Memory Aggregation For KVM

Hecatonchire Project

Benoit Hudzia; Sr. Researcher; SAP Research Belfast
With the contribution of Aidan Shribman, Roei Tell, Steve Walsh, Peter Izsak

November 2012

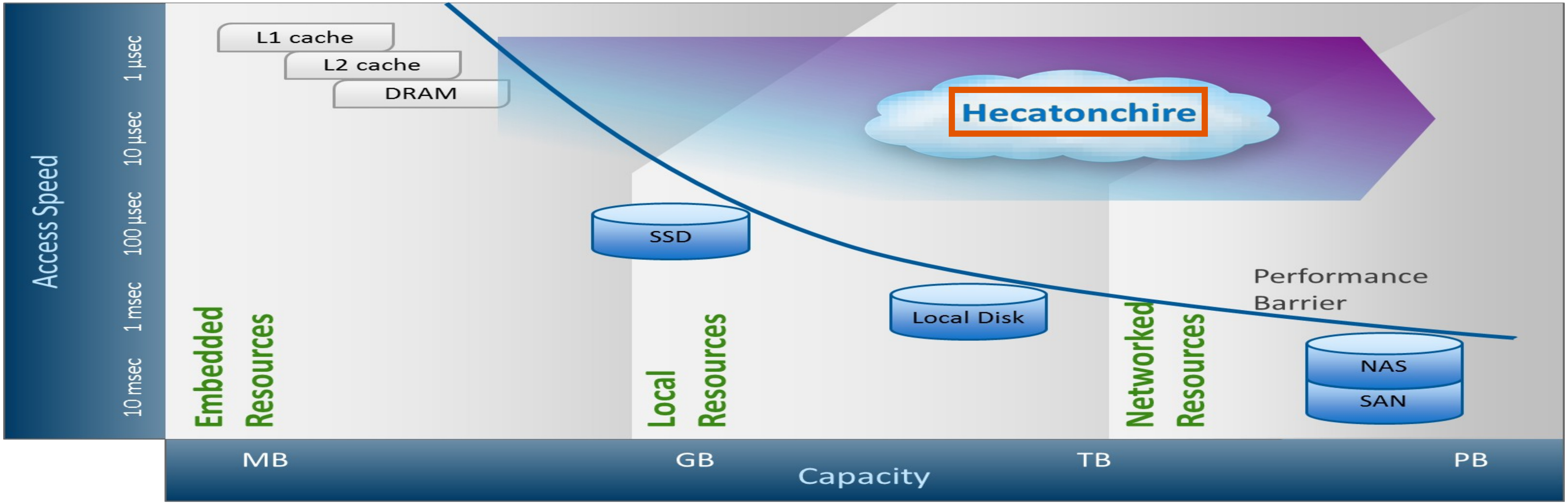
Agenda

- **Memory as a Utility**
- **Raw Performance**
- **First Use Case : Post Copy**
- **Second Use case : Memory aggregation**
- **Lego Cloud**
- **Summary**

Memory as a Utility

How we Liquefied Memory Resources

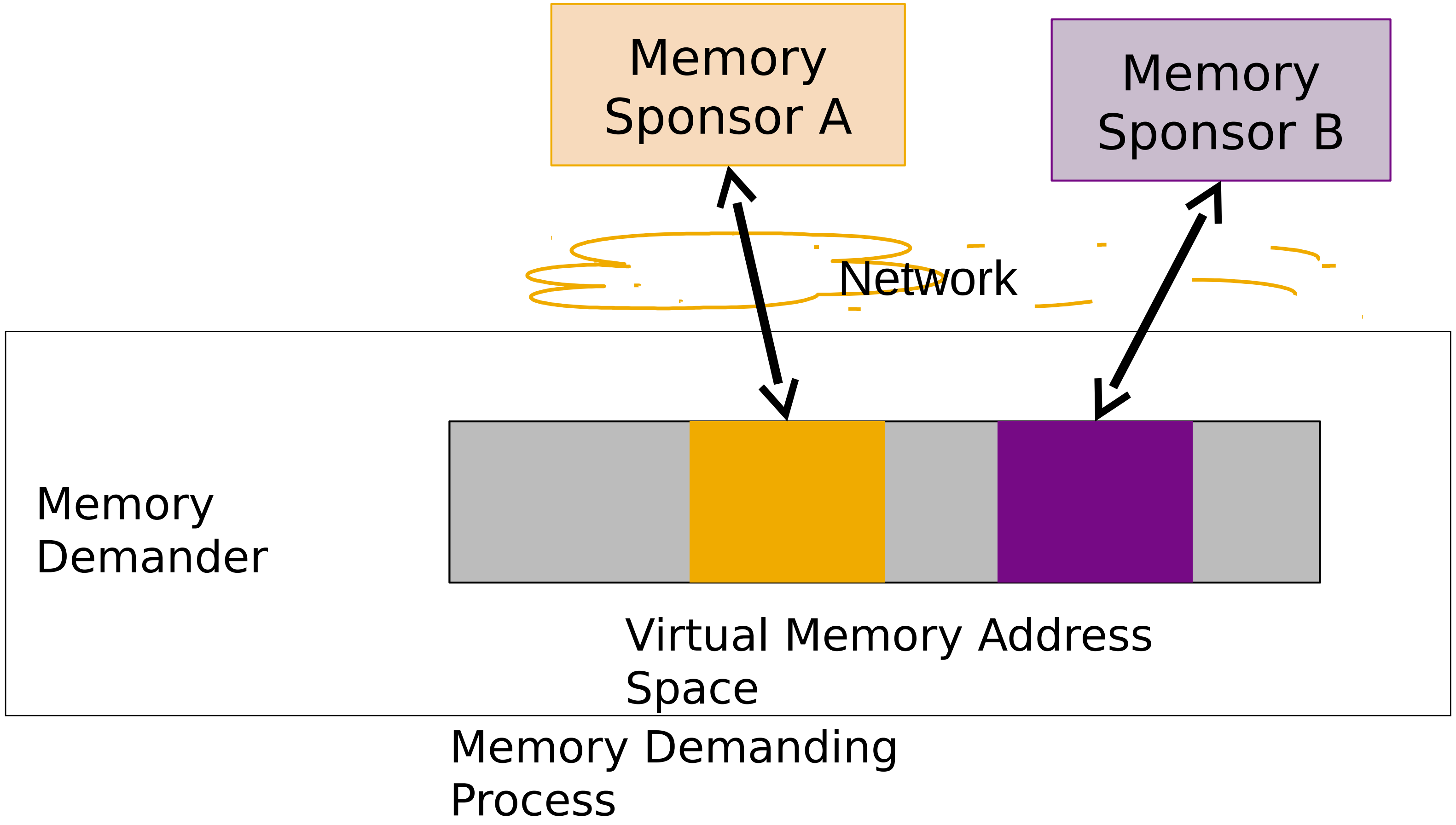
The Idea: Turning memory into a distributed memory service



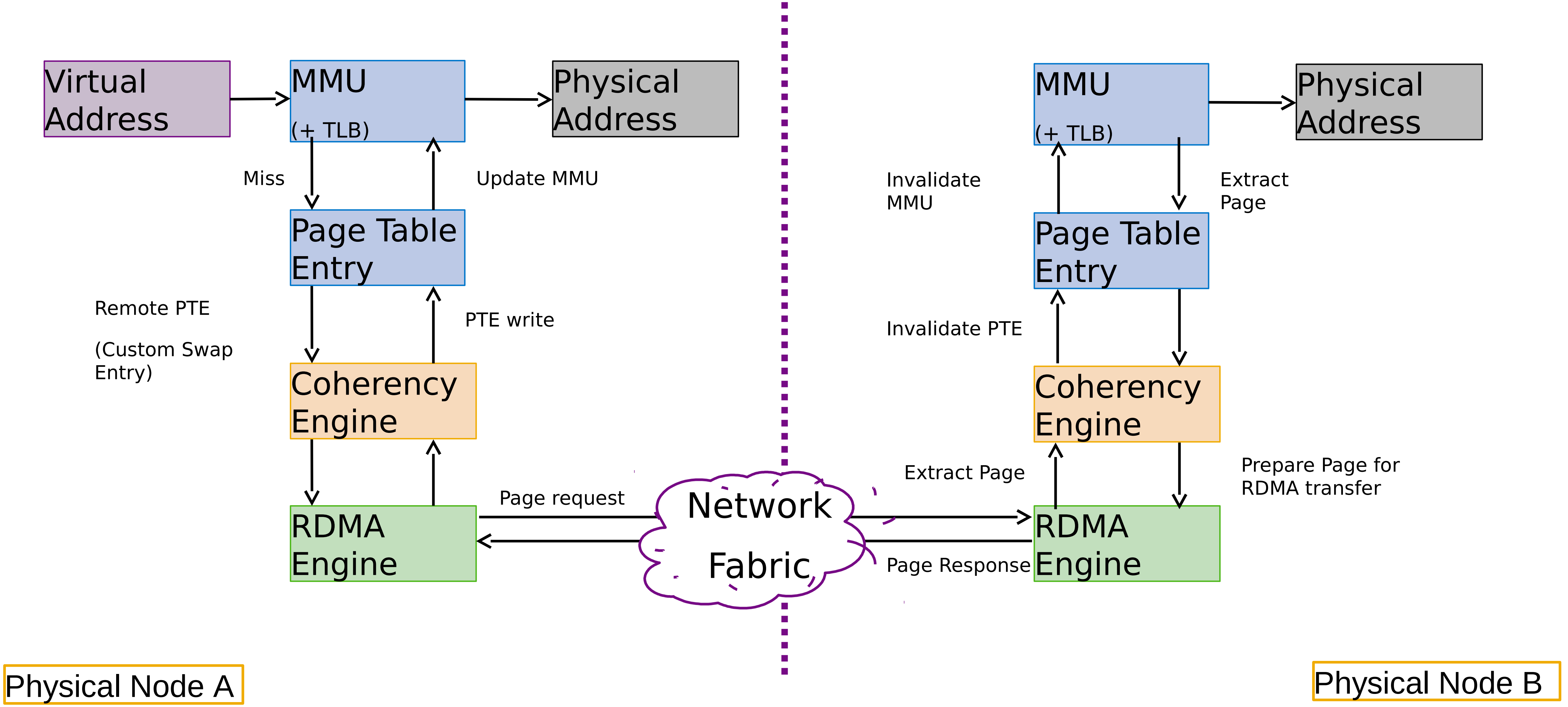
Breaks memory from the bounds of the physical box

Transparent deployment with performance at scale and Reliability

High Level Principle



How does it work
(Simplified Version)



Reducing Effects of Network Bound Page Faults

Full Linux MMU integration (<u>reducing the system-wide effects/cost of page fault</u>)

- Enabling to perform page fault transparency (only pausing the requesting thread)

Low latency RDMA Engine and page transfer protocol (<u>reducing latency/cost of page faults</u>)

- Implemented fully in kernel mode OFED VERBS
- Can use the fastest RDMA hardware available (IB, IWARP, RoCE)
- Tested with Software RDMA solution (Soft IWARP and SoftRoCE) (<u>NO SPECIAL HW REQUIRED</u>)

Demand pre-paging (pre-fetching) mechanism (<u>reducing the number of page faults</u>)

- Currently only a simple fetching of pages surrounding page on which fault occurred

Transparent Solution

Minimal Modification of the kernel (<u>simple and minimal intrusion</u>)

- 4 Hooks in the static kernel , virtually no overhead when enabled for normal operation

Paging and memory Cgroup support (<u>Transparent Tiered Memory</u>)

- Page are pushed back to their sponsor when paging occurs or if they are local they can be swapped out normally

KVM Specific support (<u>Virtualization Friendly</u>)

- Shadow Page table (EPT / NPT)
- KVM Asynchronous Page Fault

Transparent Solution (cont.)

Scalable Active – Active Mode (<u>Distributed Shared Memory</u>)

Shared Nothing with distributed index

Write invalidate with distributed index (end of this year)

Library LibHeca (<u>Ease of integration</u>)

Simple API bootstrapping and synching all participating nodes

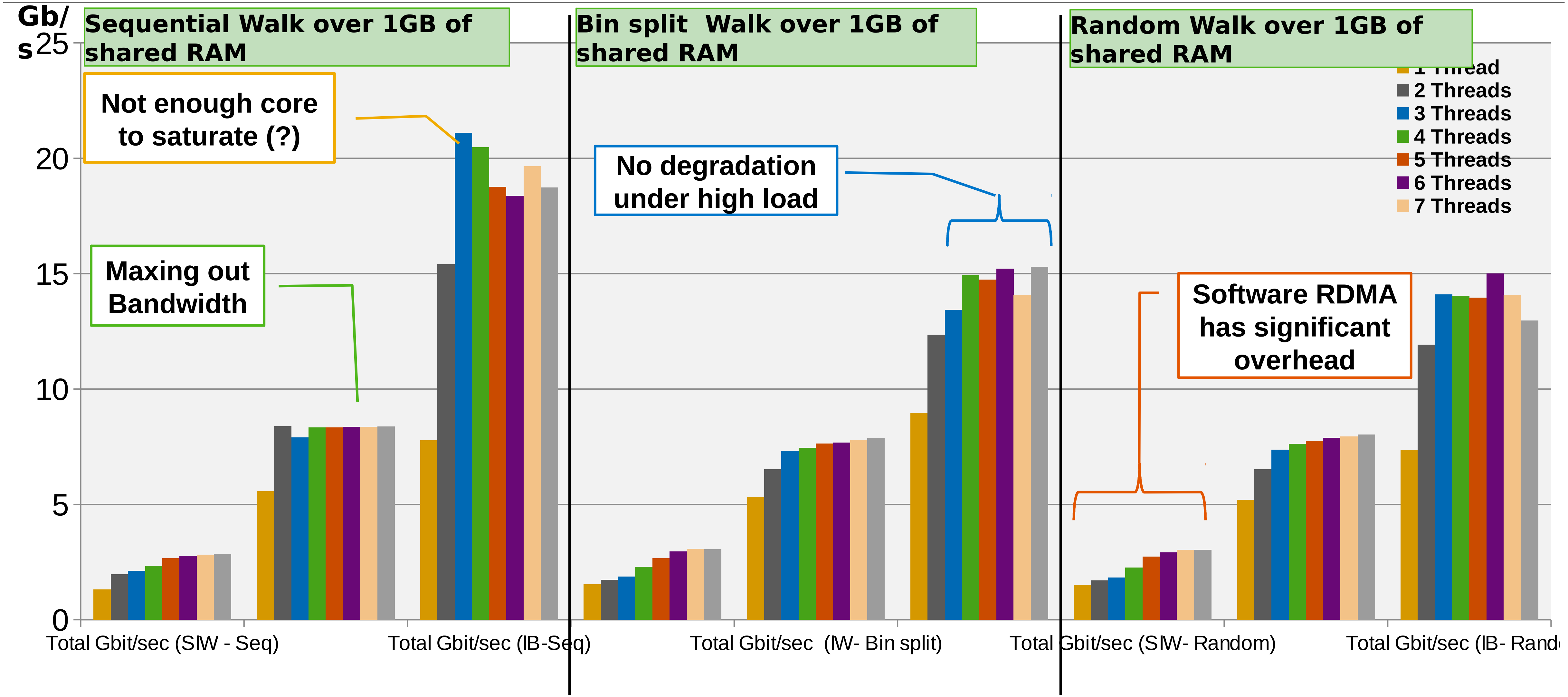
- **We also support:**
- KSM
- Huge Page
- Discontinuous Shared Memory Region
- Multiple DSM / VM groups on the same physical node

Raw Performance

How fast can we move memory around ?

Raw Bandwidth usage

HW: 4 core i5-2500 CPU @ 3.30GHz- SoftIwarp 10GbE – Iwarp Chelsio T422 10GbE - IB ConnectX2 QDR 40 Gbps

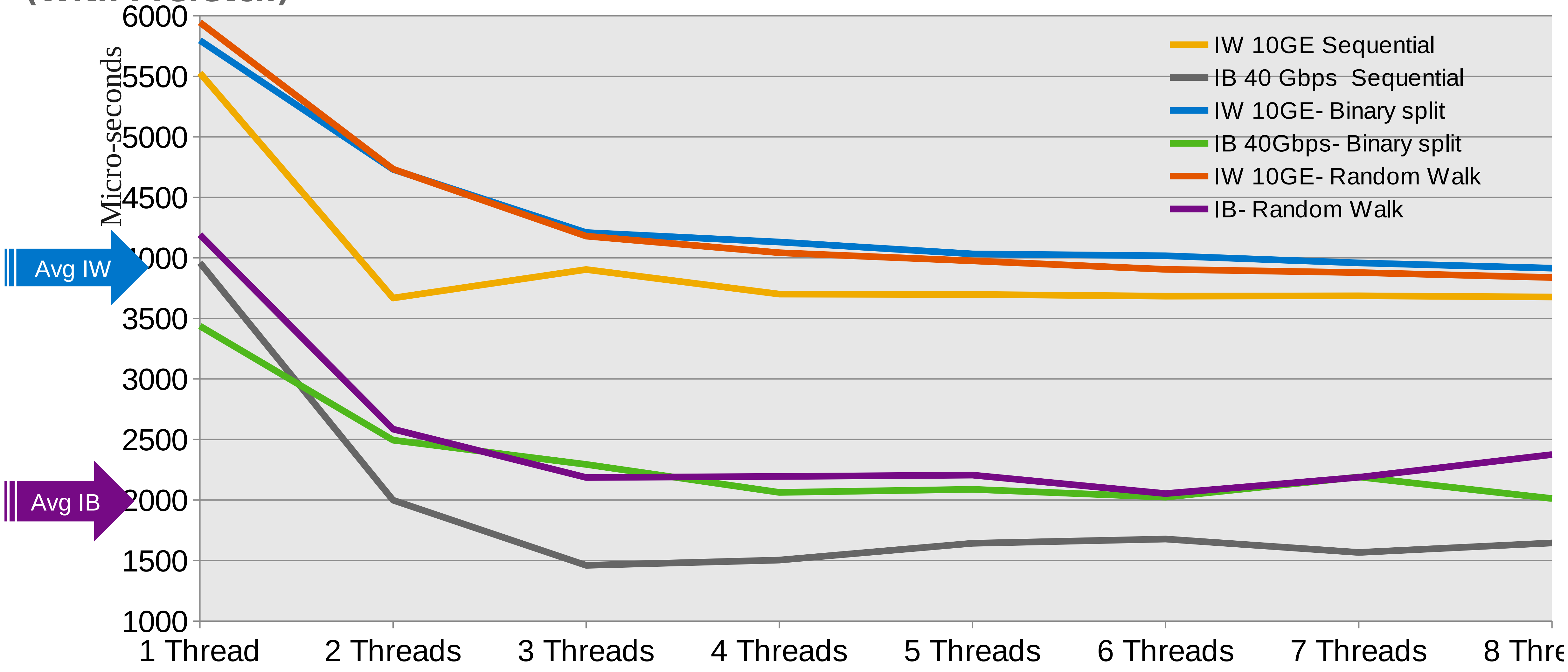


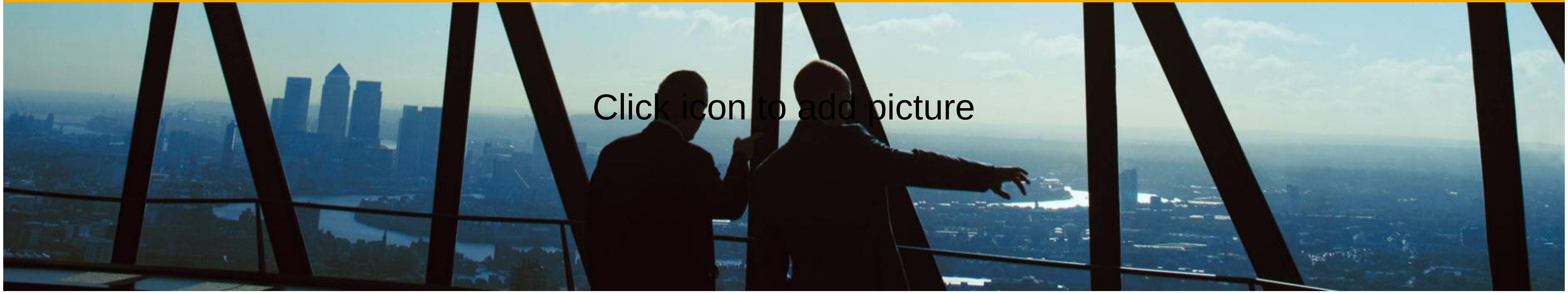
Hard Page Fault Resolution Performance

	Resolution time Average (µs)	Time spend over the wire one way Average (µs)	Resolution time Best (µs)
Softlwarp (10 GbE)	**355**	150 +	**74**
Iwarp (10GbE)	**48**	4-6	**28**
Infiniband (40 Gbps)	**29**	2-4	**16**

Average Compounded Page Fault Resolution Time

(With Prefetch)

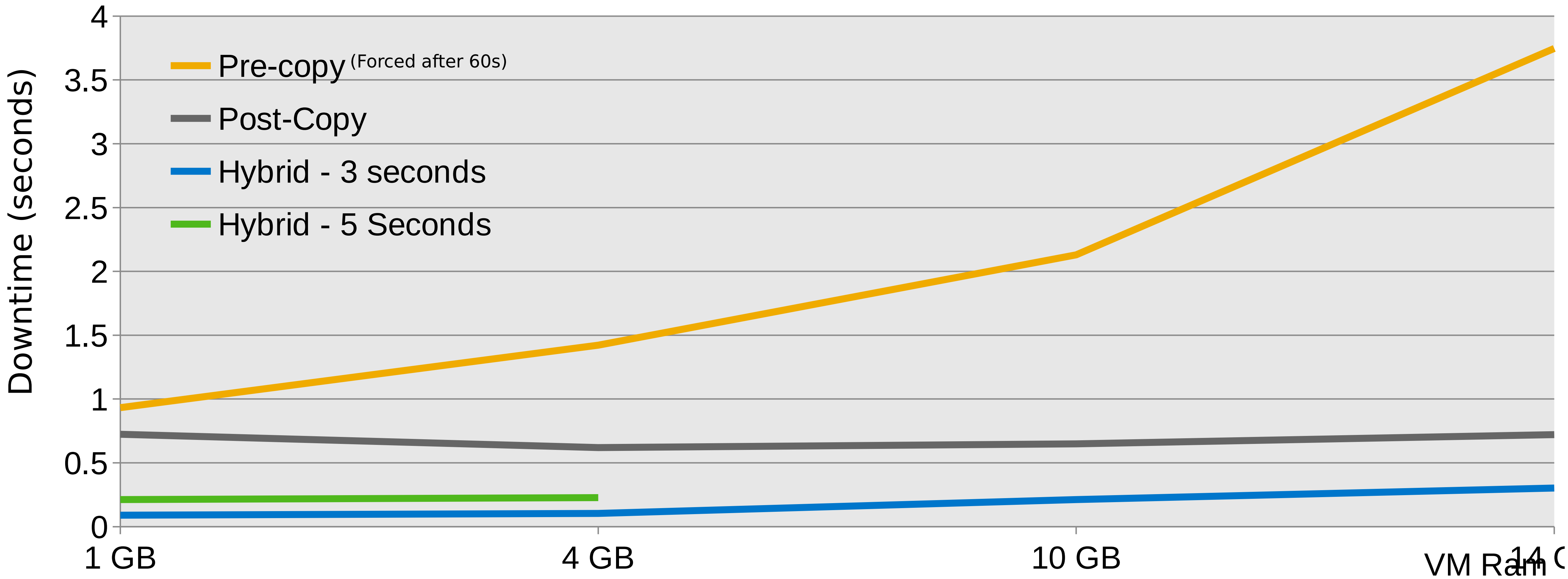




Post-Copy Live Migration

Technology first Use Case

Post Copy – Pre Copy – Hybrid Comparison

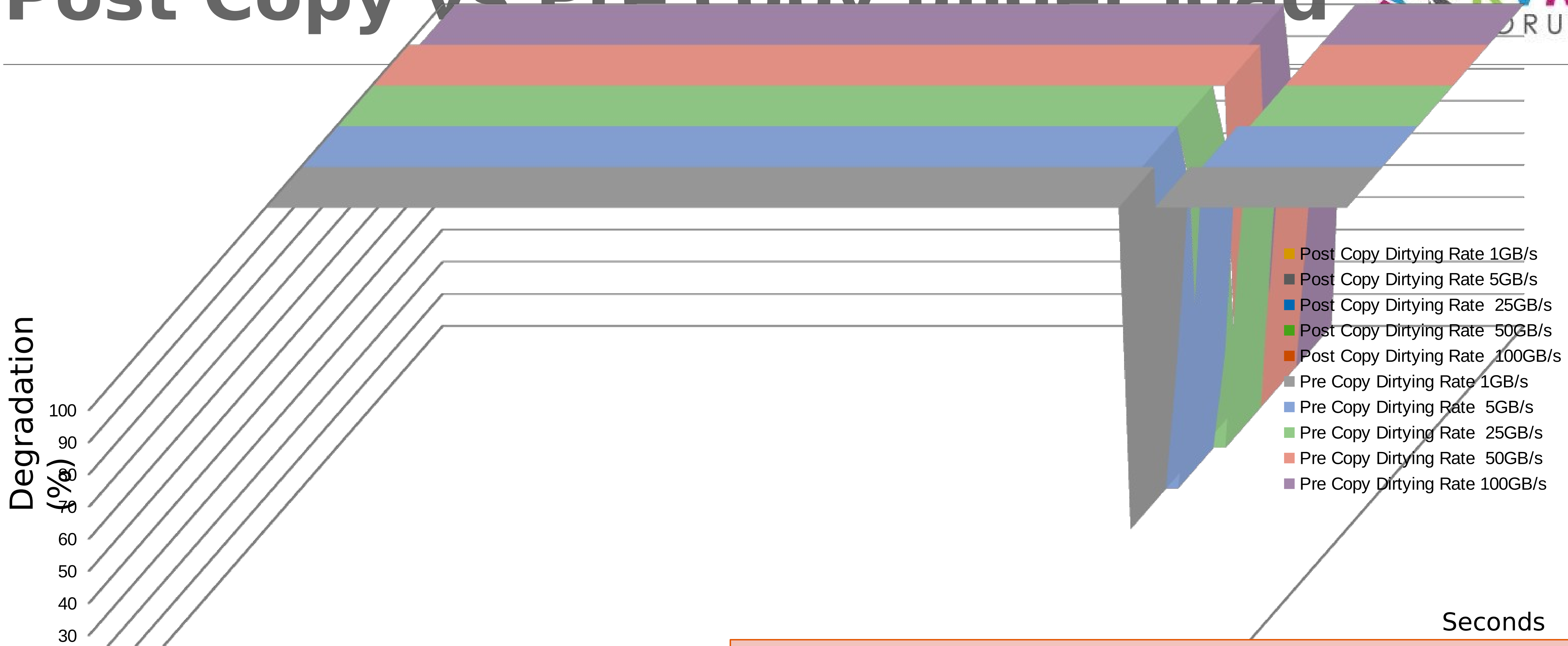


Host: Intel(R) Core(TM) i5-2500 CPU @ 3.30GHz, 4 cores, 16GB RAM

Network : **10 GB Eth** – Chelsio T422-CR IWARP

Workload App Mem Bench (~80% of the VM RAM) Dirtying Rate : **1GB/s** (256k Page dirtied per seconds)

Post Copy vs Pre copy under load



Virtual Machine :

- 1 GB RAM -1vCPU
- Workload: App Mem Bench

Hardware:

- Intel(R) Core(TM) i5-2500 CPU @ 3.30GHz, 4 cores, 16GB RAM
- Network : **10 GB Eth** Switch – NIC : Chelsio T422-CR (IWARP)

Post Copy Migration of HANA DB

	Baseline	Pre-Copy	Post-Copy
Downtime	N/A	7.47 s	**675 ms**
Benchmark Performance Degradatio	0%	Benchmark Failed	**5%**

Virtual Machine:

- **10 GB Ram , 4 vCPU**
- Application : **HANA** (In memory Database)

Hardware:

- Intel(R) Core(TM) i5-2500 CPU @ 3.30GHz, 4 cores, 16GB RAM
- Fabric: **10 GB Ethernet** Switch
- NIC: Chelsio IWARP T422-CR

Memory Aggregation

Second use case: Scaling out Memory

Scaling Out Virtual Machine Memory

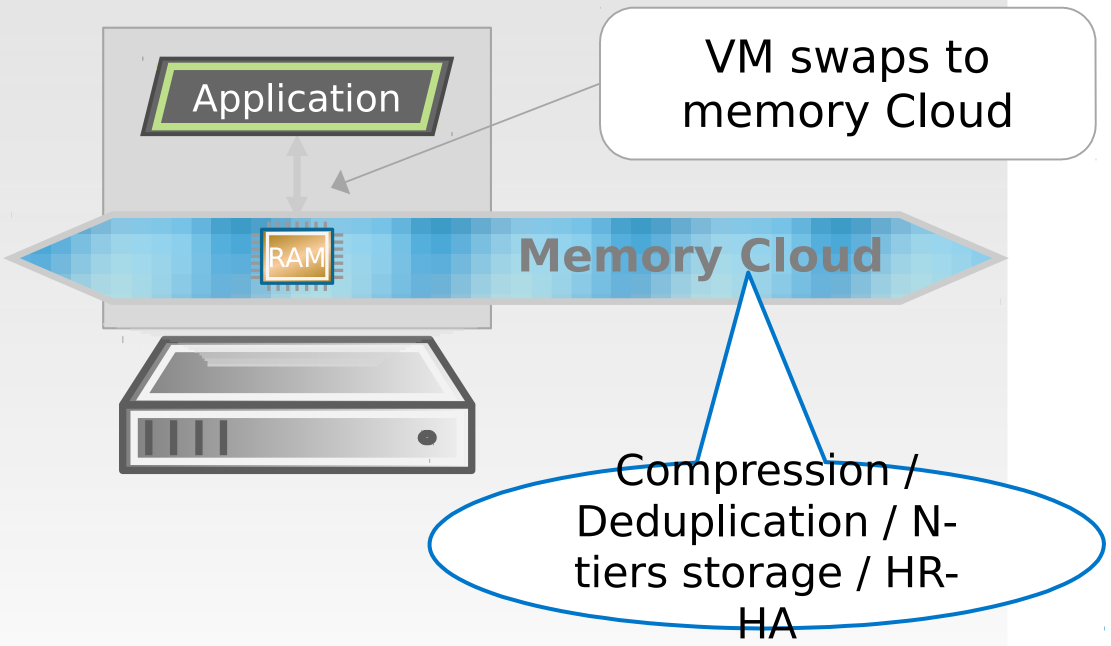
Business Problem

- Heavy swap usage slows execution time for data intensive applications

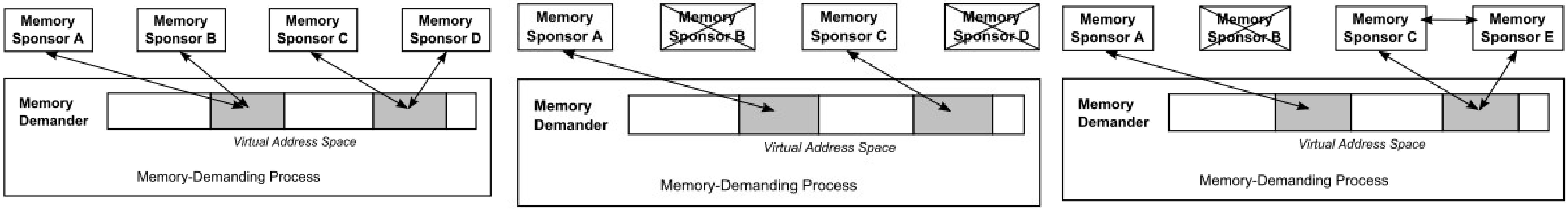
Hecatonchire/ RRAIM Solution

- Applications use memory mobility for high performance swap resource
  - Completely transparent
  - No integration required
  - Act on results sooner
  - High reliability built in
- Enables iteration or additional data to improve results

Solution



Redundant Array of Inexpensive RAM: RRAIM



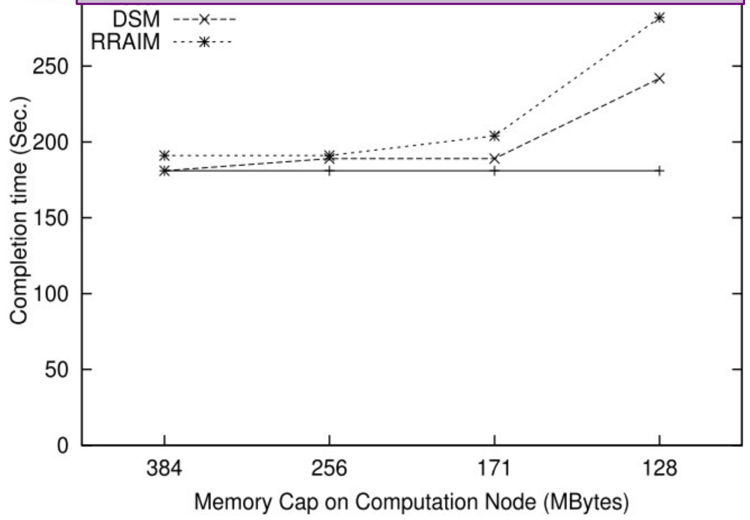
1. **Memory region backed by two remote nodes. Remote page faults and swap outs initiated simultaneously to all relevant nodes.**

2. **No immediate effect on computation node upon failure of node.**

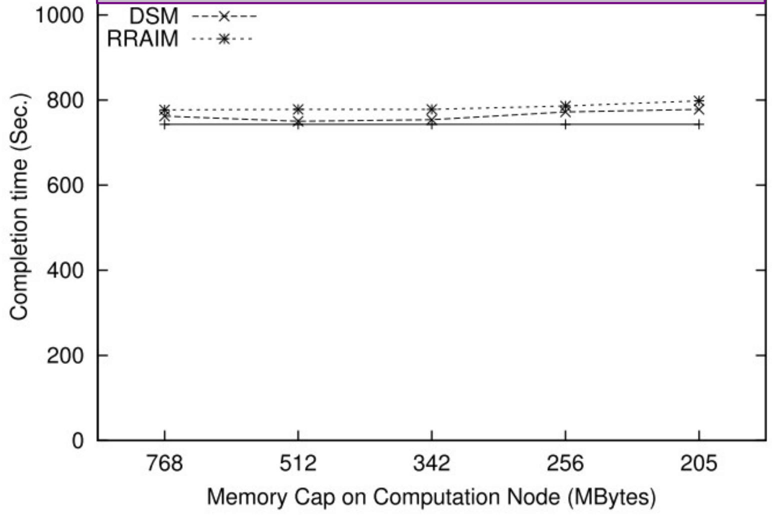
3. **When we a new remote enters the cluster, it**

Quicksort Benchmark with Memory Constraint

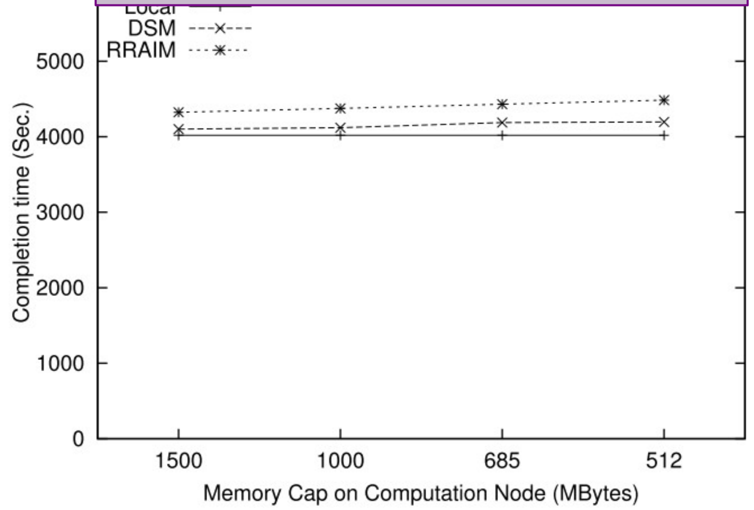
Quicksort Benchmark 512 MB Dataset



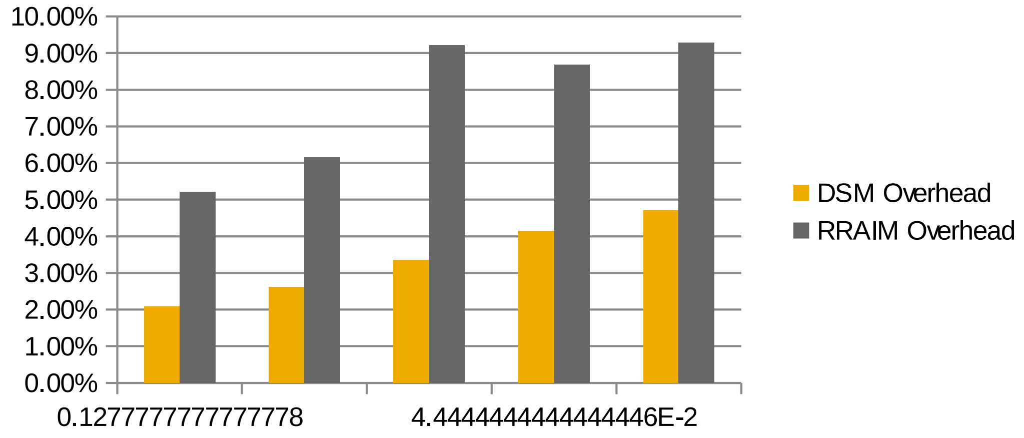
Quicksort Benchmark 1GB Dataset



Quicksort Benchmark 2GB Dataset

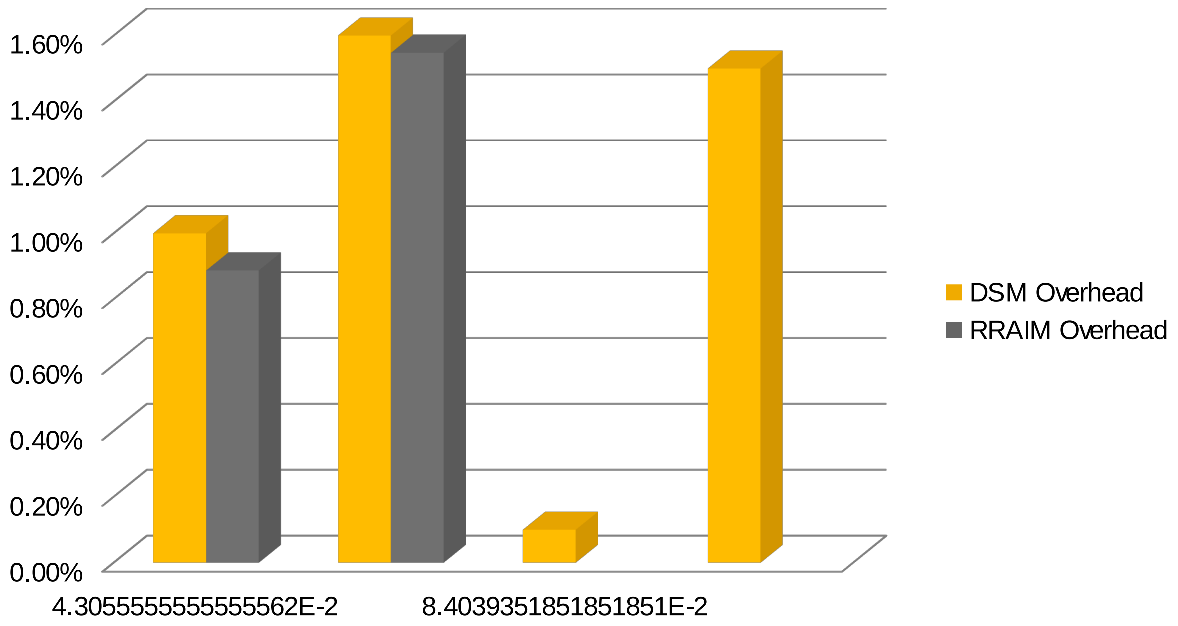


Memory Ratio (constraint using cgroup)	**DSM Overhead**	**RRAIM Overhead**
3:4	2.08%	5.21%
1:2	2.62%	6.15%
1:3	3.35%	9.21%
1:4	4.15%	8.68%
1:5	4.71%	9.28%



Scaling out HANA

Memory Ratio	DSM Overhead	RRAIM Overhead
1:2	1%	0.887%
1:3	1.6%	1.548%
2:1:1	0.1%	-
1:1:1	1.5%	-



Virtual Machine:

- **18 GB Ram , 4 vCPU**
- Application : **HANA** (In memory Database)
- Workload : SAP-H (**TPC-H** Variant)

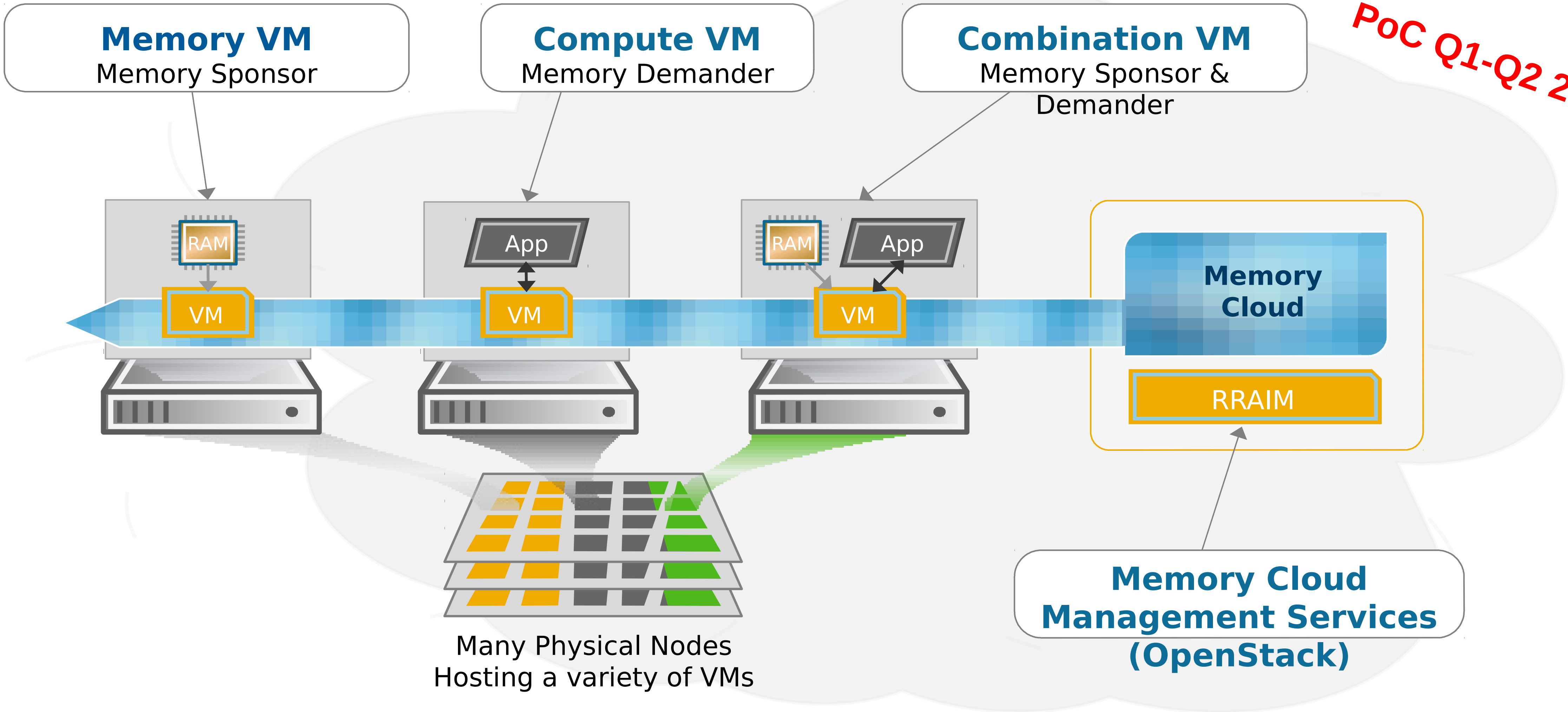
Hardware:

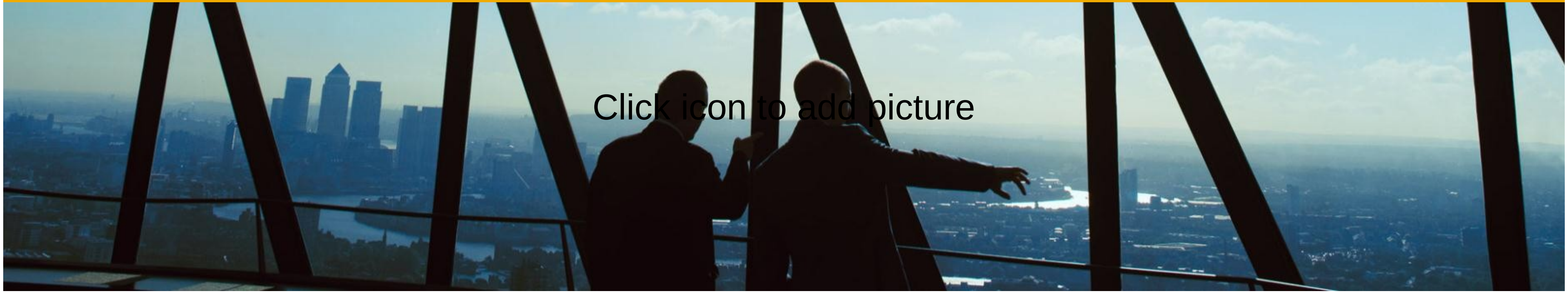
- Memory Host: Intel(R) Core(TM) i5-2500 CPU @ 3.30GHz, 4 cores, 16GB RAM
- Compute Host: Intel(R) Xeon(R) CPU X5650 @ 2.56GHz, 8 cores, 96GB RAM
- Fabric: Infiniband **QDR 40Gbps** Switch + Mellanox ConnectX2

Transitioning to a Memory Cloud

(Ongoing work)

PoC Q1-Q2 2013





Lego Cloud

Going beyond Memory

Virtual Distributed Shared Memory System

(Compute Cloud)

Compute aggregation

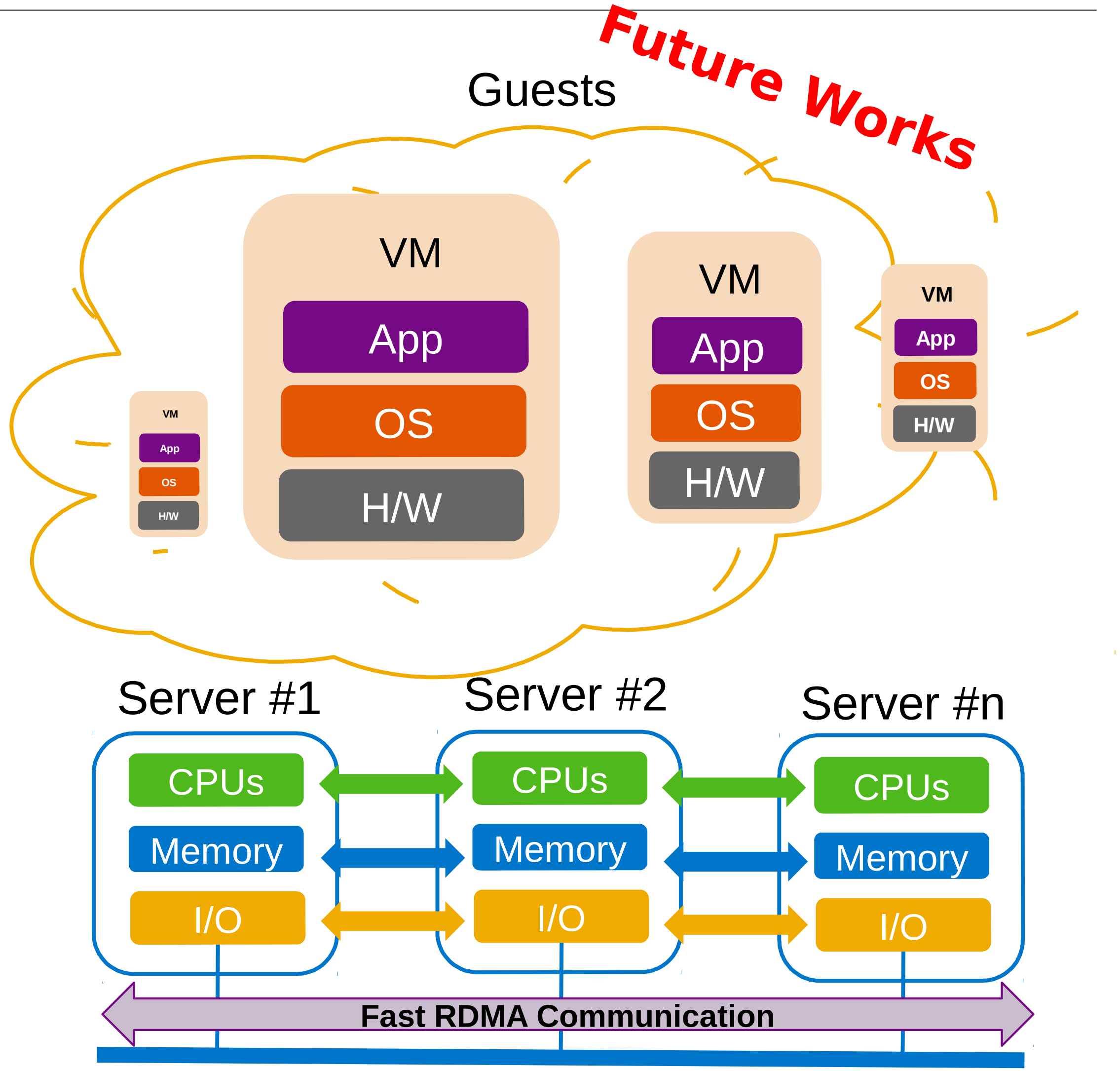
- Idea : Virtual Machine compute and memory span Multiple physical Nodes

Challenges

- Coherency Protocol
- Granularity (False sharing)

Hecatonchire Value Proposition

- Optimal price / performance by using commodity hardware
- Operational flexibility: node downtime without downing the cluster
- Seamless deployment within existing cloud



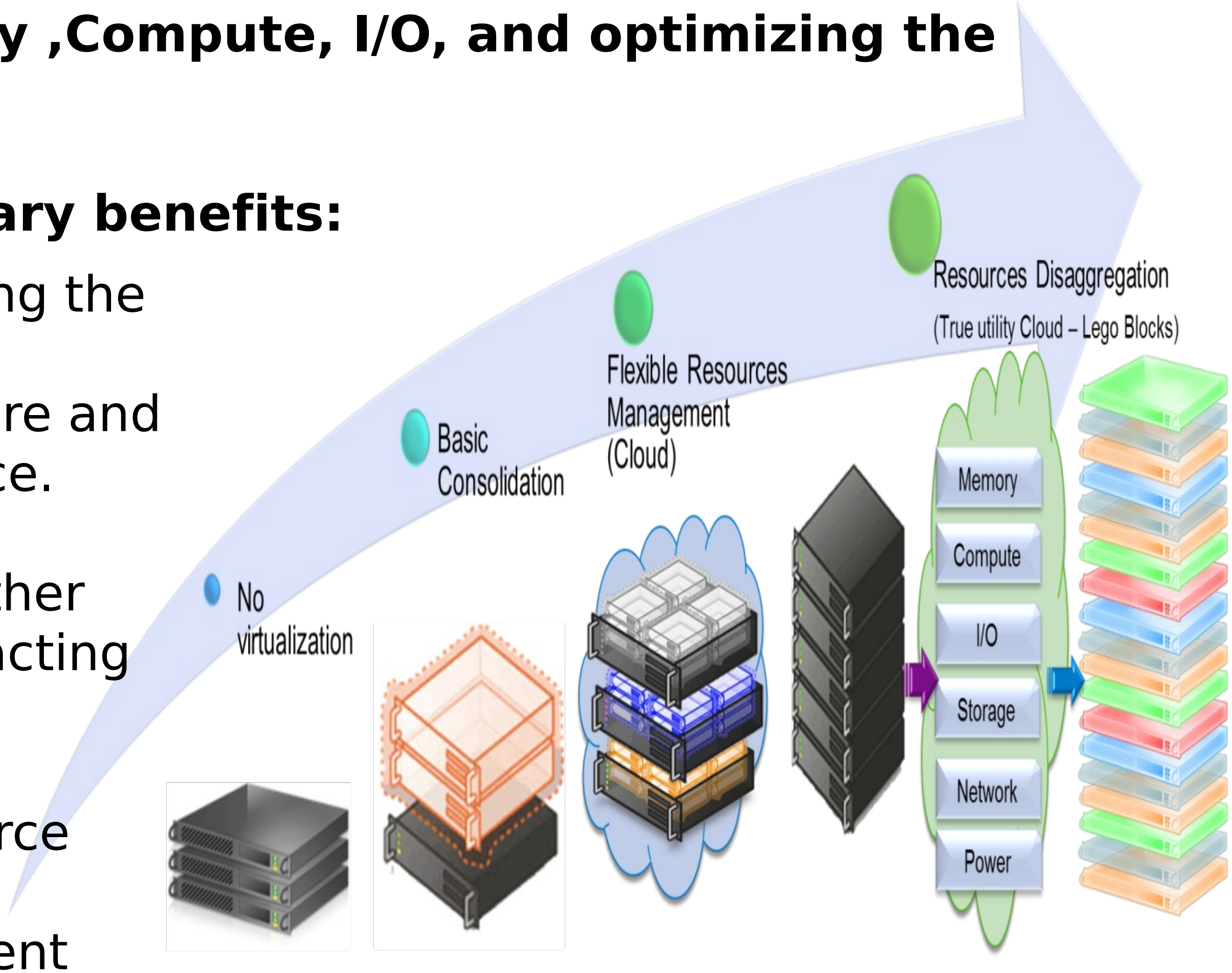
Disaggregation of datacentre (and cloud) resources

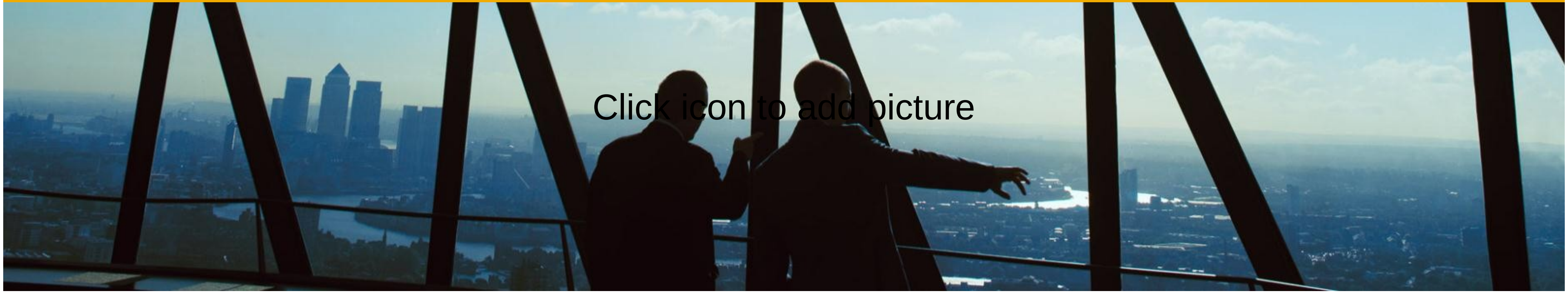
(Our Aim)

Breaking out the functions of Memory ,Compute, I/O, and optimizing the delivery of each.

Disaggregation, provides three primary benefits:

- **Better Performance:**
  - Each function is isolated => limiting the scope of what each box must do
  - We can leverage dedicated hardware and software => increases performance.
- **Superior Scalability:**
  - Functions are isolated from each other => alter one function without impacting the others.
- **Improved Economics:**
  - cost-effective deployment of resource => improved provisioning and consolidation of disparate equipment





Summary

Hecatonchire Project

– **Features:**

- Distributed Shared Memory
- Memory extension via Memory Servers
- HA features
- Future :Distributed Workload executions
  – **Use standard Cloud interface**
  – **Optimise Cloud infrastructure**
  – **Support COTS HW**

Key takeaways

- **Hecatonchire project aim at disaggregating datacentre resources**
- **Hecatonchire Project currently deliver memory cloud capabilities**
- **Enhancements to be released as open source under GPLv2 and LGPL licenses by the end of November 2012**
- **Hosted on GitHub, check: www.hecatonchire.com**
- **Developed by SAP Research Technology Infrastructure (TI) Programme**

Thank you

Benoit Hudzia; Sr. Researcher;

SAP Research Belfast

benoit.hudzia@sap.com

Backup Slide

Instant Flash Cloning On-Demand

Planned Prototype Q4 2012

Business Problem

- Burst load / service usage that cannot be satisfied in time

Existing solutions

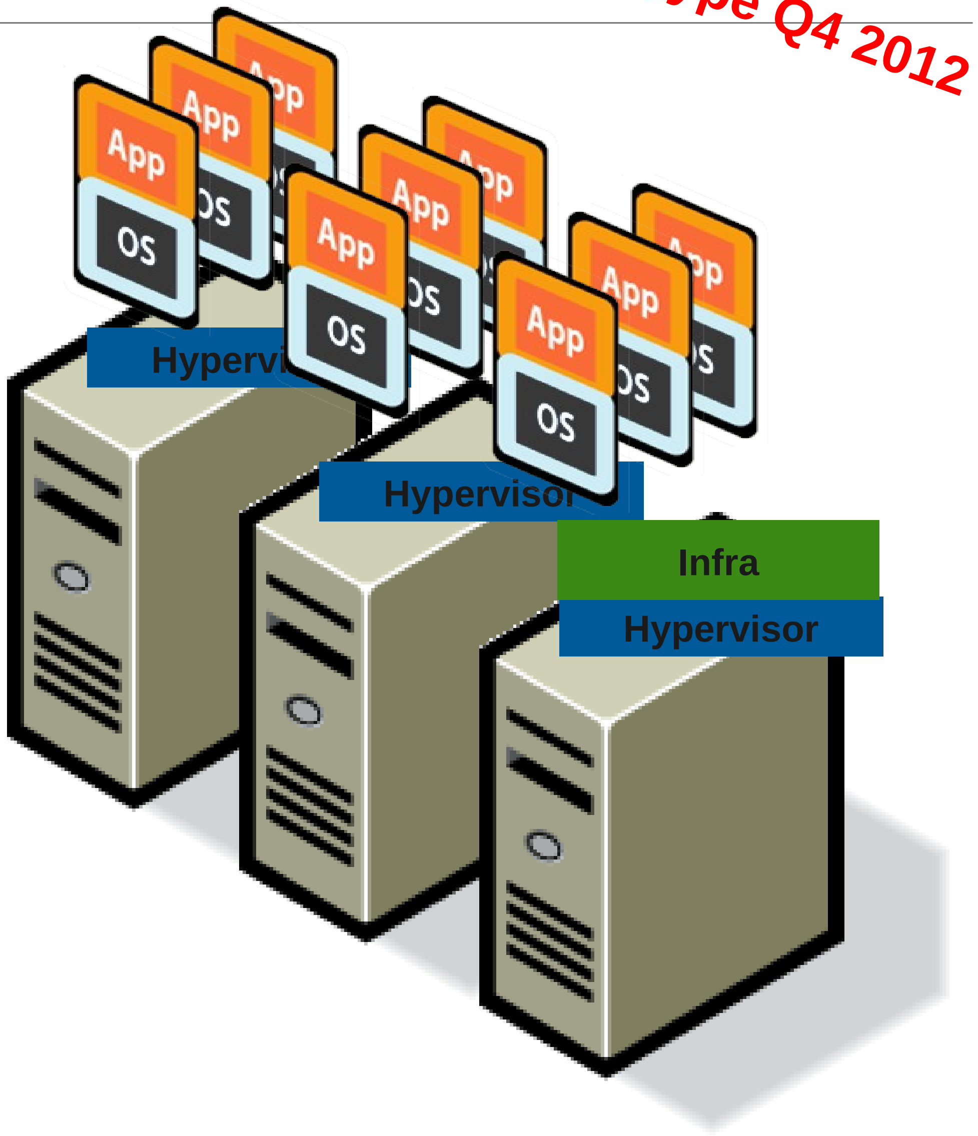
- Vendors: Amazon / VMWare/ rightscale
- Startup VM from disk image
- Requires full VM OS startup sequence

Hecatonchire Solution

- Go live after VM-state (MBs) and hot memory (<5%) cloning
- Use post-copy live-migration schema in background
- Complete background transfer and disconnect from source

Hecatonchire Value Proposition

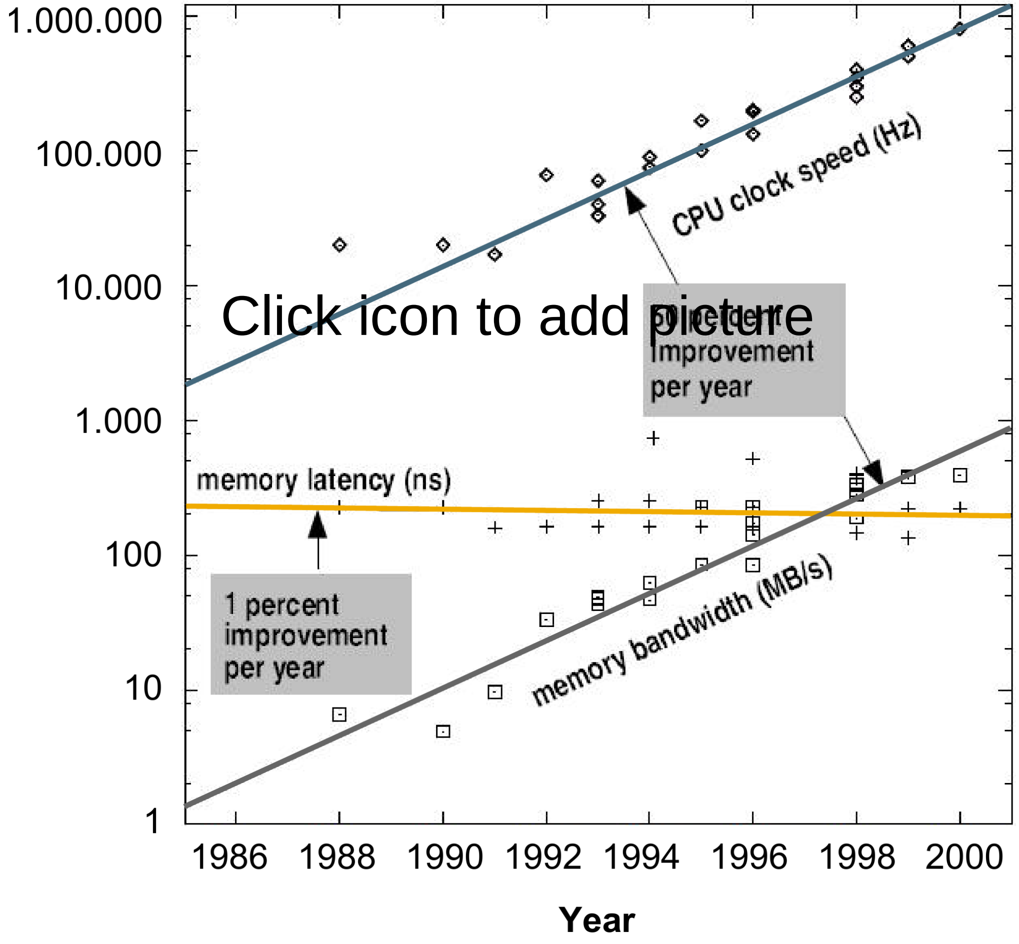
- Just in time (sub-second) provisioning



DRAM Latency Has Remained Constant

CPU clock speed and memory bandwidth increased steadily (at least until 2000)

But memory latency remained constant – so local memory has gotten slower from the CPU perspective



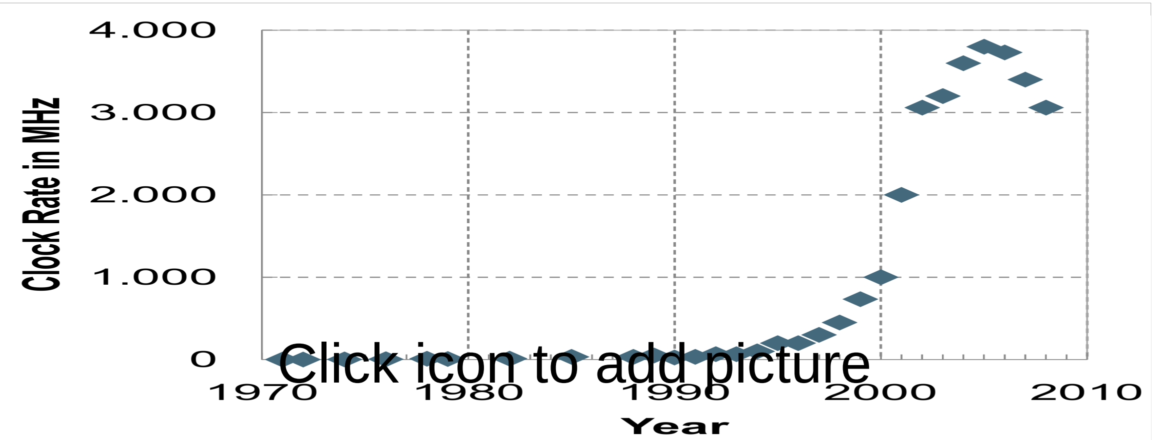
Source: J. Karstens: *In-Memory Technology at SAP. DKOM 2010*

CPUs Stopped Getting Faster

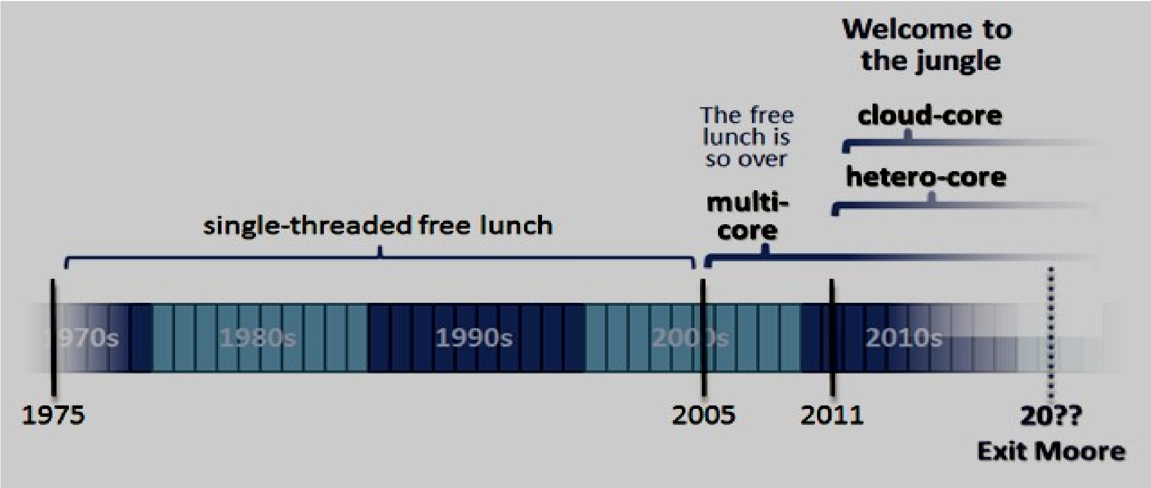
Moore's law prevailed until 2005 when core's speed hit a practical limit of about 3.4 GHz

Since 2005 you do get more cores but the "single threaded free lunch" is over

Effectively arbitrary sequential algorithms have not gotten faster since



Source: http://www.intel.com/pressroom/kits/quickrefyr.htm



Source: "The Free Lunch Is Over.." by Herb Sutter

While … Interconnect Link Speed has Kept Growing

Ethernet (1979 -)	10 Mbit/sec
Fast Ethernet (1993 -)	100 Mbit/sec
Gigabit Ethernet (1995 -)	1000 Mbit /sec
ATM (1995 -)	155/622/1024 Mbit/sec
Myrinet (1993 -)	1 Gbit/sec
Fibre Channel (1994 -)	1 Gbit/sec
InfiniBand (2001 -)	2 Gbit/sec (1X SDR)
10-Gigabit Ethernet (2001 -)	10 Gbit/sec
InfiniBand (2003)	8 Gbit/sec (4X SDR)
InfiniBand (2005 -)	16 Gbit/sec (4X DDR)
	24 Gbit/sec (12X SDR)
InfiniBand (2007 -)	32 Gbit/sec (4X QDR)
40-Gigabit Ethernet (2010 -)	40 Gbit/sec
InfiniBand (2011 -)	56 Gbit/sec (4X FDR)
InfiniBand (2012 -)	100 Gbit/sec (4X EDR)

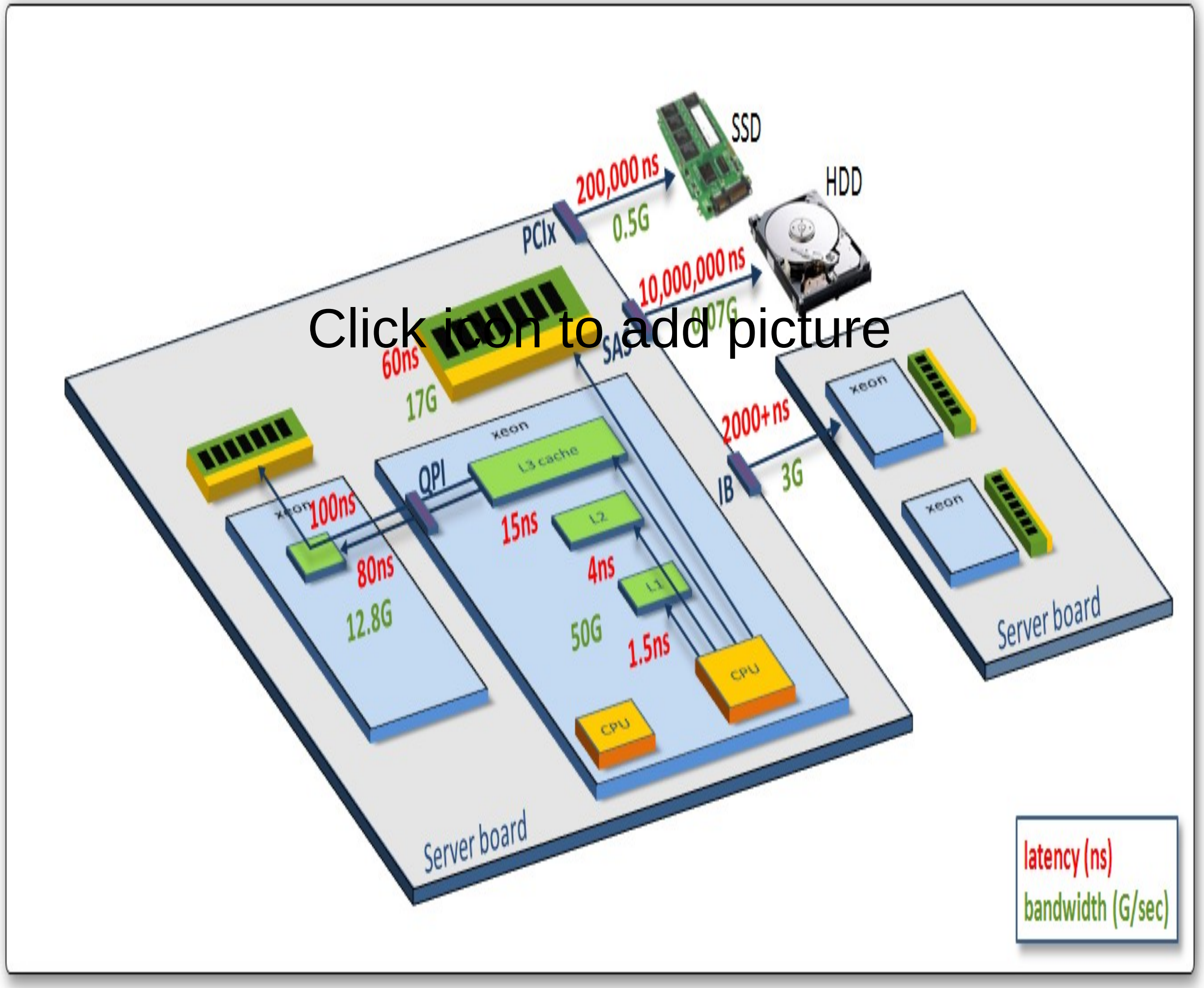
Panda et al. Supercomputing 2009

<u>Result:</u> Remote Nodes Have Gotten Closer

Accessing DRAM on a remote host via IB interconnects is only 20x slower than local DRAM

And remote DRAM has far better performance than paging in from an SSD or HDD device

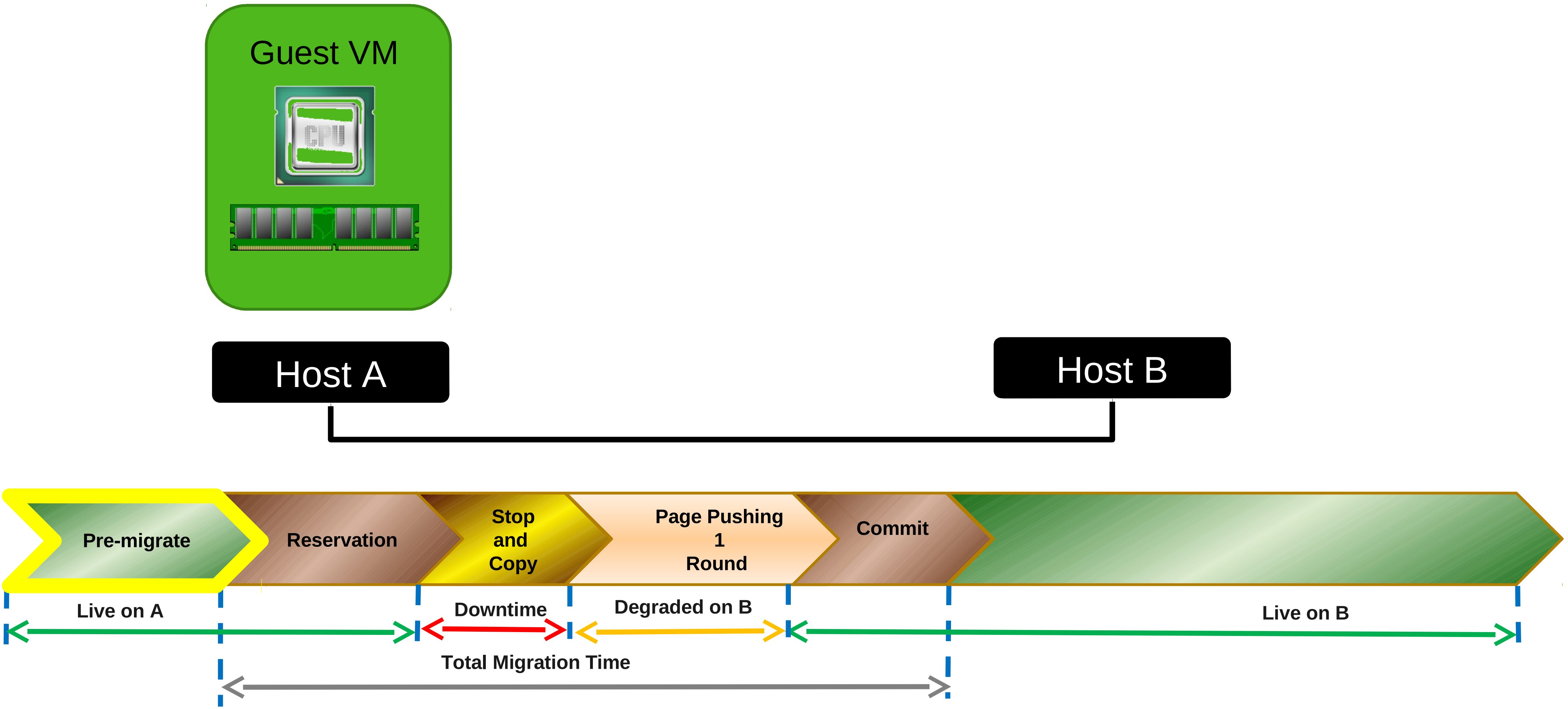
Fast interconnects have become a commodity - moving out of the High Performance Computing (HPC) niche



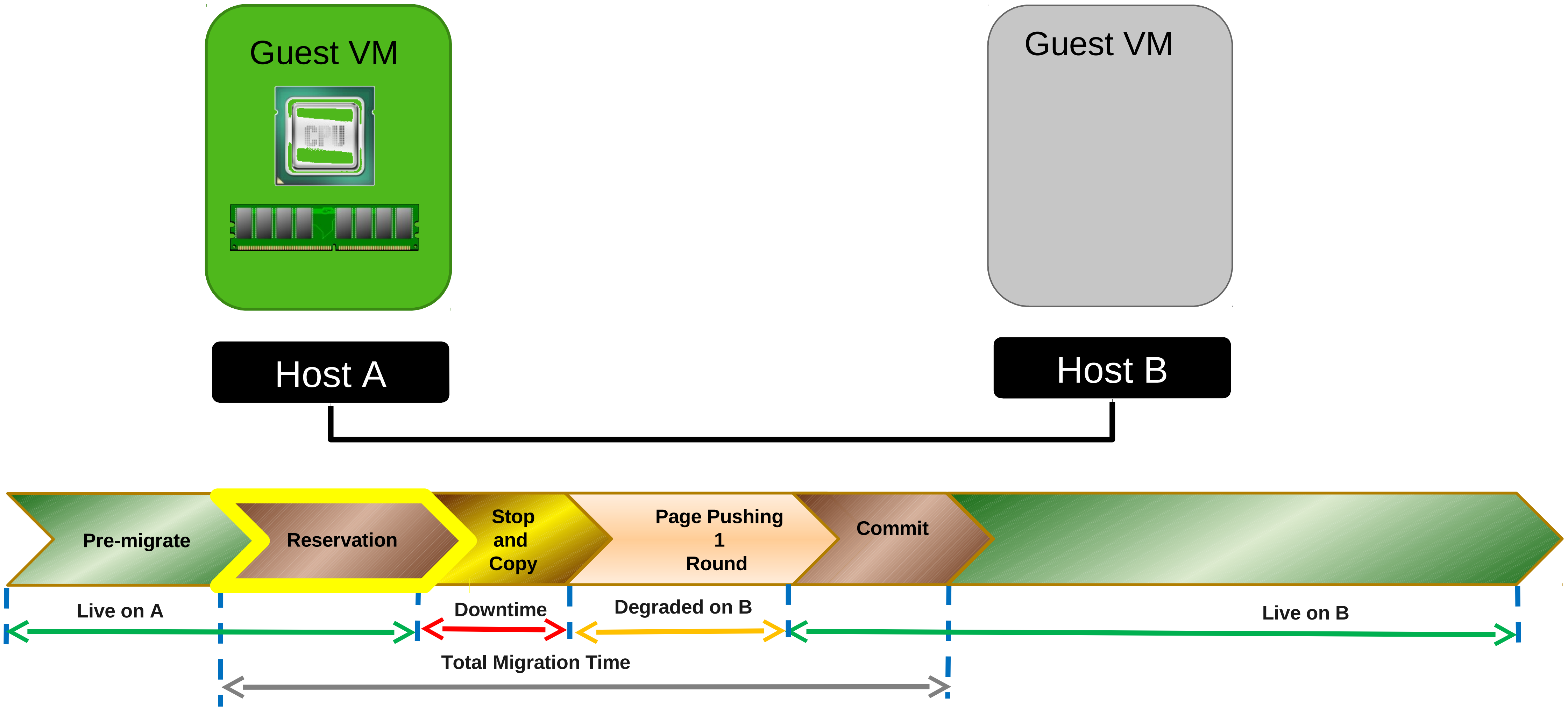
Click icon to add picture

HANA Performance Analysis, Chaim Bendelac, 2011

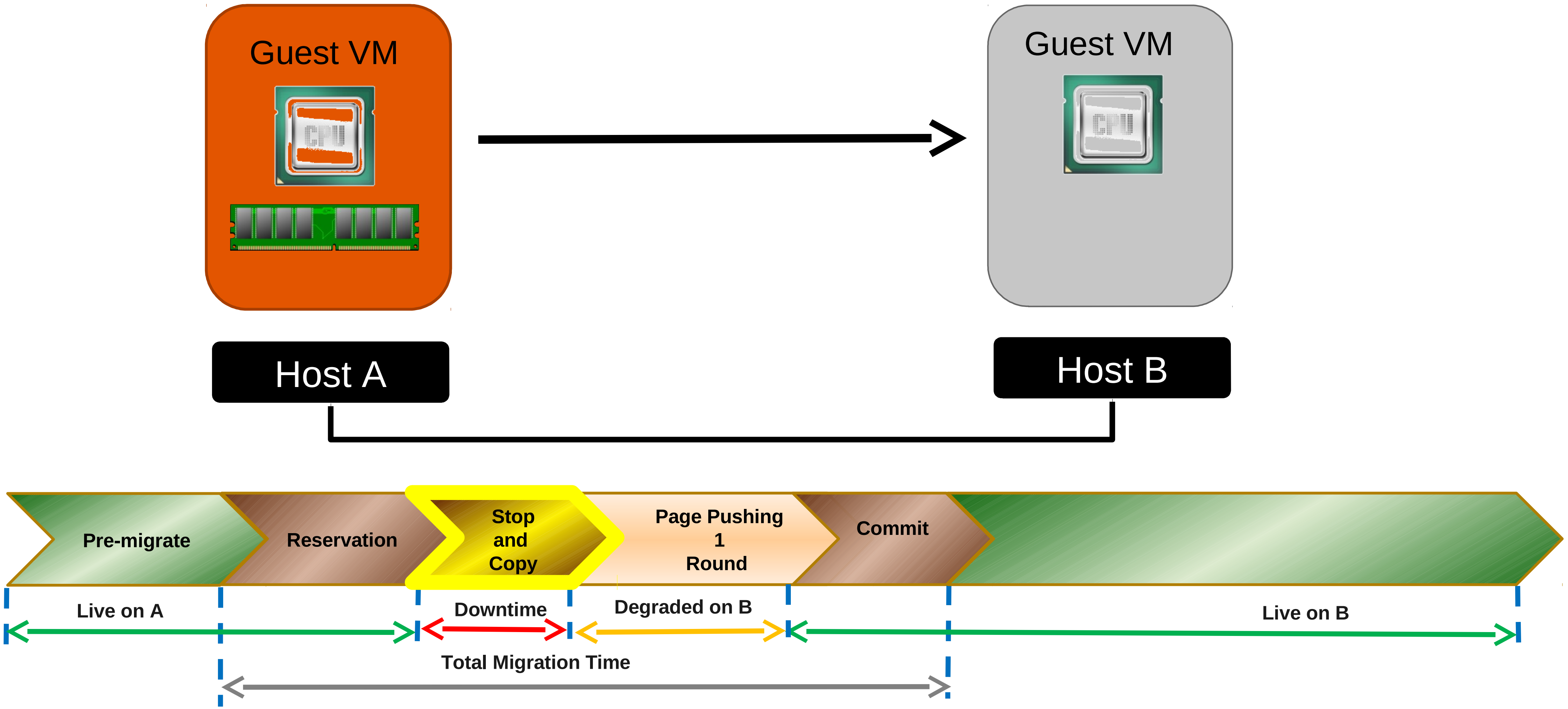
Post-Copy Live Migration (pre-migration)



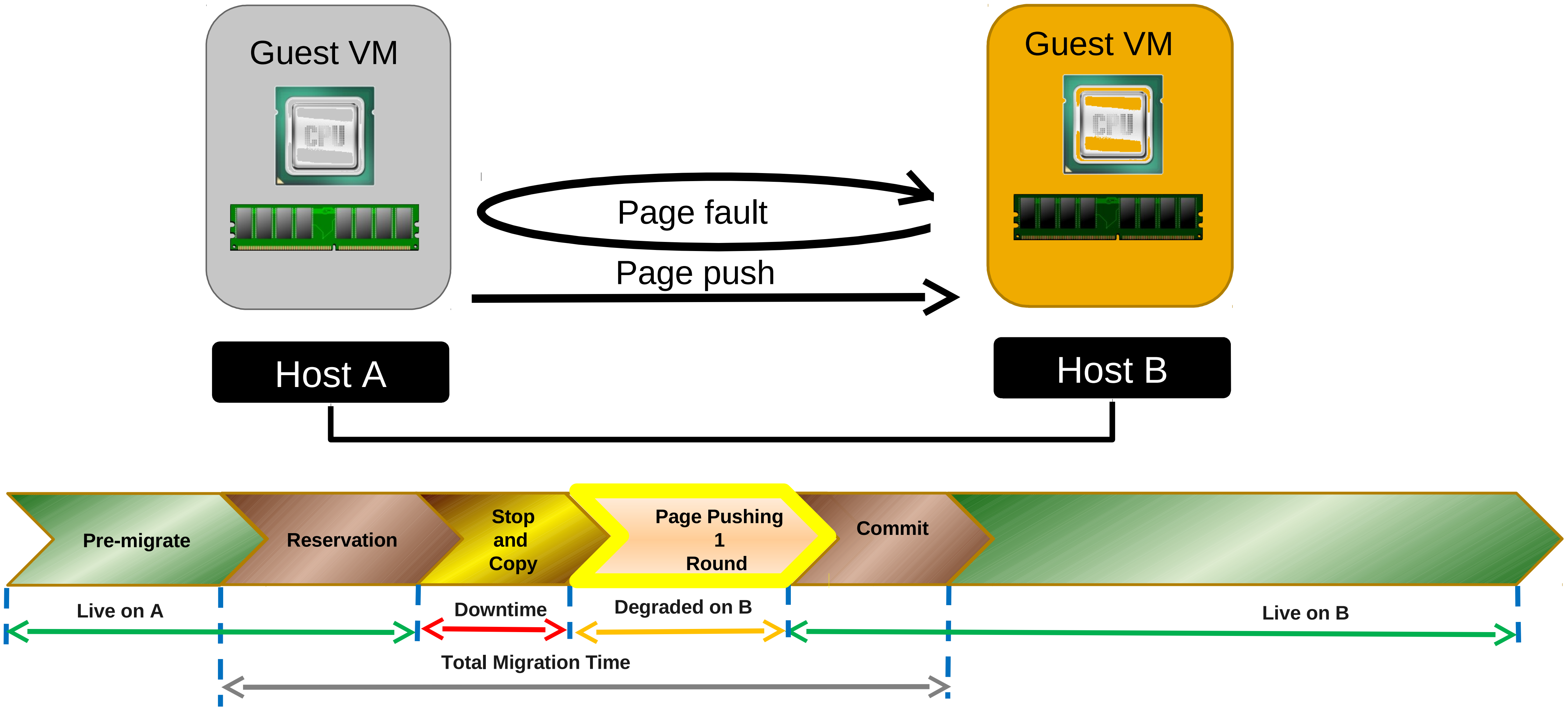
Post-Copy Live Migration (reservation)



Post-Copy Live Migration (stop and copy)



Post-Copy Live Migration (post-copy)



Post-Copy Live Migration (commit)