication)

`ADD(Mary, 90)` @ 1

$\left\{ \text{Mary, 90} \right\} \quad \left\{ \quad \right\} \quad \left\{ \quad \right\}$

Example: Top-1 (partial replication)

`ADD(Mary, 90)` @ 1

{ Mary, 90 }   { }   { Mary, 90 }

Example: Top-1 (partial replication)

{ Mary, 90 } { } { Mary, 90 }

Example: Top-1 (partial replication)

`ADD(Amy, 80)` @ 2, `ADD(John, 85)` @ 3

$$\left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Amy, 80} \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Mary, 90} \\ \text{John, 85} \\ \\ \end{array} \right\}$$

Example: Top-1 (partial replication)

`ADD(Amy, 80)` @ 2, `ADD(John, 85)` @ 3

$\left\{ \begin{array}{c} \text{Mary, 90} \\ \text{Amy, 80} \\ \\ \end{array} \right\}$ $\left\{ \begin{array}{c} \text{Amy, 80} \\ \text{John, 85} \\ \\ \end{array} \right\}$ $\left\{ \begin{array}{c} \text{Mary, 90} \\ \text{John, 85} \\ \\ \end{array} \right\}$

Example: Top-1 (partial replication)

{ Mary, 90; Amy, 80 }   { Amy, 80; John, 85 }   { Mary, 90; John, 85 }

Example: Top-1 (partial replication)

$$\left\{ \begin{array}{c} \textbf{Mary, 90} \\ \text{Amy, 80} \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Amy, 80} \\ \text{John, 85} \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \textbf{Mary, 90} \\ \text{John, 85} \\ \\ \end{array} \right\}$$

Can we create a replication model where any single object replica can answer all read operations without storing all the data?

Example: Top-1 (non-uniform replication)

$$\left\{ \qquad\qquad \right\} \left\{ \qquad\qquad \right\}$$

Example: Top-1 (non-uniform replication)

`ADD(Mary, 90)` @ 1

{ Mary, 90 } { }

Example: Top-1 (non-uniform replication)

`ADD(Mary, 90)` @ 1

$\left\{ \text{Mary, 90} \right\}$ $\left\{ \text{Mary, 90} \right\}$

Example: Top-1 (non-uniform replication)

$$\left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \end{array} \right\}$$

Example: Top-1 (non-uniform replication)

`ADD(John, 80)` @ 1

{ Mary, 90
John, 80 }

{ Mary, 90 }

Example: Top-1 (non-uniform replication)

$$\left\{ \begin{array}{c} \text{Mary, 90} \\ \text{John, 80} \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \end{array} \right\}$$

Example: Top-1 (non-uniform replication)

`ADD(John, 85)` @ 1

$$\left\{ \begin{array}{c} \text{Mary, 90} \\ \text{John, 85} \\ \text{John, 80} \\ \\ \end{array} \right\} \qquad \left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \\ \end{array} \right\}$$

Example: Top-1 (non-uniform replication)

`ADD(John, 85)` @ 1

{ Mary, 90; John, 85; ~~John, 80~~ }

{ Mary, 90 }

Example: Top-1 (non-uniform replication)

$$\left\{ \begin{array}{c} \text{Mary, 90} \\ \text{John, 85} \\ \\ \end{array} \right\} \qquad \left\{ \begin{array}{c} \text{Mary, 90} \\ \\ \\ \end{array} \right\}$$

Example: Top-1 (non-uniform replication)

`RMV(Mary)` @ 1

{ ~~Mary, 90~~ John, 85 }	{ Mary, 90 }

Example: Top-1 (non-uniform replication)

`RMV(Mary)` @ 1

$\left\{ \begin{array}{c} \text{~~Mary, 90~~} \\ \text{John, 85} \\ \\ \end{array} \right\}$ $\left\{ \begin{array}{c} \text{~~Mary, 90~~} \\ \\ \\ \end{array} \right\}$

Example: Top-1 (non-uniform replication)

$$\left\{ \begin{array}{c} \text{John, 85} \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{John, 85} \\ \\ \\ \end{array} \right\}$$

Road Map

- **Non-uniform Replication**
- Non-uniform CRDTs
- Evaluation
- Conclusion and future work

Non-uniform Replication

- A replication model where all replicas can answer all supported queries, while maintaining only a subset of the data
- Replicas of the same object are not required to have **equivalent** states, instead they are required to have **observable equivalent** states
- For two states to be **observable equivalent** a read operation must return the same result for both states

Non-uniform Replication

{ Mary, 90
John, 85 }

{ Mary, 90 }

Non-uniform Replication

```
ADD(Amy, 100)
```

$$\left\{ \begin{array}{c} \textbf{Mary, 90} \\ \text{John, 85} \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \textbf{Amy, 100} \\ \text{Mary, 90} \\ \\ \end{array} \right\}$$

Eventual Consistency

A replicated system provides **eventual consistency** if in a quiescent state:

1. Each replica executed **all** operations
2. The state of any pair of replicas is **equivalent**

Non-uniform Eventual Consistency (NuEC)

A replicated system provides **non-uniform eventual consistency** if in a quiescent state:

1. Every replica executed a set of operations that impact the final **observable state**
2. The state of any pair of replicas is **observable equivalent**

Algorithm for providing NuEC (in an op-based CRDT model)

The goal is to divide operations, using only local information, into four groups:

1. Operations that are **core**
2. Operations that are **masked** but can become **core**
3. Operations that are **forever masked**
4. Operations that are **masked** but in the context of the entire system are considered **core**

$$\left\{ \begin{array}{c} \text{Paul, 80} \\ \\ \\ \\ \end{array} \right\}$$

Algorithm for providing NuEC (in an op-based CRDT model)

The goal is to divide operations, using only local information, into four groups:

1. Operations that are **core**
2. Operations that are **masked** but can become **core**
3. Operations that are **forever masked**
4. Operations that are **masked** but in the context of the entire system are considered **core**

`ADD(John, 85)`

{
John, 85
Paul, 80
}

Algorithm for providing NuEC (in an op-based CRDT model)

The goal is to divide operations, using only local information, into four groups:

1. Operations that are **core**
2. Operations that are **masked** but can become **core**
3. Operations that are **forever masked**
4. Operations that are **masked** but in the context of the entire system are considered **core**

`ADD(Amy, 50)`

{ **John, 85**
Paul, 80
Amy, 50 }

Algorithm for providing NuEC (in an op-based CRDT model)

The goal is to divide operations, using only local information, into four groups:

1. Operations that are **core**
2. Operations that are **masked** but can become **core**
3. Operations that are **forever masked**
4. Operations that are **masked** but in the context of the entire system are considered **core**

`ADD(Amy, 52)`

$$\left\{ \begin{array}{c} \text{John, 85} \\ \text{Paul, 80} \\ \text{Amy, 52} \\ \text{Amy, 50} \end{array} \right\}$$

Algorithm for providing NuEC (in an op-based CRDT model)

The goal is to divide operations, using only local information, into four groups:

1. Operations that are **core**
2. Operations that are **masked** but can become **core**
3. Operations that are **forever masked**
4. Operations that are **masked** but in the context of the entire system are considered **core**

{
John, 85
Paul, 80
Amy, 52
~~Amy, 50~~
}

Fault-tolerance

- Not propagating masked operations raises the issue of the durability of operations
- Possible solution:
  - Source replicas propagate masked operations to at least f other replicas
- Base algorithm would have to be updated to consider the case where the source replicas of a masked operation fail

Road Map

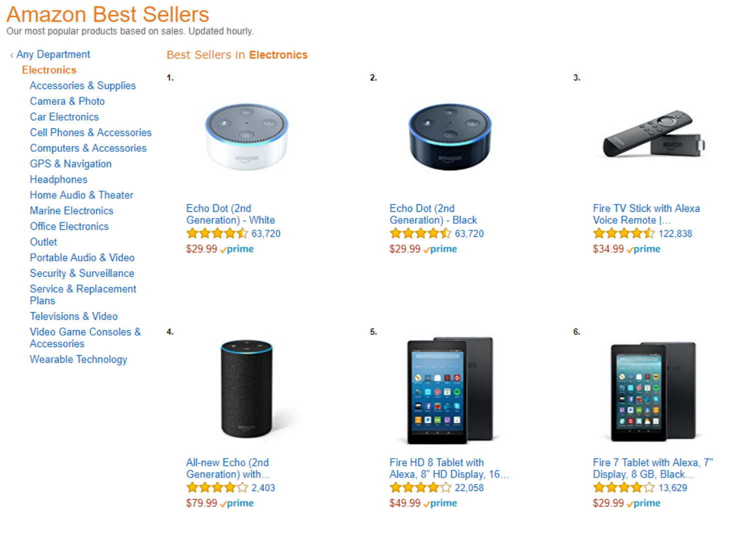
- Non-uniform Replication
- **Non-uniform CRDTs**
- Evaluation
- Conclusion and future work

Top-K with removals

Top-K with removals

- Defined as a set of tuples, $\langle$ id, score $\rangle$
- Supports two write operations
  - `ADD(id, score)`
  - `RMV(id)`

Top Sum



Amazon Best Sellers
Our most popular products based on sales. Updated hourly.
‹ Any Department
Electronics
Accessories & Supplies
Camera & Photo
Car Electronics
Cell Phones & Accessories
Computers & Accessories
GPS & Navigation
Headphones
Home Audio & Theater
Marine Electronics
Office Electronics
Outlet
Portable Audio & Video
Security & Surveillance
Service & Replacement Plans
Televisions & Video
Video Game Consoles & Accessories
Wearable Technology
Best Sellers in Electronics
1.
Echo Dot (2nd Generation) - White
63,720
$29.99 prime
2.
Echo Dot (2nd Generation) - Black
63,720
$29.99 prime
3.
Fire TV Stick with Alexa Voice Remote |...
122,838
$34.99 prime
4.
All-new Echo (2nd Generation) with...
2,403
$79.99 prime
5.
Fire HD 8 Tablet with Alexa, 8" HD Display, 16...
22,058
$49.99 prime
6.
Fire 7 Tablet with Alexa, 7" Display, 8 GB, Black...
13,629
$29.99 prime

Top Sum

- A mapping of: id $\mapsto$ value
- Supports one write operation
  - `ADD(id, value)`: increments the local value of `id` by the given `value`

Top-1 Sum

$$\left\{ \vphantom{\begin{matrix}a\\b\\c\\d\\e\end{matrix}} \qquad\qquad \right\} \left\{ \vphantom{\begin{matrix}a\\b\\c\\d\\e\end{matrix}} \qquad\qquad \right\}$$

Top-1 Sum

```
ADD(Echo, 100) @ 1
```

$$\left\{ \begin{array}{l} \text{Echo} \mapsto 100 \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{l} \quad\quad\quad\quad \\ \\ \\ \end{array} \right\}$$

Top-1 Sum

```
ADD(Echo, 100) @ 1
```

$$\left\{ \text{Echo} \mapsto 100 \right\} \quad \left\{ \text{Echo} \mapsto 100 \right\}$$

Top-1 Sum

$$\left\{ \begin{array}{l} \text{Echo} \mapsto 100 \\ \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{l} \text{Echo} \mapsto 100 \\ \\ \\ \end{array} \right\}$$

Top-1 Sum

`ADD(Fire, 25)` @ 1

$$\left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 25 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \\ \\ \end{array} \right\}$$

Top-1 Sum

`ADD(Fire, 25)` @ 1

$$\left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 50 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \\ \\ \end{array} \right\}$$

Top-1 Sum

$\texttt{ADD(Fire, 25)} @ 1$

$$\left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 50 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 50 \\ \\ \end{array} \right\}$$

Top-1 Sum

$$\left\{ \begin{array}{c} \textbf{Echo} \mapsto \textbf{100} \\ \text{Fire} \mapsto 50 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \textbf{Echo} \mapsto \textbf{100} \\ \text{Fire} \mapsto 50 \\ \\ \end{array} \right\}$$

Top-1 Sum

$\texttt{ADD(Fire, 30)} @ 1, \texttt{ADD(Fire, 30)} @ 2$

$$\left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 80 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \text{Echo} \mapsto 100 \\ \text{Fire} \mapsto 80 \\ \\ \end{array} \right\}$$

Top-1 Sum

$\texttt{ADD(Fire, 30)} @ 1, \texttt{ADD(Fire, 30)} @ 2$

$$\left\{ \begin{array}{c} \textbf{Fire} \mapsto \textbf{110} \\ \text{Echo} \mapsto 100 \\ \\ \end{array} \right\} \quad \left\{ \begin{array}{c} \textbf{Fire} \mapsto \textbf{110} \\ \text{Echo} \mapsto 100 \\ \\ \end{array} \right\}$$

Road Map

- Non-uniform Replication
- Non-uniform CRDTs
- **Evaluation**
- Conclusion and future work

Evaluation: Questions

- What questions do we want to answer with this evaluation?
- Do our designs reduce…
  - the amount of data transmitted?
  - the replica sizes?

Evaluation: Setup

- Performed by simulation
- Evaluation setup uses 5 replicas per object
- Replicas synchronize every 100 operations
- We compare our NuCRDTs with state-of-the-art CRDT designs

State-of-the-art CRDT designs

- We compare our designs with the following state-of-the-art CRDT designs:
  - Delta-based CRDTs, that maintain full object replicas efficiently by propagating updates as deltas of the state
  - Computational CRDTs (CCRDTs), that maintain non-uniform replicas using a state-based approach
- For the evaluation to be fair both our NuCRDT designs and the CCRDT designs were adjusted to support up to 2 replica faults

Top-K with removals: dissemination cost

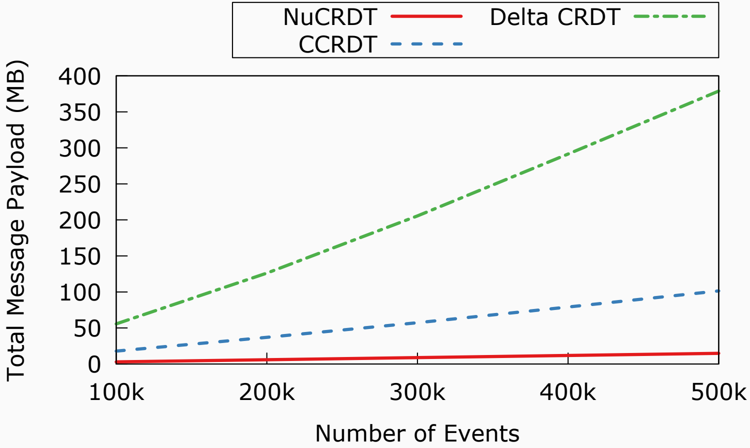


Figure 1: Total message size, workload of 95% adds

Top-K with removals: storage cost

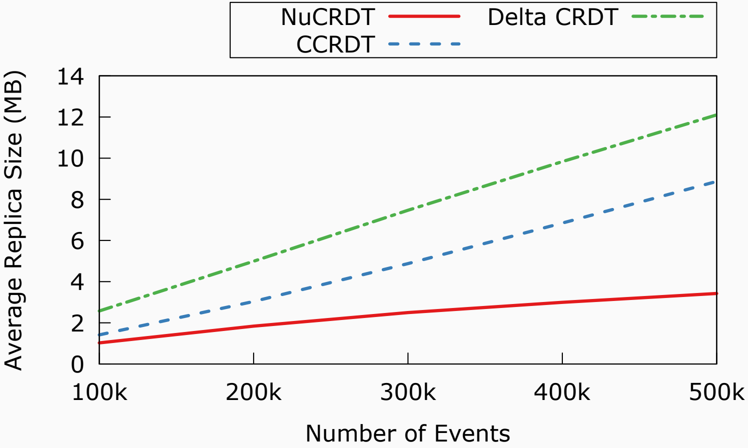


Figure 2: Mean replica size, workload of 95% adds

Top-K with removals: dissemination cost

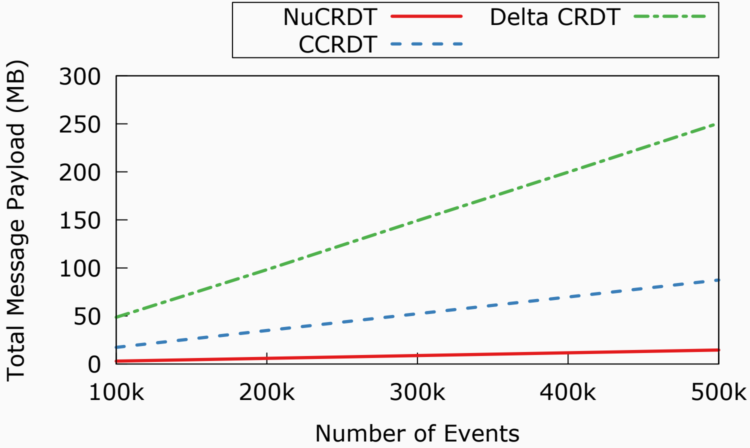


Figure 3: Total message size, workload of 99.95% adds

Top-K with removals: storage cost

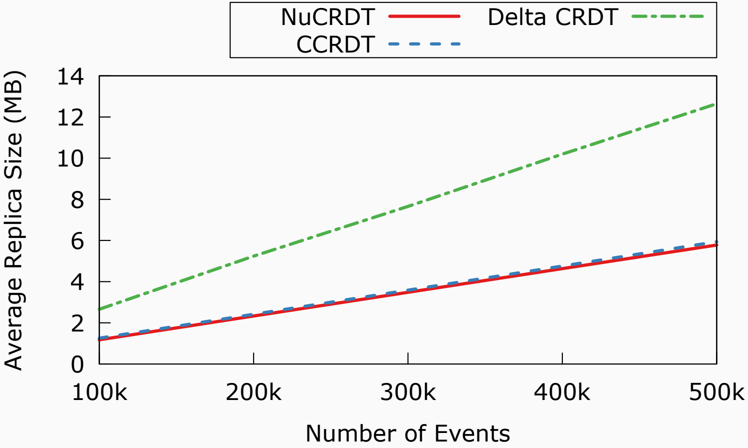


Figure 4: Mean replica size, workload of 99.95% adds

Top Sum: dissemination cost

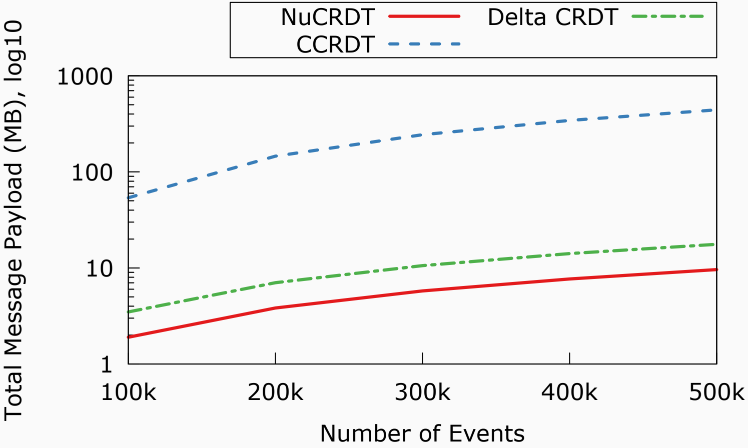


Figure 5: Total message size

Top Sum: storage cost

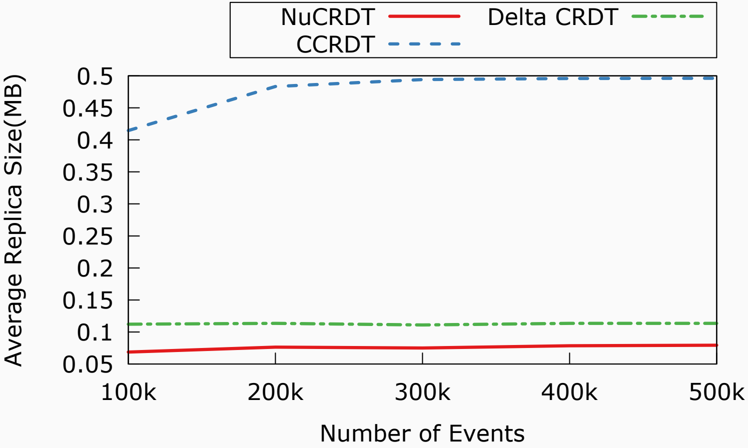


Figure 6: Mean replica size

Road Map

- Non-uniform Replication
- Non-uniform CRDTs
- Evaluation
- **Conclusion and future work**

Conclusion

- Introduced the non-uniform replication model and formalized its semantics for an eventually consistent system
- Showed how the model can be applied to CRDTs
- Compared our NuCRDT designs with state-of-the-art CRDT alternatives via simulation, showing the gains in network bandwidth and storage space

Future work

- Study the applicability of this replication model to stronger consistency models, such as linearizability
- Design other data types that benefit from this model

Questions?