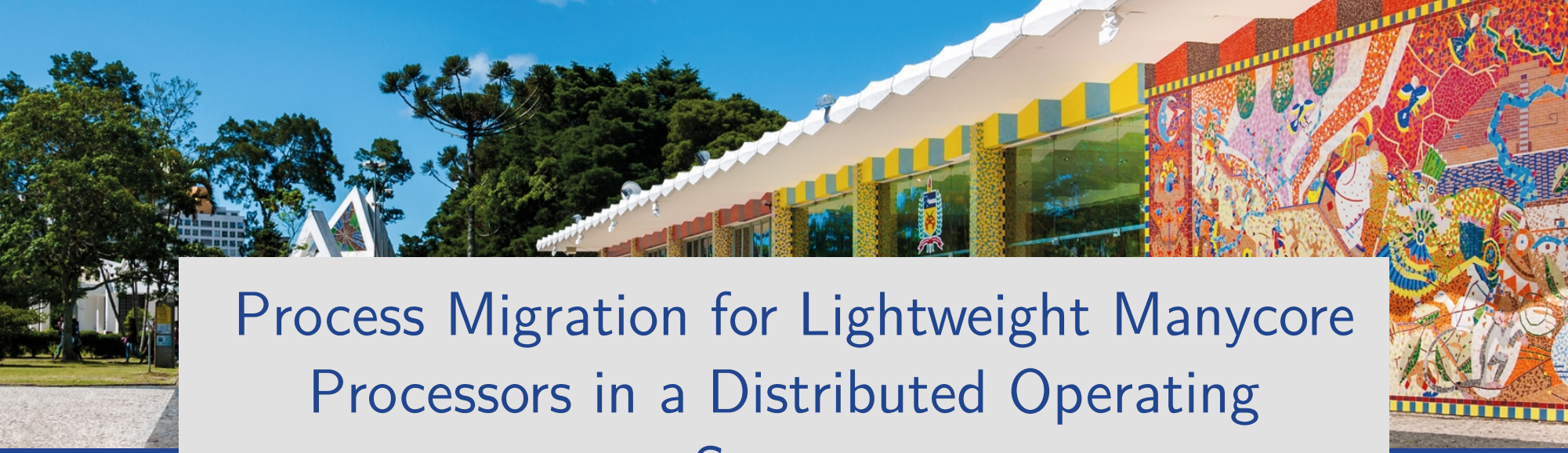


Process Migration for Lightweight Manycore Processors in a Distributed Operating System

João Vicente Souto

joao.vicente.souto@posgrad.ufsc.br

Parallel Computing - Computer Science
INE/UFSC, Florianópolis

September 28, 2020

LAPESD
INE UFSC

UFSC

Presentation Outline

LAPESD

Introduction

Process Migration

- **Transferring a process between machines**
- Proposed for:
  - Load balancing
  - Fault tolerance
  - System administration
  - Data access locality
- Arising of distributed systems
- Solutions established:
  - MOSIX (1985)
  - V (1988)
  - OSF/1 AD TNC (1995)

LAPESD

Process Migration

- Transferring a process between machines
- Proposed for:
  - Load balancing
  - Fault tolerance
  - Improved system administration
  - Data access locality
- Arising of **distributed systems**
- Solutions established:
  - MOSIX (1985)
  - V (1988)
  - OSF/1 AD TNC (1995)

LAPESD

Process Migration

- Despite the research efforts, **migration has not achieved widespread use**
- Treating residual dependency is a difficult task:
  - Opened files stored in the source node
  - Opened communicators with other processes
  - Shared resources
  - Internal kernel state.
- Solution: Virtualization

LAPESD

Process Migration

- Despite the research efforts, migration has not achieved widespread use
- **Treating residual dependency is a difficult task**:
  - Opened files stored in the source node
  - Opened communicators with other processes
  - Shared resources
  - Internal kernel state.
- Solution: Virtualization

LAPESD

Process Migration

- Despite the research efforts, migration has not achieved widespread use
- Treating residual dependency is a difficult task:
  - Opened files stored in the source node
  - Opened communicators with other processes
  - Shared resources
  - Internal kernel state.
- Solution: **Virtualization**

LAPESD

Virtualization

Virtualization

- Create a **Virtual Version** of Computing Resources
  - CPU
  - Memory
  - Storage device
  - Network device
- Features
  - Improved previous features
  - Power management
  - Isolation
  - Security
- Two types
  - Virtual Machines (VM)
  - Containers

LAPESD

Virtualization

- Create a Virtual Version of Computing Resources
  - CPU
  - Memory
  - Storage device
  - Network device
- **Features**
  - Improved previous features
  - Power management
  - Isolation
  - Security
- Two types
  - Virtual Machines (VM)
  - Containers

LAPESD

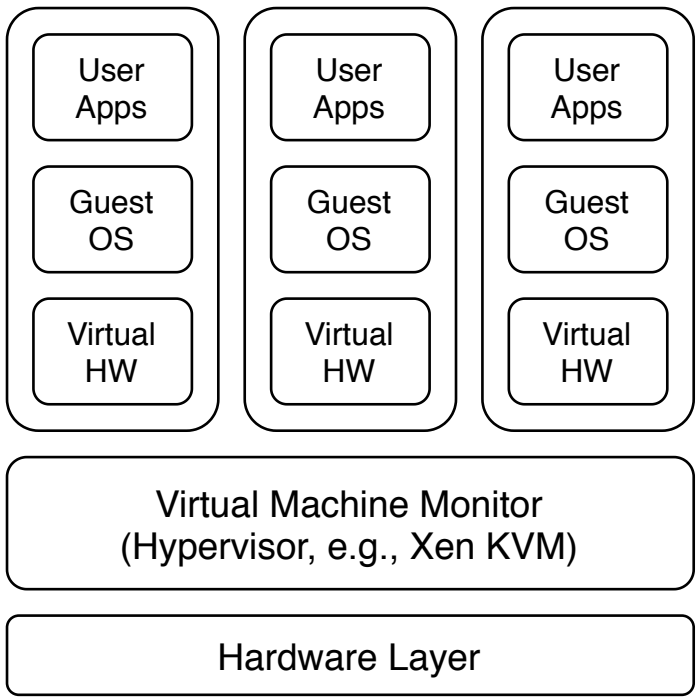
Virtualization

- Create a Virtual Version of Computing Resources
  - CPU
  - Memory
  - Storage device
  - Network device
- Features
  - Improved previous features
  - Power management
  - Isolation
  - Security
- **Two types**
  - Virtual Machines (VM)
  - Containers

LAPESD

Virtual Machines (VM)

- Encapsulates a whole operating system
- Depends on a Hypervisor to provide the hardware resources
  - Full virtualization
  - Paravirtualization



LAPESD

Migration of VMs

- **Memory content**
  - Hypervisor allocate memory
  - Guest OS allocate memory
  - Application requested memory
- Disk content
  - Hypervisor allocate blocks
  - Guest OS and Application used blocks

LAPESD

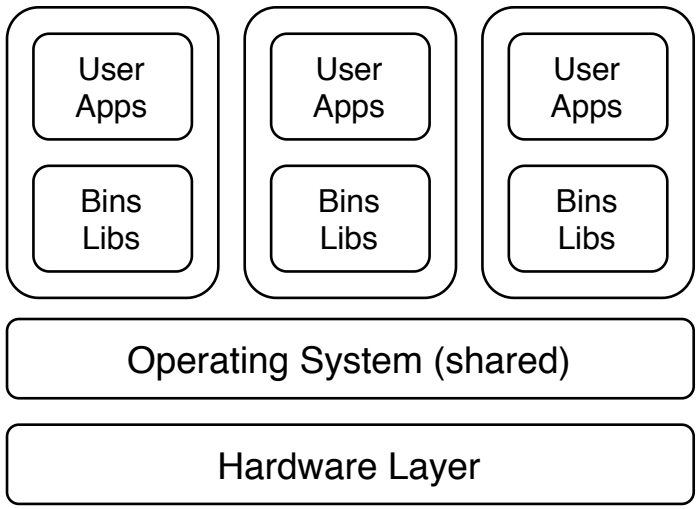
Migration of VMs

- Memory content
  - Hypervisor allocate memory
  - Guest OS allocate memory
  - Application requested memory
- **Disk content**
  - Hypervisor allocate blocks
  - Guest OS and Application used blocks

LAPESD

Containers

- OS-level virtualization
- Multiple isolated user space instances (Shared OS)
- Lightweight compared to VMs
  - Linux Containers
  - Docker



LAPESD

Migration of Containers

- Transfer the file system of the container
- Save the state of the container into a file (all processes and their resources)
- Transfer the container file
- Restart the container

LAPESD

Migration

Process Migration

- Generally, there are **three stages of memory transfer**:
  - Push Copy (iteration copy) stage
  - Stop-and-Copy stage
  - Pull Copy (On-demand copy) stage
- Migration techniques
  - Stop-and-Copy
  - *Pre-Copy*
  - *Post-Copy*
  - Hybrid

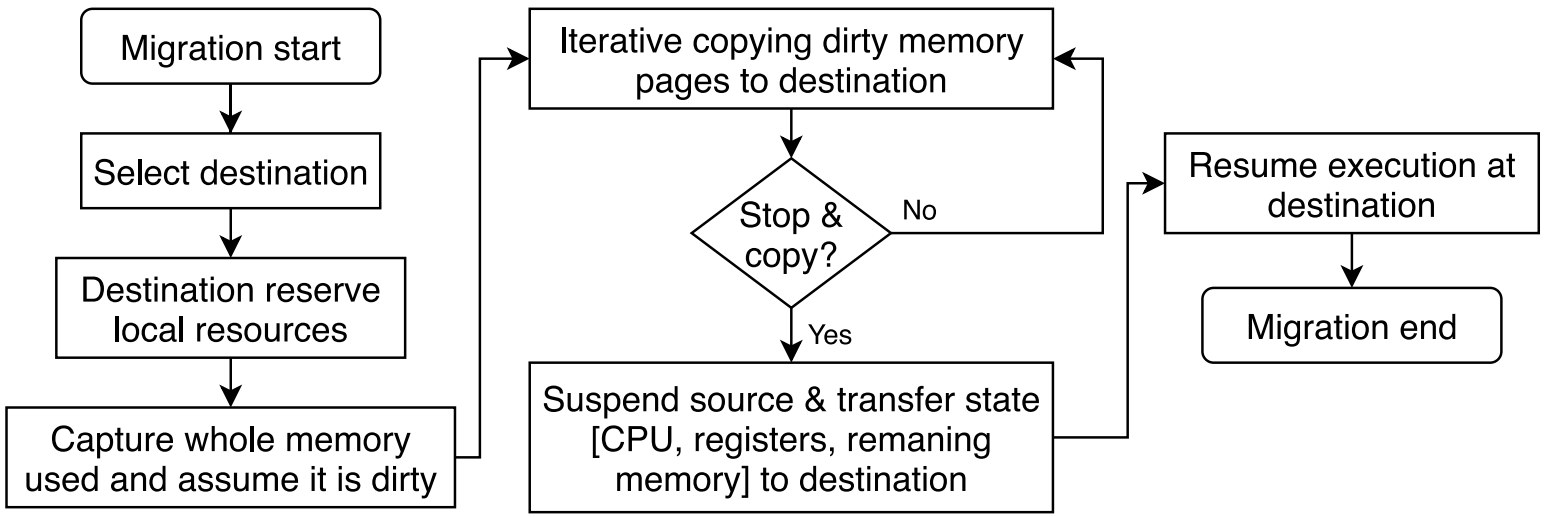
LAPESD

Process Migration

- Generally, there are three stages of memory transfer:
  - Push Copy (iteration copy) stage
  - Stop-and-Copy stage
  - Pull Copy (On-demand copy) stage
- **Migration techniques**:
  - Stop-and-Copy
  - ***Pre-Copy***
  - ***Post-Copy***
  - Hybrid

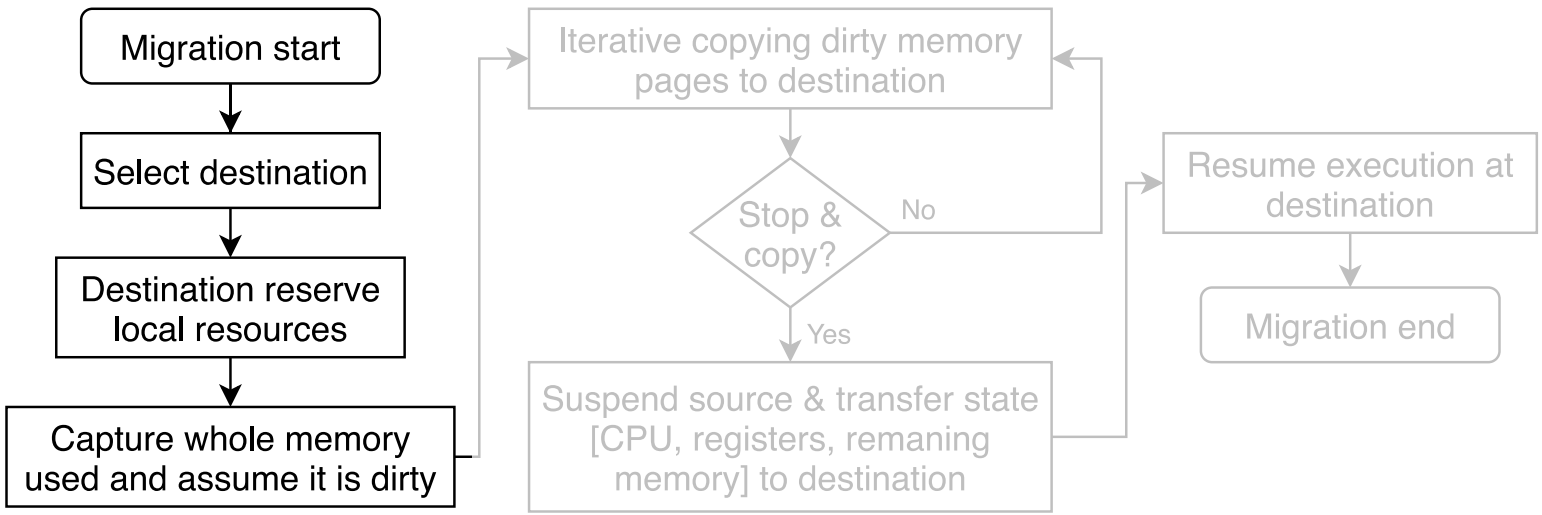
LAPESD

Pre-Copy Migration



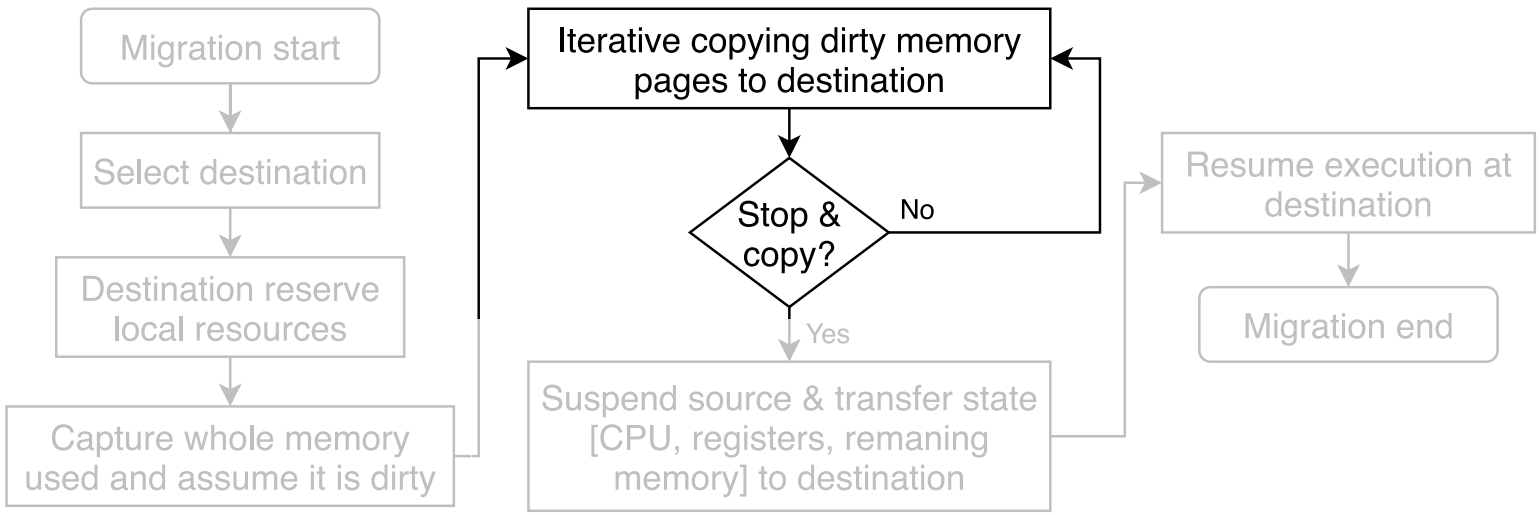
Pre-Copy Scenario

Pre-Copy Migration



Resource reservation and **transfer preparation**

Pre-Copy Migration

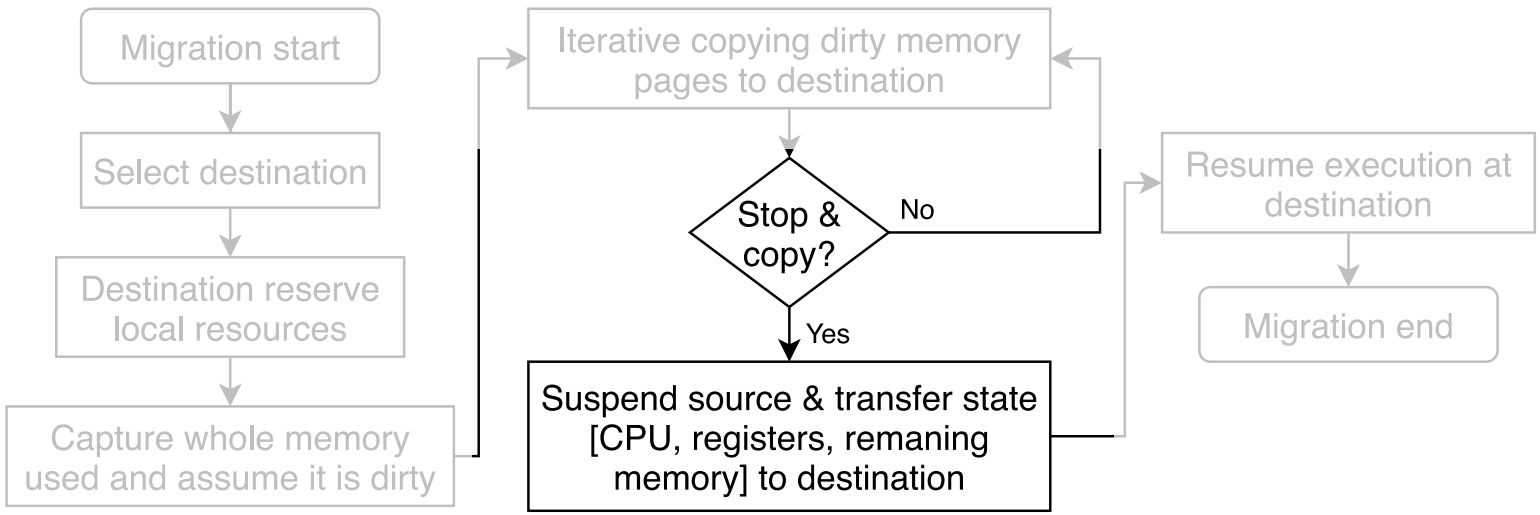


Sending the memory pages before the execution context

Push Copy Stage

LAPESD

Pre-Copy Migration

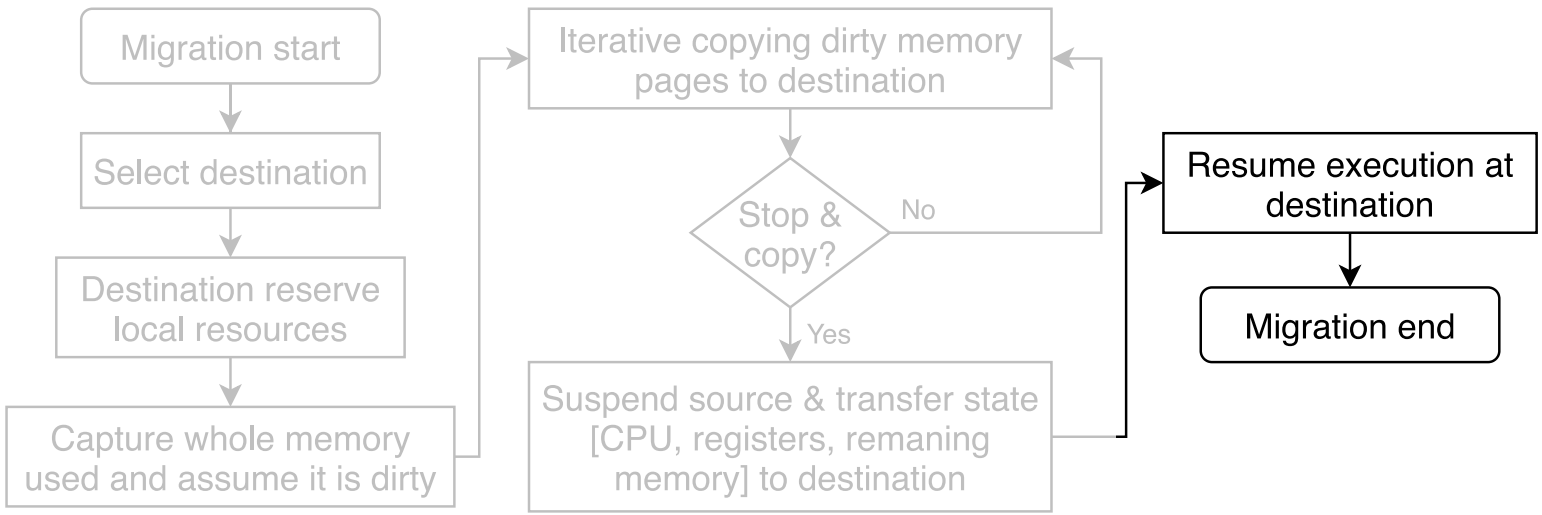


When enough memory has been transferred, send the execution context

Stop-and-Copy Stage

LAPESD

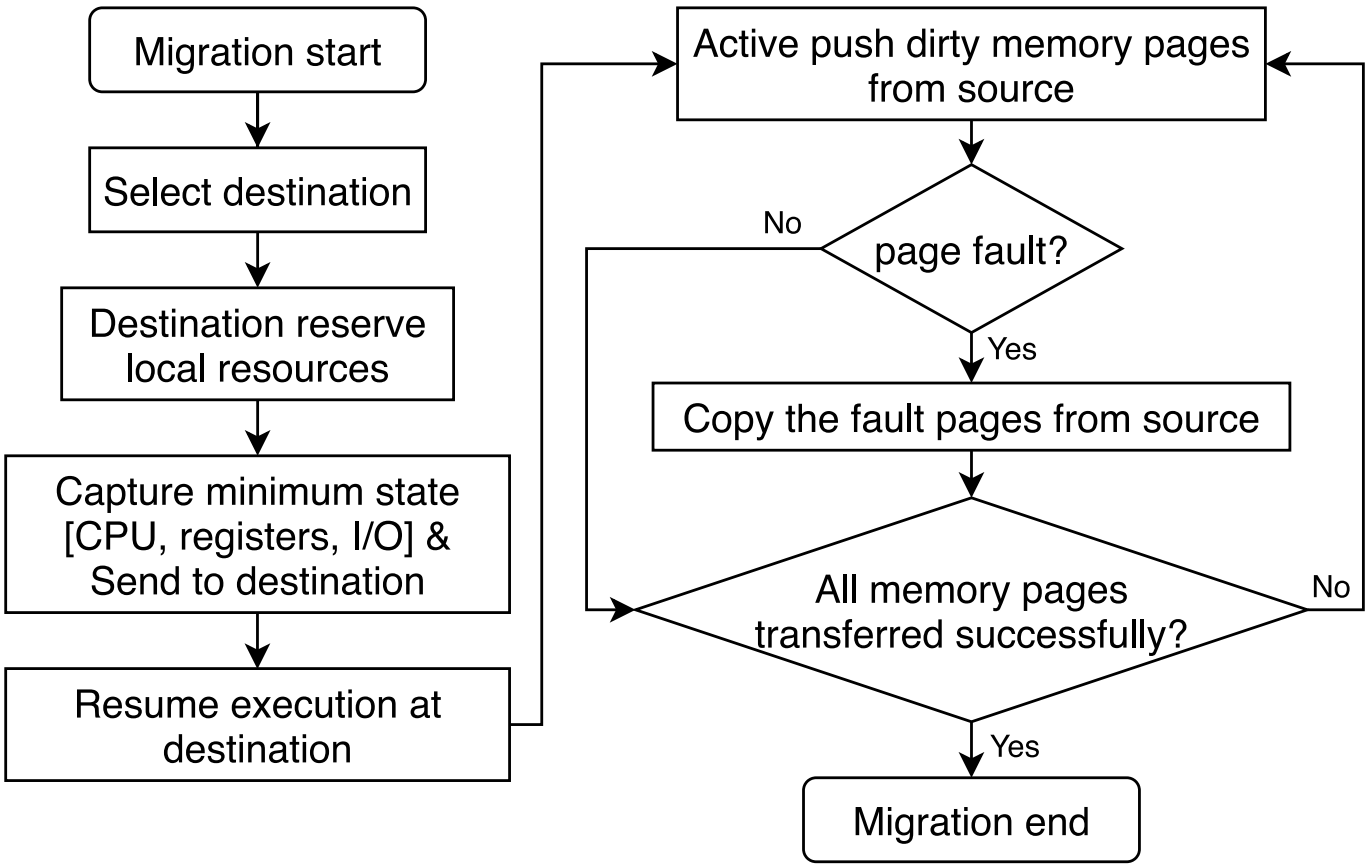
Pre-Copy Migration



Resume stopped execution context in the destination

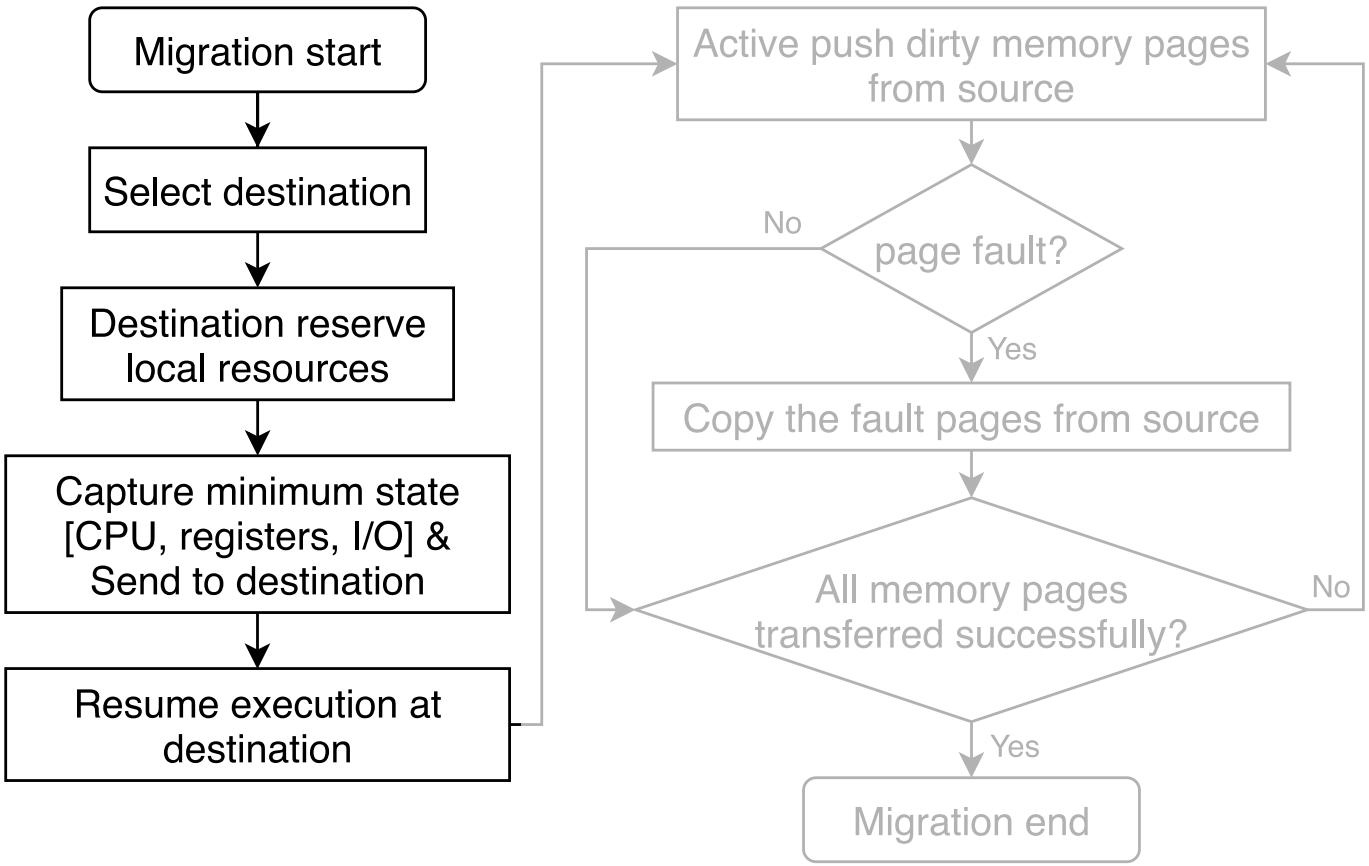
LAPESD

Post-Copy Migration



Post-Copy Scenario

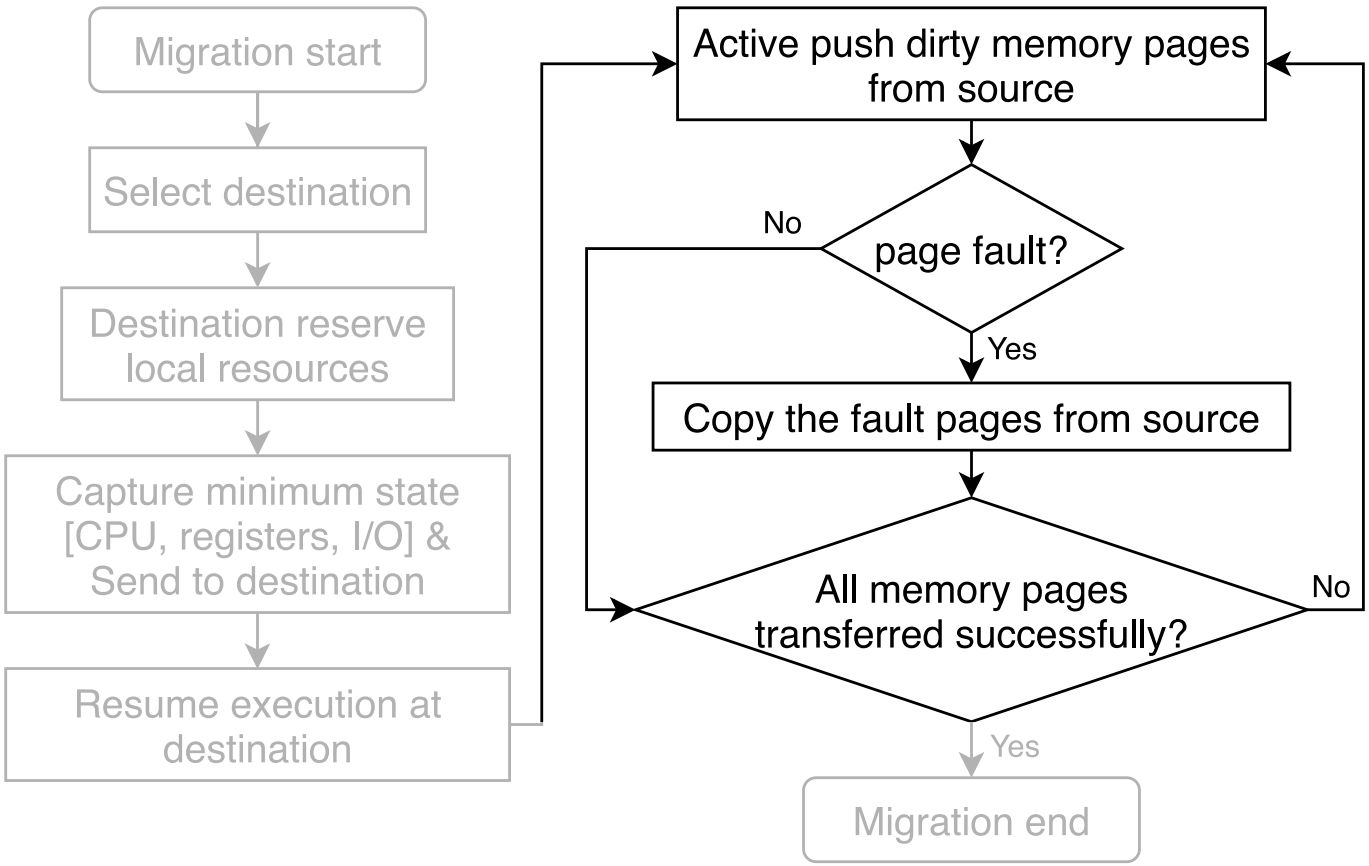
Post-Copy Migration



Sends execution context before memory pages

Stop-and-Copy Stage

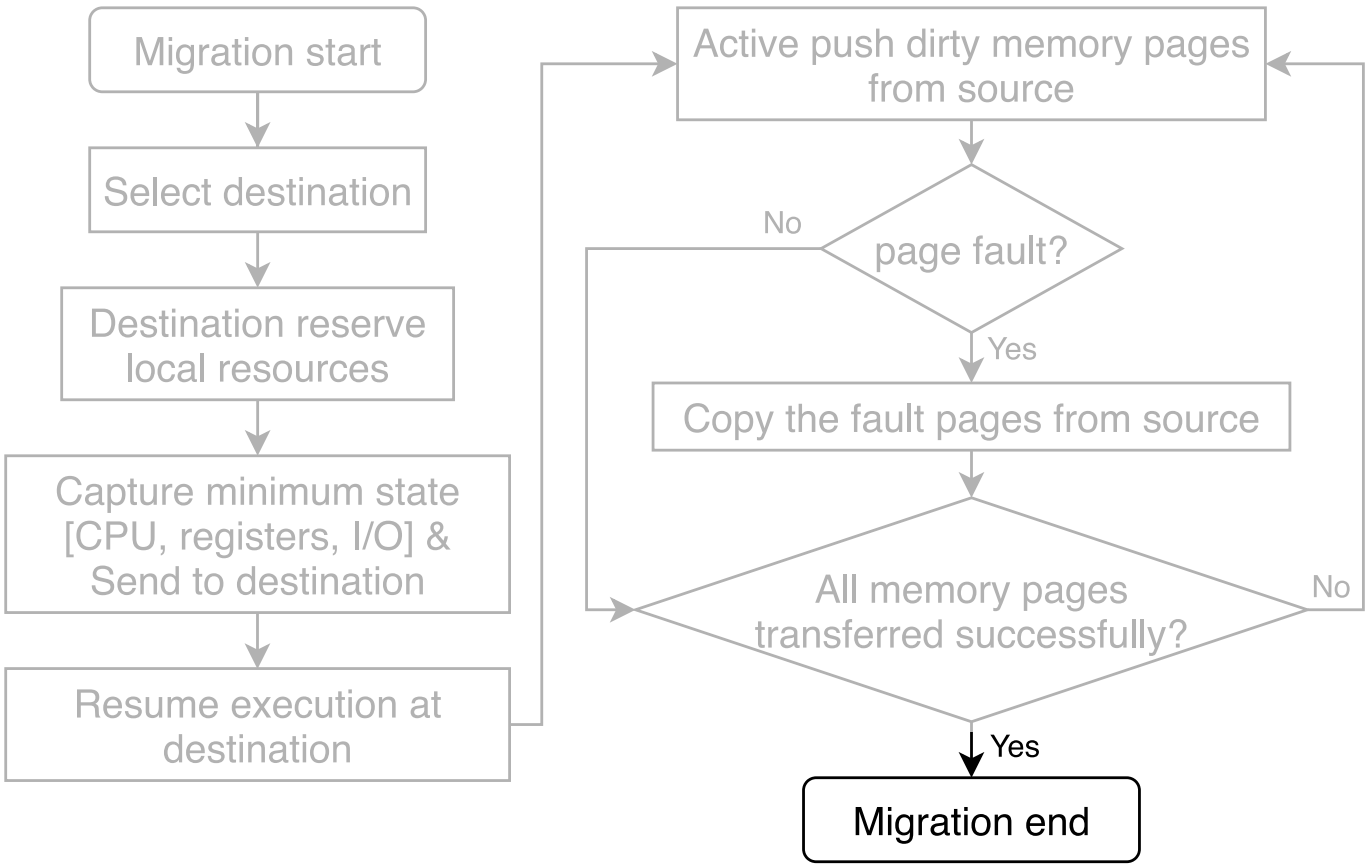
Post-Copy Migration



Merges **pull rest of memory pages** and any page that generate **page faults**

Pull Copy Stage

Post-Copy Migration



Continues execution normally

LAPESD

Metrics

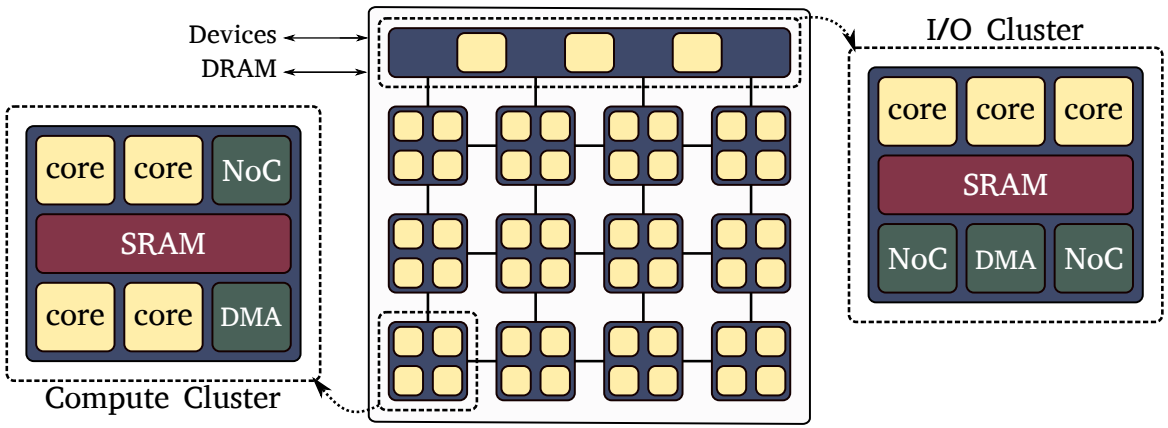
Performance Metrics

- Total Migration Time
- Downtime
- Pages Transfered
- Page Dirty Rate
- Preparation Time
- Resume Time
- Application Degradation
- Link Degradation

LAPESD

LW Processors

Lightweight Manycores Processors

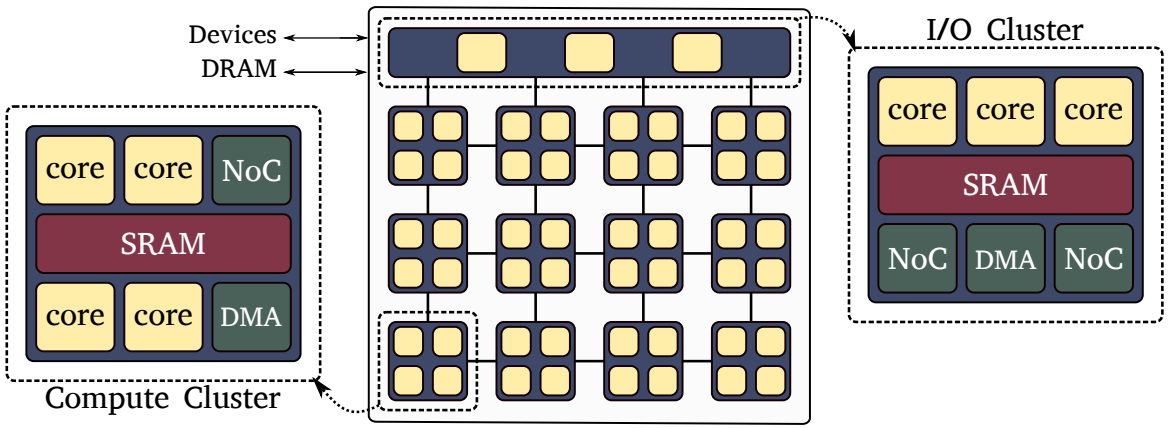


Overview of a Manycore

- **Hundreds of Lightweight Cores**
  - Expose Massive thread-level parallelism
  - Feature low-power consumption
  - Target MIMD workloads
- Distributed Memory Architecture
- On-Chip Heterogeneity

LAPESD

Lightweight Manycores Processors

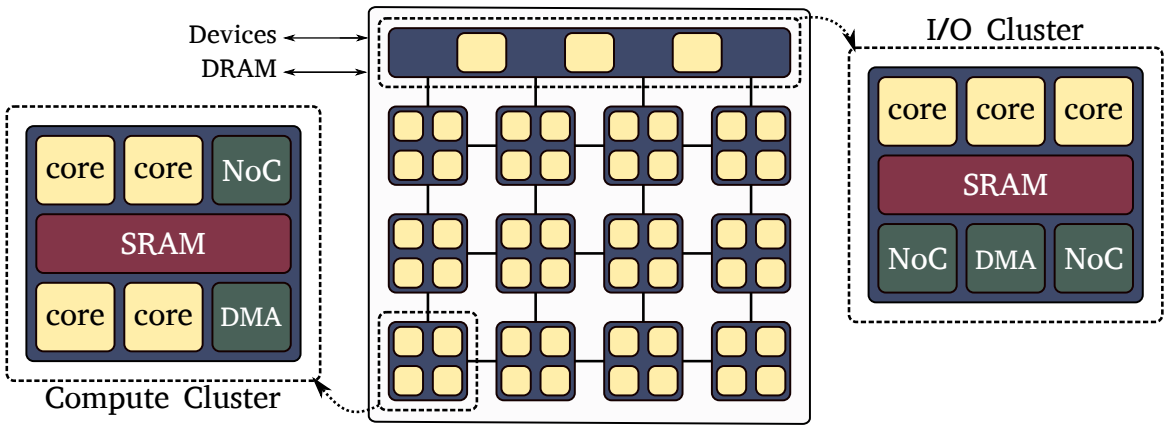


Overview of a Manycore

- Hundreds of Lightweight Cores
- **Distributed Memory Architecture**
  - Grants scalability
  - Relies on a Network-on-Chip (NoC)
  - Has constrained memory systems
- On-Chip Heterogeneity

LAPESD

Lightweight Manycores Processors



Overview of a Manycore

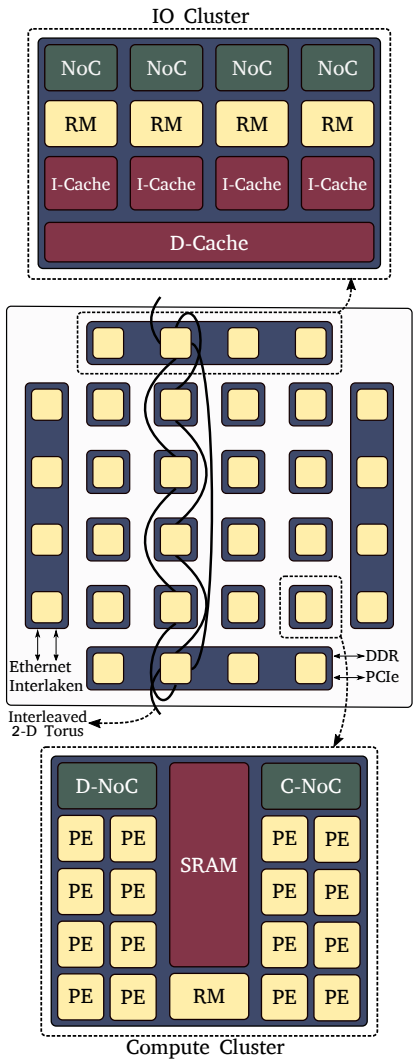
- Hundreds of Lightweight Cores
- Distributed Memory Architecture
- **On-Chip Heterogeneity**
  - Features different components

LAPESD

Kalray MPPA-256

A Lightweight Manycore Processor

- **288 processing cores**
  - 16 Compute Cluster (CC)
  - 4 I/O Cluster (IO)
- **Data NoC (D-NoC)**
  - 256 RX slots
  - 8 TX channels
  - 8 μthreads for async TX
- **Control NoC (C-NoC)**
  - 128 RX slots
  - 4 TX channels



Motivation And Justify

Conclusions

- **Motivation**
  - Historical evolution from single-cores to Lightweight Manycores
- **Contribution**
  - A Inter-Cluster Communication Facility for LW Processors
- **Results**
  - **Optimal sizes** for large data transfers
  - Well-known **distributed algorithms can be efficiently supported** by Nanvix OS
- **Future Works on Nanvix OS**
  - Remove limitation on asynchronous send
  - MPI port (BSc dissertation)
  - Shared Memory Service (MSc dissertation)
  - **Distributed Process Scheduling (MSc dissertation)**

LAPESD

Thank you!
Questions?

João Vicente Souto

joao.vicente.souto@posgrad.ufsc.br

Parallel Computing - Computer Science
INE/UFSC, Florianópolis

September 28, 2020

LAPESD
INE UFSC

UFSC

References I