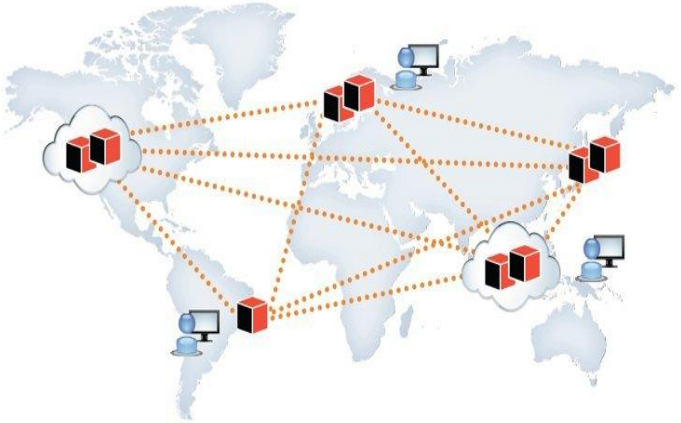
Undoing CRDT Operations Automatically

Provakar Mondal and Eli Tilevich
Software Innovations Lab, Virginia Tech, USA
{provakar, tilevich}@cs.vt.edu

Background

Modern distributed applications replicate data to different geo-locations to achieve:

- Low latency
- High availability
- Scalability
- Fault tolerance



Examples: Google Docs, Calendar, Figma, Facebook's OpenR, etc.

Background

- Distributed State Synchronization to ensure consistency
- Eventual Consistency
- Conflict-free Replicated Data Type (CRDT)

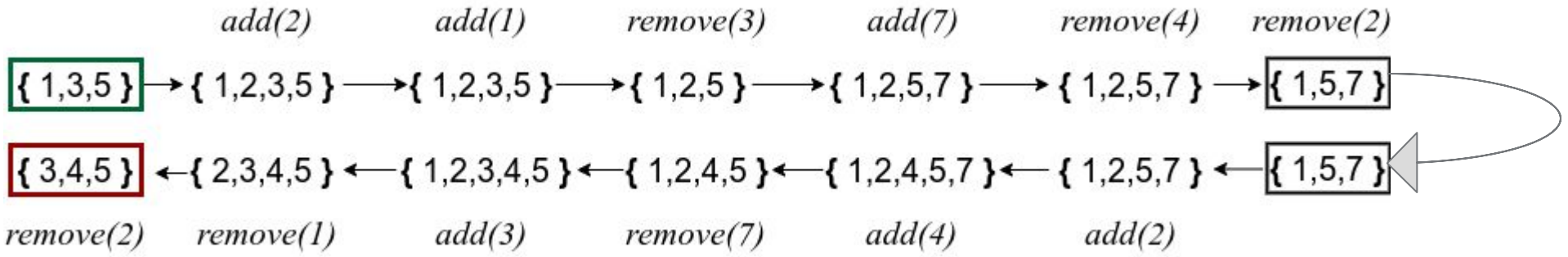
Problem Motivation

- Program errors cause *correct* updates of *incorrect* data
- CRDTs lack built-in *undo* functionality
- Ad-hoc undo implementations hard to reuse
- Modifying CRDT code by hand hurts portability

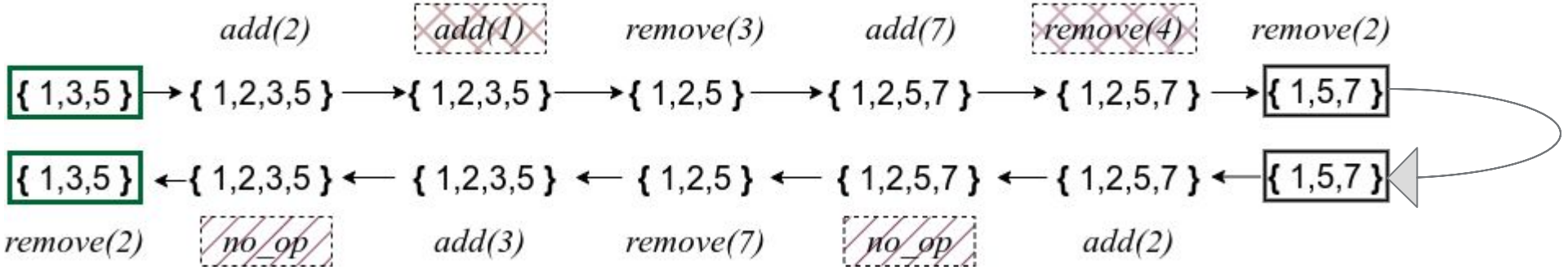
Paper's Contribution

- AUTO-UNDO
- Declarative metadata
  - undo pairs
  - no_op operation
- Undo Script
- No manual modifications of CRDT library

Motivating Example



Incorrect Undo Procedure



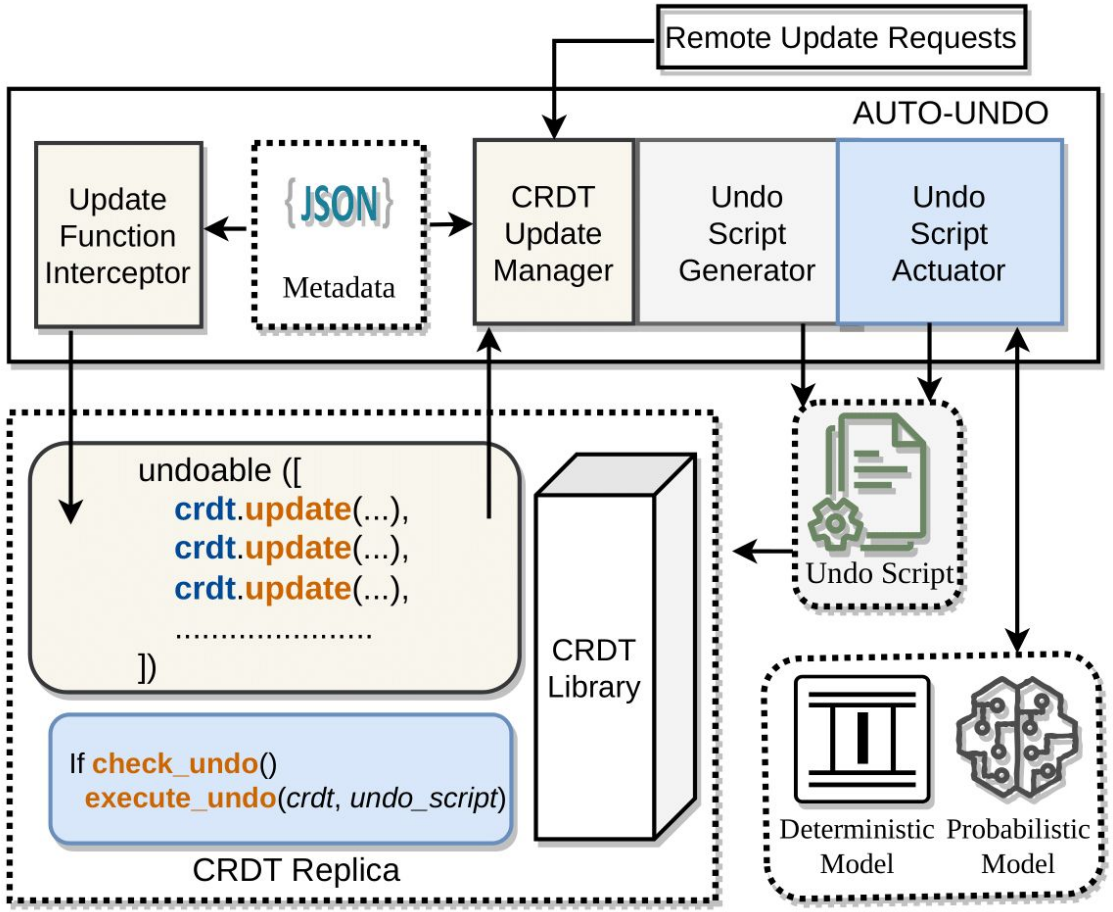
no_op Undo Procedure

AUTO-UNDO: Key Components

Four key components.

1. Update Function Interceptor
2. Update Manager
3. Undo Script Generator
4. Undo Script Actuator

Key Components of AUTO-UNDO



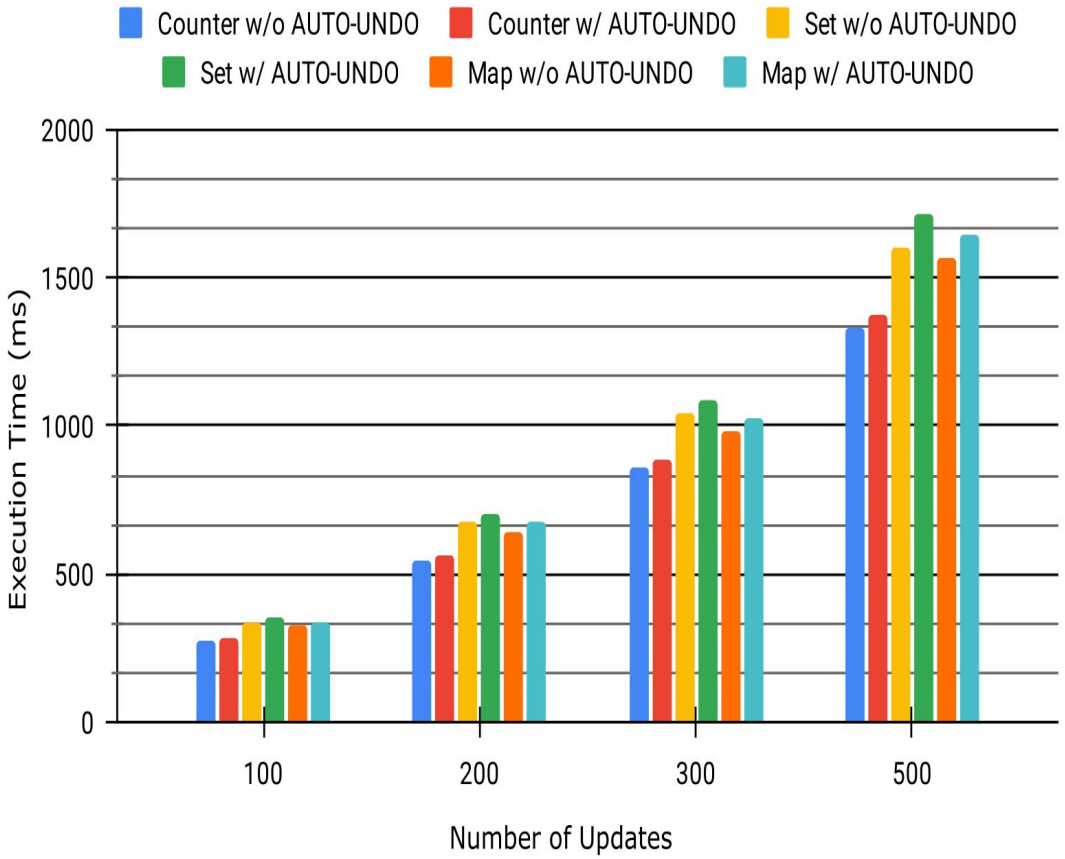
Undo Generation and Actuation with AUTO - UNDO

Evaluation

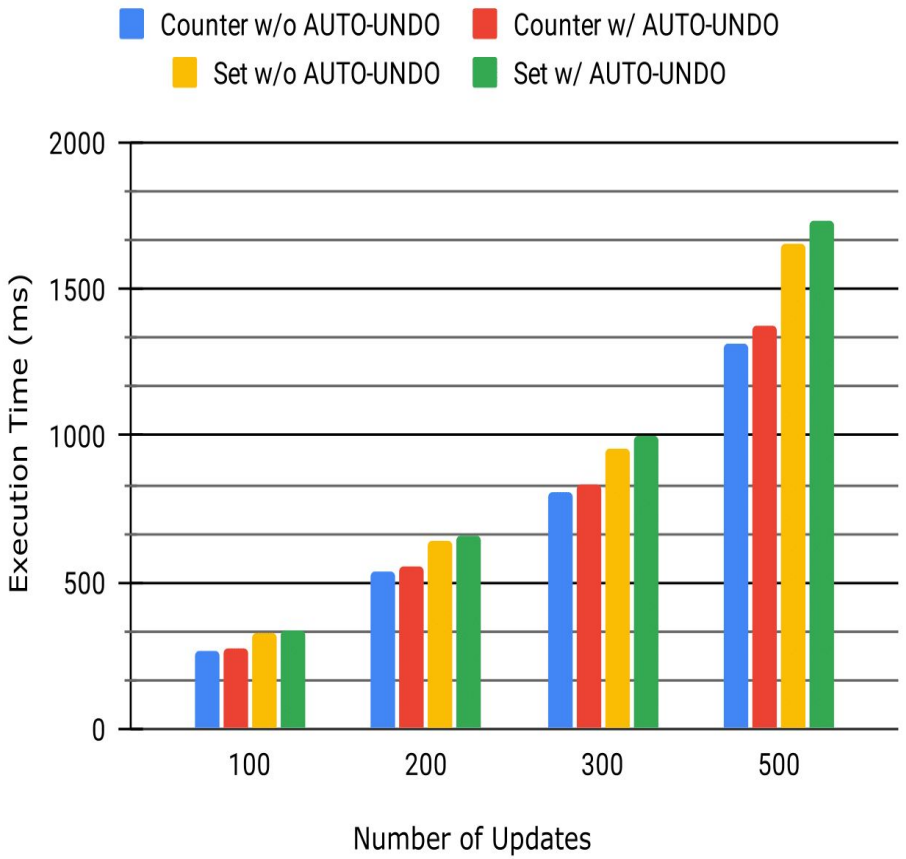
RQ1: What is the performance overhead of applying AUTO - UNDO to add the undo capability in existing CRDT libraries under different workloads?

RQ2: What is the accuracy/performance trade-offs between the deterministic and probabilistic models for determining whether to execute an undo procedure?

RQ1: Performance Overhead



Overhead for *js-delta-crdts* library



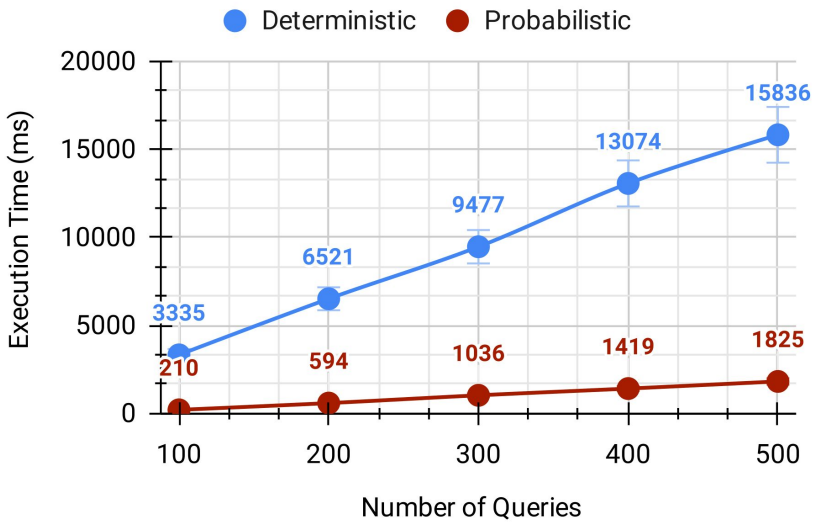
Overhead for *crdts* library

RQ1: Performance Overhead

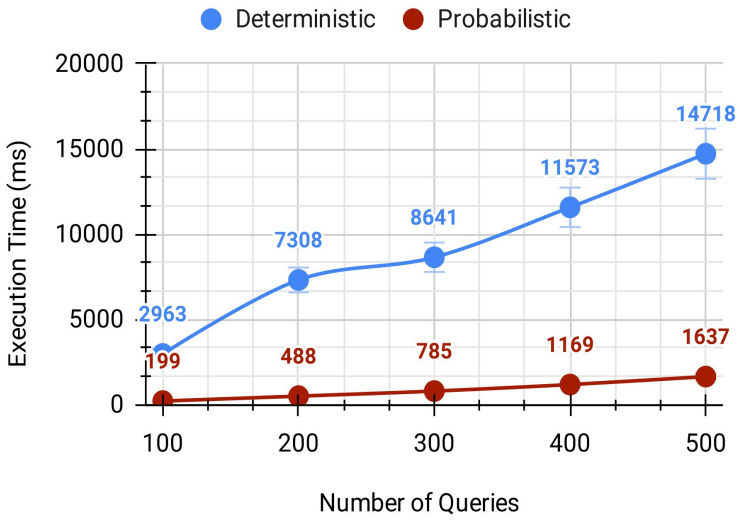
Number of Updates	*js-delta-crdts*			*crdts*	
	Counter	Set	Map	Counter	Set
100	2.93	3.87	3.36	2.64	3.08
200	2.19	3.98	4.18	3.13	3.45
300	3.27	4.32	4.37	3.49	4.07
500	3.61	4.94	4.78	4.33	4.56

Latency Overhead (%)

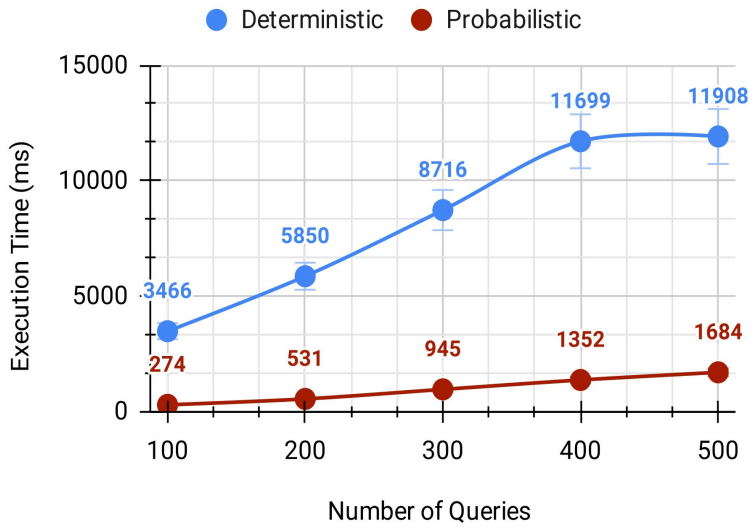
RQ2: Accuracy/Performance Trade-off



Add-Remove Set (js-delta-crdts)



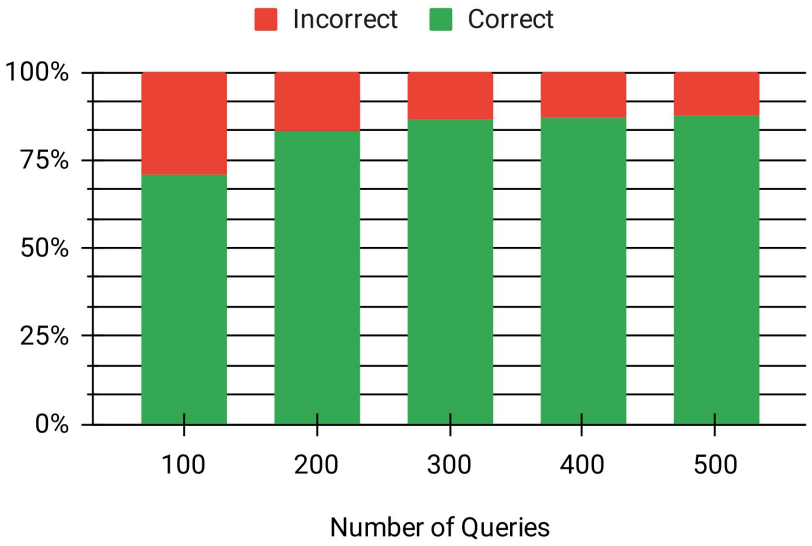
Map (js-delta-crdts)



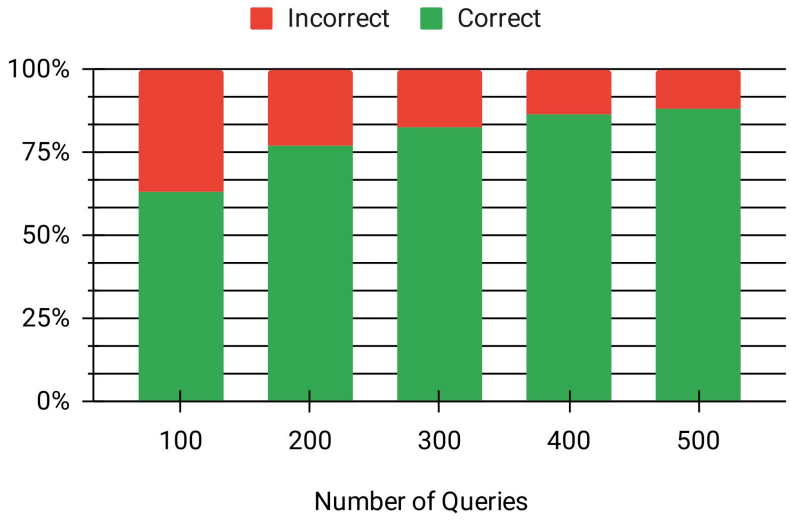
Add-Remove Set (crdts)

Latency Comparison of Deterministic vs Probabilistic Model

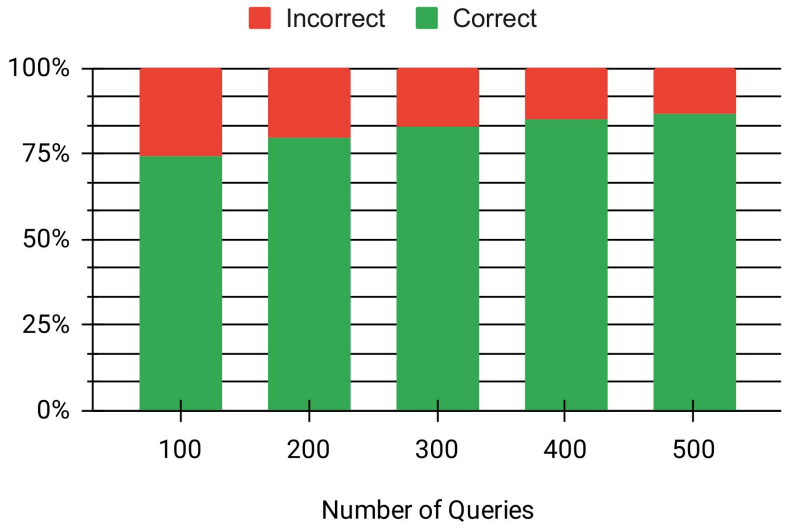
RQ2: Accuracy/Performance Trade-off



Add-Remove Set (js-delta-crdts)



Map (js-delta-crdts)



Add-Remove Set (crdts)

Correct/Incorrect Ratio for the Probabilistic Model

Conclusion

- An automatic enhancement of CRDTs with the undo capability
- Undo functionality external to CRDT library
- Application of declarative meta-programming to CRDT
- Low performance overhead
- Ability to trade off correctness and performance