Optimizing Post-Copy Live Migration with System-Level Checkpoint Using Fabric-Attached Memory

Chih Chieh Chou, Yuan Chen, Dejan Milojicic, A. L. Narasimha Reddy, and Paul V. Gratz

Nov 18, 2019
Outline

• Introduction
• Motivation and Goal
• Our Approach
• Conclusions
• Acknowledgements
Introduction

- Emerging **non-volatile memory** (NVM) has become a promising storage device due to:
  - Byte-addressability
  - Non-volatility
  - Low latency
  - Low idle power (except for NVDIMM)

HPE 8GB NVDIMM single Rank x4 DDR4-2133 Module
Introduction (cont.)

• Fabric-Attached Memory (FAM), which can be accessed by memory semantics, provides high bandwidth, low latency, and shared memory pool across machines in a rack scale.
Introduction of migrations

Source

App 1
App 2
App 3

Target

App 1
Introduction of migrations

- **Non-live** migration: the state of application is **checkpointed** entirely to storage devices, copied to target, and resumed at target.
- **Post-copy** live migration: Processor state, registers, etc., are transferred first, then application is resumed at target. When pages are accessed at target, a **page fault is triggered** to acquire faulty pages.
Migrations

Source

App 1

1

Storage/MEM

Target

App 1

3

Storage/MEM

2
Motivations

- Non-live migration can benefit from FAM directly by avoiding page-copying phase.

- What about post-copy migration? Can we do more by using FAM?
Goal

• Almost all previous work focus on total migration time of “victim” application
  – If we can predict the working set correctly, an approach with longer migration time might be better in terms of overall system performance

• Instead, we propose “busy time” (of source node): the time from the start of migration to the time “victim” can be killed at source node
  – Meaning how long the remaining applications at source having to wait for the resources, such as CPU and memory, occupied by “victim” to be released
  – Non-live migration has the optimal busy time
Our approach

• Like non-live migration, we propose our post-copy with FAM by first checkpointing the “victim” into FAM
  – Checkpointed-based post-copy migration
  – Therefore, “victim” can be killed after checkpoint immediately
    • Almost the same busy time as non-live migration
  – Due to the nature of FAM, the checkpointed pages can be accessed by target node directly
    • Achieve shorter latency of the page fault

• We have implemented our approach at CRIU (Checkpoint/Restart in Userspace), a Linux open source tool
Existing CRIU post-copy migration

- All Pages are stored in memory at source.
- Faulting pages transferred via socket interface to memory at target.
Our implementation

- On background, checkpoint “victim” into FAM
- Asynchronous accessed pages fault if pages are not ensured to be dumped
Our implementation (cont.)

• **Synchronously** accessed if pages have known to be dumped

```
  lazy page daemon
    Target
  app

  Lazy page file
  page
  FAM
```
Evaluation of benchmarks

- Source container
  - app

- Target container
  - app

- File
  - page

FAM (NVM)
System model

PCM (2 GB/s) or NVDIMM (6.6 GB/s)

Gen-Z fabric network

Add delay when R/W FAM

Source

Target

Add delay when transferring

Ethernet (10Gb/s or 40Gb/s)
Demanding paging

- NPB
- PARSEC
- SPLASH2X

- Socket total mig. time
- FAM (2GB/s) total mig. time
- FAM (2GB/s) busy time
Active pushing
Evaluation: Redis + YCSB

1. Insert records
2. post-copy migration
3. Access records
Results of YCSB throughput

One thread

- FAM (2GB/s)
- Socket (10Gb/s)
- FAM (6.6 GB/s)
- Socket (40Gb/s)

four threads

- FAM (2GB/s)
- Socket (10Gb/s)
- FAM (6.6GB/s)
- Socket (40Gb/s)
Results of speedup

- Socket (10Gb/s) total mig. time
- FAM (2Gb/s) total mig. time
- FAM (2Gb/s) busy time
- FAM total mig. time
- Socket total mig. Time
- FAM busy time
Acknowledgements

• We thank the generous support from Hewlett Packard Enterprise and National Science Foundation through I/UCRC (Industry–University Cooperative Research Centers) Program
Gen-Z DDR interface

- Ref: Gen-Z white paper: DRAM and Storage-Class Memory (SCM) Overview
Gen-Z memory interfaces

- Ref: Gen-Z white paper: DRAM and Storage-Class Memory (SCM) Overview