Specification, Implementation, and Complexity of Replicated Data Types with Composite Operations

Hengfeng Wei
State Key Laboratory for Novel Software Technology, Nanjing University, China
hfwei@nju.edu.cn
https://orcid.org/0000-0002-0427-9710

Huang Yu
State Key Laboratory for Novel Software Technology, Nanjing University, China
yuhuang@nju.edu.cn
https://orcid.org/0000-0001-8921-036X

Jian Lu
State Key Laboratory for Novel Software Technology, Nanjing University, China
lj@nju.edu.cn

Abstract

2012 ACM Subject Classification

Keywords and phrases Replicated set

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

2 Preliminaries

2.1 Observed-Remove Set (`OR-Set`)

$$\mathcal{F}_{\texttt{orset}}(\texttt{rd}, E, \mathsf{op}, \mathsf{vis}, \mathsf{ar}) = \{a \mid \exists e \in E.\mathsf{op}(e) = \texttt{add}(a) \tag{1}$$

$$\land \left(\forall f \in E.\mathsf{op}(f) = \texttt{rm}(a) \implies \neg(e \xrightarrow{\mathsf{vis}} f)\right)\}. \tag{2}$$

3 Replicated Data Types with Composite Operations

3.1 Specification

We consider a composite operation of a replicated data type τ in the form of $C = A \oplus B$, where A, B, and C are different objects of type τ.

Q : Generalize to different data types?

Following [1], we specify the semantics of a composite operation $A \oplus B$ of a replicated data type τ by a function $\mathcal{F}_\tau$ that determines the return value of $\oplus$ based on prior operations performed on the two objects involved (i.e., A and B). However, $\mathcal{F}_\tau$ for a composite operation $\oplus$ takes as parameters two, not one as in [1], *operation contexts*, one on each object involved.

Note: Partial operation context [2]

▶ **Definition 1** (Product of Operation Contexts)**.** Consider two operation contexts for the same replicated data type τ:

$$\mathcal{C}_A = (E_A, \mathsf{op}_A, \mathsf{vis}_A, \mathsf{ar}_A) \tag{3}$$

$$\mathcal{C}_B = (E_B, \mathsf{op}_B, \mathsf{vis}_B, \mathsf{ar}_B) \tag{4}$$

The product $\mathcal{C} = \mathcal{C}_A \times \mathcal{C}_B$ of $\mathcal{C}_A$ and $\mathcal{C}_B$ is also an operation context defined as $\mathcal{C} = (E, \mathsf{op}, \mathsf{vis}, \mathsf{ar})$, where

- $E = E_A \times E_B$
- $\mathsf{op} = \mathsf{op}_A \sqcup \mathsf{op}_B$
- $\mathsf{vis} = \mathsf{vis}_A \times \mathsf{vis}_B$
- $\mathsf{ar} = \mathsf{ar}_A \times \mathsf{ar}_B$

▶ **Definition 2.**

$$\mathcal{F}_\tau(\oplus, \mathcal{C}_A, \mathcal{C}_B) = \mathcal{F}_\tau(\oplus, \mathcal{C}_A \times \mathcal{C}_B) \tag{5}$$

4 Replicated Set with Composite Operations

We consider the replicated set data type with composite operations including union ($\cup$), intersection ($\cap$), and set difference ($\setminus$).

4.1 Specification

General method for set union, set intersection, and set difference: There are four $\cdots$

$$\mathcal{F}_{\mathtt{orset}}(A \setminus B, \mathcal{C}_A, \mathcal{C}_B) \tag{6}$$

$$= \{a \mid \exists (e, e') \in E_A \times E_B. \Big(\big(\mathtt{op}(e) = \mathtt{add}(a) \land \mathtt{op}(e') = \mathtt{rm}(a)\big) \tag{7}$$

$$\land \forall (f, f') \in E_A \times E_B. \big((\mathtt{op}(f) = \mathtt{add}(a) \land \mathtt{op}(f') = \mathtt{add}(a)) \lor (\mathtt{op}(f) = \mathtt{rm}(a) \land \mathtt{op}(f') = \mathtt{add}(a \tag{8}$$

$$\implies \neg \big((e, e') \xrightarrow{\mathsf{vis}} (f, f')\big)\Big)\}. \tag{9}$$

▶ **Example 3** (Specification for Composite Operations of Replicated Set)**.**

4.2 Protocol

4.2.1 Methodology

Op-based

We always assume that $(\mathbb{A} \cup \mathbb{R}) \cap (\mathbb{A}' \cup \mathbb{R}') = \emptyset$. Note that for any replica state $(a, \mathbb{A}, \mathbb{R})$, we have $\mathbb{R} \subseteq \mathbb{A}$.

The goal is to find $\Box$ and $\Diamond$ such that

$$\cdots \iff \Box \subseteq \Diamond$$

4.2.2 Set Union ($\cup$)

▶ **Theorem 4** (Set Union)**.** *Given that $\mathbb{R} \subseteq \mathbb{A}$ and $\mathbb{R}' \subseteq \mathbb{A}'$, we have*

$$(\mathbb{R} \subset \mathbb{A}) \lor (\mathbb{R}' \subset \mathbb{A}') \iff (\mathbb{R} \cup \mathbb{R}') \subset (\mathbb{A} \cup \mathbb{A}').$$

$$\begin{array}{lcl} U & : & \text{universe of elements} \\ I & : & \text{set of identifiers} \\ \mathbb{A}, \mathbb{R} & \in & \text{SUBSET } \mathcal{I} \\ \Sigma & = & \mathcal{U} \times \mathbb{A} \times \mathbb{R} \\ \sigma_0 & = & \lambda r.\, \emptyset \end{array}$$

Figure 1 Implementation of replicated set with composite operations.

4.2.3 Set Intersection ($\cap$)

The goal is to find $\Box$ and $\Diamond$ such that

Another way: IF THEN ELSE

$$(\mathbb{R} \subset \mathbb{A}) \vee (\mathbb{R}' \subset \mathbb{A}') \iff \Box \subset \Diamond$$

▶ **Theorem 5** (Set Intersection). *Given that* $\mathbb{R} \subseteq \mathbb{A}$ *and* $\mathbb{R}' \subseteq \mathbb{A}'$*, we have*

$Q : \emptyset$ issues

$$(\mathbb{R} \subset \mathbb{A}) \wedge (\mathbb{R}' \subset \mathbb{A}') \iff (\mathbb{A} \times \mathbb{R}') \cup (\mathbb{R} \times \mathbb{A}') \subset (\mathbb{A} \times \mathbb{A}').$$

4.2.4 Set Difference ($\setminus$)

The goal is to find $\Box$ and $\Diamond$ such that

$$(\mathbb{R} \subset \mathbb{A}) \wedge (\mathbb{R}' \not\subset \mathbb{A}') \iff \Box \subset \Diamond$$

▶ **Theorem 6** (Set Difference). *Given that* $\mathbb{R} \subseteq \mathbb{A}$ *and* $\mathbb{R}' \subseteq \mathbb{A}'$*, we have*

$Q : \emptyset$ issues

$$(\mathbb{R} \subset \mathbb{A}) \wedge (\mathbb{R}' \not\subset \mathbb{A}') \iff (\mathbb{R} \times \mathbb{A}') \subset (\mathbb{A} \times \mathbb{R}').$$

5 Related Work

[3]

6 Conclusion and Future Work

References

1. Sebastian Burckhardt, Alexey Gotsman, Hongseok Yang, and Marek Zawirski. Replicated data types: Specification, verification, optimality. In *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '14, pages 271–284, New York, NY, USA, 2014. ACM. URL: `http://doi.acm.org/10.1145/2535838.2535848`, `doi:10.1145/2535838.2535848`.
2. Alexey Gotsman and Hongseok Yang. Composite replicated data types. In *European Symposium on Programming Languages and Systems*, pages 585–609. Springer, 2015.
3. Christopher Meiklejohn and Peter Van Roy. Lasp: A language for distributed, coordination-free programming. In *Proceedings of the 17th International Symposium on Principles and Practice of Declarative Programming*, PPDP '15, pages 184–195, New York, NY, USA, 2015. ACM. URL: `http://doi.acm.org/10.1145/2790449.2790525`, `doi:10.1145/2790449.2790525`.