Thomas Shull

Hardstrasse 201 Floor 17, 8005 Zürich CH, **email:** mail@tomshull.com, **mobile:** +41-76-499-46-13, thomasshull.net

Research Interests

Hardware and software designs to improve the performance of managed languages. Development of new persistent programming frameworks. Virtual Machine modifications to utilize emerging byte-addressable persistent memory technologies. Profiling-based compiler optimizations. Techniques to reduce the overhead of automatic memory management.

Education

University of Illinois at Urbana-Champaign August 2012 - August 2020
Ph.D. in Computer Science
Advisor: Prof. Josep Torrellas
Thesis: *Making Non-Volatile Memory Programmable*
Committee: Prof. Josep Torrellas, Prof. Jian Huang, Prof. David Padua, Prof. James Larus, and Prof. Steven Swanson

Washington University of St. Louis June 2008 - May 2012
B.Sc. in Computer Science and B.Sc. in Computer Engineering
Summa Cum Laude

Professional Experience

Oracle Labs June 2020 – Present

Senior Researcher with GraalVM Team

- Responsible for maintaining and improving the GraalVM AArch64 port.
- Added features to improve performance of Substrate VM (SVM) runtime operations. SVM is a framework and runtime environment for the ahead-of-time compilation of Java applications.

Arm Ltd. July 2019 – April 2020

Open Source Software

- Worked on AArch64 port of Substrate VM (SVM).
- Identified and fixed multiple bugs within the SVM AArch64 port.
- Added multiple performance improvements to SVM.
- Enabled SVM AArch64 runtime code installation.

Non-Volatile Memory Research

- Improved and fixed AArch64 port of the Persistent Memory Development Kit (PMDK). PMDK is the primarly framework available to create persistent applications leveraging byte-addressable non-volatile memory (NVM).
- Proposed ISA extensions to improve crash-consistent application performance by enabling more aggressive instruction reordering.
- Implemented proposed ISA extension within the gem5 simulator. Initial evaluation shows promising performance gains.

Selected Recent Publications

Execution Dependence Extension (EDE): ISA Support for Eliminating Fences
ISCA 2021
***Thomas Shull**, Ilias Vougioukas, Nikos Nikoleris, Wendy Elsasser, and Josep Torrellas*
By allowing loads and stores to complete out of order, relaxed memory models can provide significant performance improvements. Unfortunately, far too often coarse-grain fences must be inserted for fine-grain correctness requirements, thereby negating these performance gains. In this paper we propose a new technique to describe instruction-level orderings as *execution dependences* that can be encoded within the ISA, and show how hardware can be modified to honor such execution dependences.

AutoPersist: An Easy-To-Use Java NVM Framework Based on Reachability
PLDI 2019
***Thomas Shull**, Jian Huang, and Josep Torrellas*
Emerging NVM technologies have led to the creation of many frameworks to assist developers in creating persistent applications. Unfortunately, we find existing NVM frameworks are still too burdensome for programmers and require many markings. To remedy this, we propose AutoPersist, a new Java NVM framework which requires substantially fewer markings by relying on the JVM runtime to perform much of the heavy lifting of creating a persistent application.

Reusable Inline Caching for JavaScript Performance PLDI 2019
*Jiho Choi, **Thomas Shull**, and Josep Torrellas*
Fast JavaScript startup time is paramount to a user's web browsing experience. Unfortunately, in current implementations, most of the profiling results observed in previous executions cannot be reused to improve the startup time of subsequent runs. To fix this, we develop a new technique which allows inline caching data to be reused, significantly improving website intialization times.

Using Speculation to Reduce the Checking Overhead of Persistent Objects in NVM Frameworks VEE 2019
***Thomas Shull**, Jian Huang, and Josep Torrellas*
Emerging programmer-friendly NVM frameworks have many actions predicated on whether an object is persistent or not. We find that even if these actions are not taken, performing checks to determine an object's persistent state is very expensive. Based on online profiling information, we devise a technique to *bias* each persistent check towards its expected behavior, thereby minimizing performance overhead.

NoMap: Speeding-Up JavaScript Using Hardware Transactional Memory HPCA 2019
***Thomas Shull**, Jiho Choi, María J. Garzarán, and Josep Torrellas*
JavaScript implementations use multiple compilers to ensure "hot" code regions are fully optimized without prohibitive startup times. We recognize a signficant performance overhead in multi-tiered compiler codes is their inter-tier jump points, in spite of these jump points rarely being invoked. To this end, we propose to use hardware transactional memory to limit the number of inter-tier jump points needed, thereby allowing the compiler to generate more efficient code.

Biased Reference Counting: Minimizing Atomic Operations in Garbage Collection PACT 2018
*Jiho Choi, **Thomas Shull**, and Josep Torrellas*
We profile Apple's Swift reference counting implementation and find its use of atomic compare-and-swap operations for reference counting updates to be a significant source of overheads. Furthermore, we find often an object's counter is only updated by one thread. Based on these insights, we split each object's reference count into two counters: a *biased counter* which a single thread can update without atomic operations, and a *shared counter* which is updated atomically by all other threads.

Defining a High-level Programming Model for Emerging NVRAM Technologies ManLang 2018
***Thomas Shull**, Jian Huang, and Josep Torrellas*
We find that existing NVRAM framework models are inappropriate for managed languages. To correct this, we propose a new NVRAM programming model which offloads much of the heavy-lifting of creating a persistent application to the managed language implementation's runtime.

ShortCut: Architectural Support for Fast Object Access in Scripting Languages ISCA 2017
*Jiho Choi, **Thomas Shull**, and Josep Torrellas*
Popular JavaScript implementations use a technique for fast property accesses which requires a level of indirection and does not effectively utilize hardware branch predictors. To this end, we propose new hardware which is able to store (access site ⇒ final destination) mappings for better branch prediction and fold this software level of indirection. We also propose advanced hardware which is able to elide the entire object property lookup routine.

Technical Skills

Programming Skills: C/C++, Java, Python.
Managed Language Implementations: JavaScript–V8, JavaScriptCore; Swift; Java–HotSpot, Maxine, Substrate VM
Compiler Implementations: Graal, LLVM

References

Christian Wimmer, christian.wimmer@oracle.com Oracle Labs
Stuart Monteith, stuart.monteith@arm.com Arm Ltd.
Josep Torrellas, torrella@illinois.edu University of Illinois at Urbana-Champaign
Additional references available upon request