Mapping Proofs between Weak Memory Concurrency

CS4560 - Parallel and Concurrent Programming

February, 2026

Project description

In this project, you will prove the correctness of a mapping scheme from a programming language to an underlying architecture considering the respective consistency model [1, 2, 3, 4] .
[1]

Background Information

Concurrency Models

We consider release-acquire as the programming language consistency model and a simplied-Arm as the architecture model.

Release-acquire model In this model, the accesses are release-write (W_{REL}), acquire-read (R_{ACQ}), and acquire-release atomic-update ($[\mathsf{R}_{\text{ACQ}}]; \mathsf{rmw}; [\mathsf{W}_{\text{REL}}]$).

Definitions:

$$\mathsf{sw} = [\mathsf{W}_{\text{REL}}]; \mathsf{rf}; [\mathsf{R}_{\text{ACQ}}]$$
$$\mathsf{hb} = (\mathsf{po} \cup \mathsf{sw})^+$$

The axioms are

- $\mathsf{hb}; (\mathsf{rf} \cup \mathsf{mo} \cup \mathsf{fr})^*$ is irreflexive.
- $(\mathsf{poloc} \cup \mathsf{rf} \cup \mathsf{mo} \cup \mathsf{fr})$ is acyclic.
- $\mathsf{rmw}; (\mathsf{fr}; \mathsf{mo}) = \emptyset$.

Simplified Arm model In this model, the memory accesses are write, read, atomic-update. The model has three types of fences – (i) load fence (F_{RM}) that orders a read with a po-successor read or write (ii) store-fence (F_{WW}) that orders a pair of write events.

Definitions:

$$\mathsf{ob} \triangleq (\mathsf{bob} \cup \mathsf{rfe} \cup \mathsf{moe} \cup \mathsf{fre})^+ \text{ where}$$
$$\mathsf{bob} \triangleq (\mathsf{po}; [\mathsf{F}]; \mathsf{po}) \cup ([\mathsf{R}]; \mathsf{po}; [\mathsf{F}_{\text{RM}}]; \mathsf{po}) \cup ([\mathsf{W}]; \mathsf{po}; [\mathsf{F}_{\text{WW}}]; \mathsf{po}; [\mathsf{W}])$$

The axioms are

- ob is irreflexive.
- $(\mathsf{poloc} \cup \mathsf{rf} \cup \mathsf{mo} \cup \mathsf{fr})$ is acyclic.
- $\mathsf{rmw}; (\mathsf{fr}; \mathsf{mo}) = \emptyset$.

Note that we consider atomic update as a single event)

For both models, we define behavior as

$$\mathsf{Behavior}(X) \triangleq \{(location(e), Value(e)) \mid \exists e \in X.W.\ [\{e\}]; X.\mathsf{mo} = \emptyset\}$$

[1] The project description is subject to small changes and updates. Please get in touch with the TA's and the teachers if you have any questions.

Mapping Scheme

We propose a mapping scheme as follows

- $\mathsf{W}_{\text{REL}} \rightsquigarrow \mathsf{F}_{\text{WW}} \cdot \mathsf{W}$
- $\mathsf{R}_{\text{ACQ}} \rightsquigarrow \mathsf{R} \cdot \mathsf{F}_{\text{RM}}$

Mapping correctness proof

Prove the following theorem:

For each simple-Arm consistent execution, there exists a release-acquire consistent source execution such that both executions have the same behavior.

Projects

In this project, you will prove the theorem above in a proof assistant of your choice e.g. Coq, Agda, Lean.

Roadmap for the project:

The project involves the following steps:

- Set up the proof assistant.
- Define the events, relations, execution, and the other required definitions.
- Define the memory models with the respective axioms.
- Define the mapping scheme and the correctness theorem.
- Prove the theorem.

References

[1] Mark Batty, Scott Owens, Susmit Sarkar, Peter Sewell, and Tjark Weber. Mathematizing C++ concurrency. In *POPL'11*, pages 55–66. ACM, 2011.

[2] Anton Podkopaev, Ori Lahav, and Viktor Vafeiadis. Bridging the gap between programming languages and hardware weak memory models. *Proc. ACM Program. Lang.*, 3(POPL), 2019.

[3] Redha Gouicem, Dennis Sprokholt, Jasper Ruehl, Rodrigo C. O. Rocha, Tom Spink, Soham Chakraborty, and Pramod Bhatotia. Risotto: A dynamic binary translator for weak memory model architectures. ASPLOS 2023, page 107–122, 2022.

[4] C/C++11 mappings to processors. `https://www.cl.cam.ac.uk/~pes20/cpp/cpp0xmappings.html`.