Artifact: A Computational Interpretation of Compact Closed Categories: Reversible Programming with Negative and Fractional Types

Chao-Hong Chen and Amr Sabry

October 2, 2020

This document contains a list of claims in the paper and corresponding code.

2 Core Reversible Language: Π

- The syntax of Π in Fig.1 is formalized in **Pi/Syntax.agda**.

2.1 Abstract Machine Semantics

- The δ function in Fig.2 is defined in **Pi/Opsem.agda:L47**.
- The well-formed continuation stacks in Fig.3 is defined in **Pi/Opsem.agda:L71**.
- The machine states in Def.1 is defined in **Pi/Opsem.agda:L81**.
- The reduction relation in Fig.4 is defined in **Pi/Opsem.agda:L86**.
- Lem.2 is proved in **Pi/NoRepeat.agda:L13**.
- Lem.3 is proved in **Pi/NoRepeat.agda:L29**.
- Def.4 is defined in **Pi/Eval.agda:L76**.
- Def.5 is defined in **Pi/Eval.agda:L121**.
- Thm.6 is proved in **Pi/Properties.agda:L31**.

2.2 Interpreter

- The interpreter in Fig.5 is defined in **Pi/Interp.agda:L9**.
- Thm.7 is proved in **Pi/Properties.agda:L49**.

3 Termination of Reversible Abstract Machines

- The reversible abstract machine in Def.8 is defined in **RevMachine.agda:L8**.
- Lem.9 is proved in **RevNoRepeat.agda:L112**.
- Lem.10 is proved in **Pi/Eval.agda:L13**.
- Thm.11 is proved in **Pi/Eval.agda:L76**.
- The reversible abstract machine in Def.12 is defined in **RevMachine.agda:L15**.
- Lem.13 is proved in **PartialRevNoRepeat.agda:L123**.

4 Space and Time Resources and Trade-Offs

5 Negative Types: $\prod^m$

6 Fractional Types: $\prod^d$

7 Combining Negative and Fractional Types: $\prod^{\mathbb{Q}}$

8 Programming with Negative and Fractional Types