Language-Based Optimizations for Persistence on Nonvolatile Main Memory Systems

Achievement: Designed several novel language-based optimization techniques for programming NVM as persistent memory and demonstrated them as an extension of our NVL-C programming system.

Significance and Impact: This work enhances our ability to efficiently utilize NVM as high-performance, persistent memory in HPC systems.

Research Details:
- We designed language-based techniques to automate shadow updates within NVM transactions to improve their performance and reduce their memory footprint.
- We created an abstract cost model for reasoning about the performance benefit of shadow updates.
- We designed an auto-tuned concrete cost model that enables the runtime to dynamically decide whether to perform a particular shadow update.
- We described our compiler-based approach to automate undo log aggregation, a key building block of shadow updates.
- We evaluated our NVL-C extensions on several applications with both real and emulated NVM hardware.

Sponsor/Facility: Work was performed on ExCL in CSMD.

PI and affiliation: Jeffrey S. Vetter from CSMD – Oak Ridge National Laboratory

Team: Joel E. Denny, Seyong Lee, and Jeffrey S. Vetter.


Overview:
Substantial advances in nonvolatile memory (NVM) technologies have motivated widespread integration of NVM into mobile, enterprise, and HPC systems. Recently, considerable research has focused on architectural integration of NVM and respective programming systems, exploiting NVM’s trait of persistence correctly and efficiently. In this regard, we design several novel language-based optimization techniques for programming NVM and demonstrate them as an extension of our NVL-C system. Specifically, we focus on optimizing the performance of atomic updates to complex data structures residing in NVM. We build on two variants of automatic undo logging: canonical undo logging, and shadow updates. We show these techniques can be implemented transparently and efficiently, using dynamic selection and other logging optimizations. Our empirical results on several applications gathered on an NVM test bed illustrate that our cost-model-based dynamic selection technique can accurately choose the best logging variant across different NVM modes and input sizes. In comparison to statically choosing canonical undo logging, this improvement reduces execution time to as little as 53% for block-addressable NVM and 73% for emulated byte-addressable NVM on a Fusion-io ioScale device.