IIT Roorkee

eict IIT Roorkee

IntelliPaat

Advanced Certification in Cloud Computing & DevOps

VM Migration

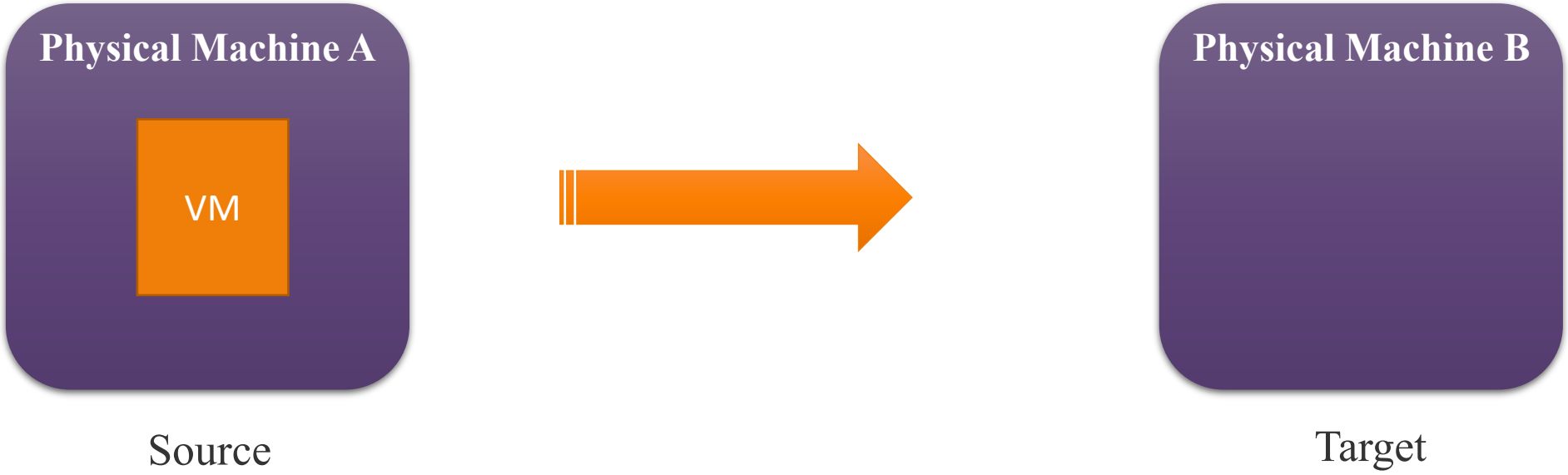
CONTENT SOURCE

- “Live Migration of Virtual Machines”, Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, Andrew Warfield
- “Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning”, Michael R. Hines and Kartik Gopalan

Source Contents Credit to:

- **Prof. Rajkumar Buyya** (*University of Melbourne, Australia*.)
- **Prof. Mythili Vutukuru** (IIT Bombay, *India*.)

VM Migration



Review: Virtual Machines

Virtualization provides **interface identical to underlying bare hardware for each VM**

- i.e., all devices, interrupts, memory, page tables etc.

Virtualization Software: VMWare, Hyper-V, Oracle virtual box, Xen etc.

- It allows clean **separation between hardware and software**.
- **Process level migration problems can be avoided** by migrating a virtual machine.

- **Virtualization provides facility to migrate** virtual machine from one host (source) to another physical host (destination).
- Virtual Machine Migration is a **useful tool for administrator** of data center and clusters.

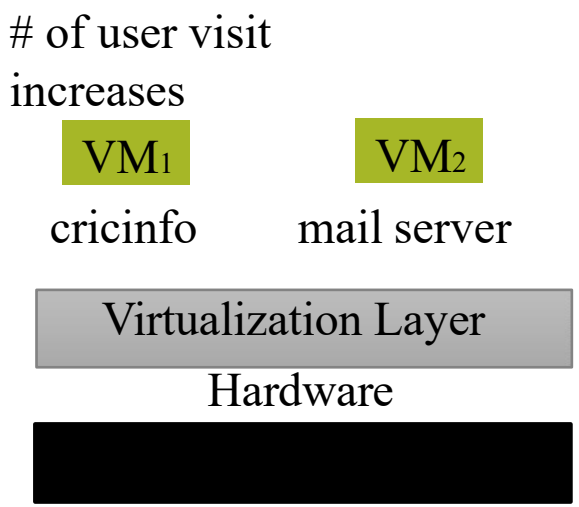
Review: Virtual Machines in Cloud

- **Benefits** of Virtual Machines
  - Virtualization helps in making **efficient use of hardware resources**
  - Facilitates a greater **degree of abstraction**
  - **Easily move** from one piece of hardware to another
  - **Replicate** them at will
  - Create more **scalable and flexible infrastructure**
  - **High Availability and Fault tolerance (How?)**

- Cloud computing has taken that **degree of efficiency and agility realized** from virtualization
  - Pooled resources
  - Geographic diversity
  - Universal connectivity

Motivation

- Consider a **data center** consisting of **"one" physical machines (PM) hosting "two" VMs implementing one customer application each**
- **Resources**(CPU, Network, Memory, I/O) are **allocated to each VM** to **handle the workload** and operate at certain performance level **(SLA)**
- Each **VM sees workload fluctuation (WHY?)** from time to time => resource requirement changes



PM Capacity:
- Network Bandwidth = 10 Gbps
- Memory = 16 GB
- CPU = 8 cores

Resource Allocation:

VM1	**VM2**
N = 5Gbps	N = 5Gbps
M = 8GB	M = 8GB
C = 4 cores	C = 4 cores

Motivation

- An **increase in workload** can be handled by **allocating more resources** to it, **if idle** resources are available

❑ Main Issues:

What if PM does not have (enough or no) idle resources to satisfy VM's requirement?

- **Performance** of the application degrades
- **SLA violation** occurs

❑ **Key Ideas**

- Replication VMs
- Migrating VMs

Virtual Machine Migration

- Why do we need migration?
- When do we need to migrate?
- How migration is done?
- Issues in long distance migration (across data centers)

• Why do we need migration?

- To Maintain/Upgrade
- To Optimize the Performance
- To Protect SLA
- To Improve QoE

When we need to migrate? [NSDI' 07]

- **Hotspots can cause SLA violations**
  - Burden on some Virtual or Physical Machines are called hotspots

- **Hotspot Detection**
  - ***Black-box Monitoring***
    - CPU
    - Network
    - Memory
  - ***Gray-box Monitoring***
    - Gather OS level statistics and application logs

- A hotspot is **flagged only if thresholds or SLAs are exceeded** for a sustained time

When we need to migrate? [FGCS' 12]

- **SLA violation detection**
  - **Mapping** low-level resource metrics to high-level SLAs
    - ***CPU speed maps to Response Time***
    - ***Occupied memory size maps to number of concurrent clients***
  - **Predictive Strategy** for detection of possible SLA violations
  - **Detection interval**
    - **Short measurement intervals** may **degrade** performance
    - **Long measurement intervals** may cause **ignorance of heavy SLA** violations.

Migration Techniques

- The **different virtual machine migration** techniques are as follows:
  - Fault Tolerant Migration Techniques
  - Load Balancing Migration Techniques
  - Energy Efficient Migration Techniques

Fault Tolerant Migration Techniques

- Fault tolerance **allows the VMs to continue its job even any part of system fails.**
- This technique **migrates the VM from one physical server to another** physical server based upon the **prediction of the failure occurred**.
- Fault tolerant migration technique is **to improve the availability of physical server** and **avoids performance degradation of applications**.

Load Balancing Migration Techniques

- The Load balancing migration technique **aims to distribute load across the physical servers** to **improve the scalability of physical servers** in cloud environment.
- **The Load balancing improves the**
  - resource consumption
  - implementation of failover
  - enhancing scalability

Energy Efficient Migration Techniques

- **The power consumption of Data center is mainly based on the utilization** of the servers and **their cooling systems**.
- The servers **typically need up to 70 percentage of their maximum power consumption even at their low utilization level.**
- Therefore, there is a need for migration techniques that **conserves the energy of servers by optimum resource utilization.**

VM Migration Methods

- Virtual Machine Migration methods are divided into **two types:**
  - **Cold (non-live) migration**
  - **Hot (live) migration**

- The **status of the VM loses** and **user can notice the service interruption** in cold migration. First, **VM is suspended**, then its **state is transferred**, and lastly, **VM is resumed at destination** host.

- In **live migration process**, **the state of a running VM is transferred/migrated from Host A to Host B.**
- **VM keeps running** while migrating and **does not lose its status**. **User doesn't feel any interruption** in service in hot (live) migration.

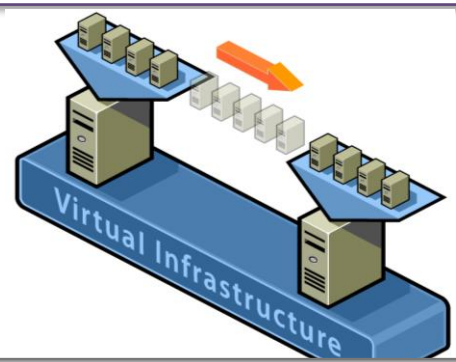
Live (Hot)VM Migration

Recall: Live Migration

- VM live migration can be a extremely **powerful tool for cluster/system administrators.**
  - Hardware / Software maintenance / upgrades
  - Load balancing / resource management
  - Distributed power management

Live Migration

- **Move VM instances across distinct physical hosts** with **little or no downtime** for running services.
  - **Services are unaware** of the migration.
  - Maintain **network connections of the guest OS**.
  - **VM is** treaded as a **black box**.
  - **VM (OS level) Migration is easier than migrating processes**



"Live Migration of Virtual Machines", Christopher Clark, Keir Fraser, Steven Hand, Jacob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, Andrew Warfield
"Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning", Michael R. Hines and Kartik Gopalan

Live Migration

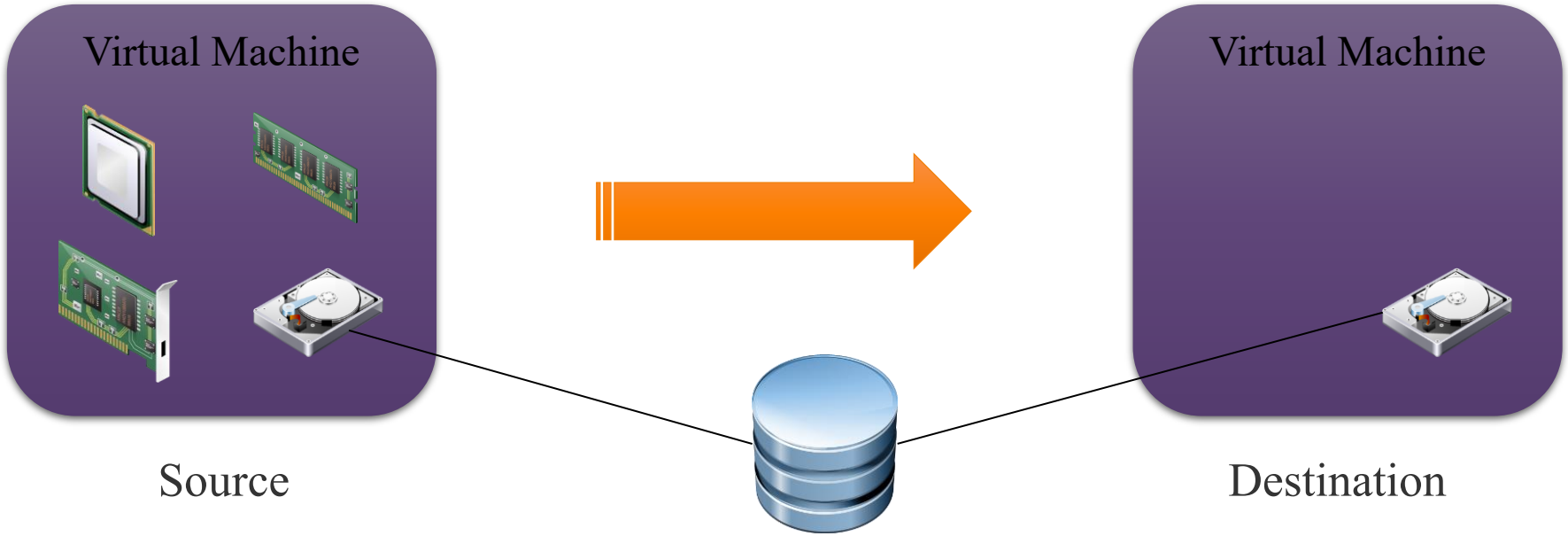
Why OS-level migration, instead of process-level?

- ***Avoid 'residual dependencies'***
  - Original host can be power-off / sleep once migration completed.
- ***Can transfer in-memory state in a consistent and efficient fashion***
  - E.g., No reconnection for media streaming application
- ***Allow a separation of concerns b/w the users and operator of a cluster***
  - Users can fully control of the software and services within their VM.
  - Operators don't care about what's occurring within the VM.

What is migrated?

- **CPU context of VM, contents of main memory**
  - Narrow interface, easier than process migration
- **Disk: assume NAS (network attached storage) that is accessible from both hosts, or local disk is mirrored**
  - We do not consider migrating disk data
- **Network: assume both hosts on same LAN**
  - Migrate IP address, advertise new MAC address to IP mapping via ARP reply
  - Migrate MAC address, let switches learn new MAC location
  - Network packets redirected to new location (with transient losses)
- **I/O devices are provisioned at target**
  - Virtual I/O devices easier to migrate.

Design-local resources



Steps to Migrate a VM

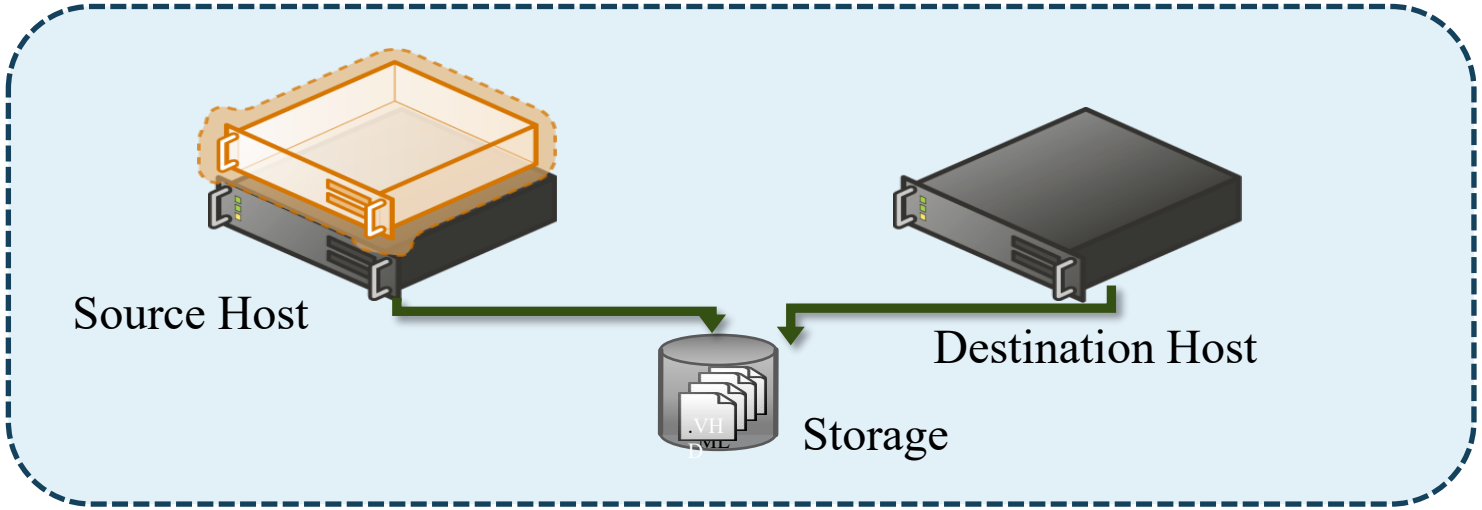
- Broad steps in any migration technique: **Suppose** we are migrating a **VM from host A to host B**
  1. **Setup target host B, reserve resources for the VM**
  2. **Push phase**: push some memory of VM from A to B
  3. **Stop-and-copy**: stop the VM at A, copy CPU context, and some memory
  4. **Pull phase**: Start VM at host B, pull any further memory required from A
  5. **Clean up state from host A, migration complete**
- **Total migration time**: time for steps 2,3,4
- **Service downtime**: time for step 3
- **Other metrics:** impact on application performance, network bandwidth consumed, total pages transferred

Flavors of migration techniques

- Pure stop-and-copy: VM stopped, **all state transferred to target**, VM restarts
  - Too much downtime to be classified as "live" migration
- Pre-copy: **most state is transferred in the push phase**, followed by a brief stop-and-copy phase
- Post-copy: VM stopped, **bare minimum state required to run the VM is transferred to the target host. Remaining state is pulled on demand** while the VM is running at the new location.
- Hybrid: **a mix of pre-copy and post-copy**. Some pushing of state followed by stop-and-copy, followed by pulling of state on demand.

Design-challenges

- ❑ Minimize service downtime
- ❑ Minimize migration duration
- ❑ Avoid disrupting running service



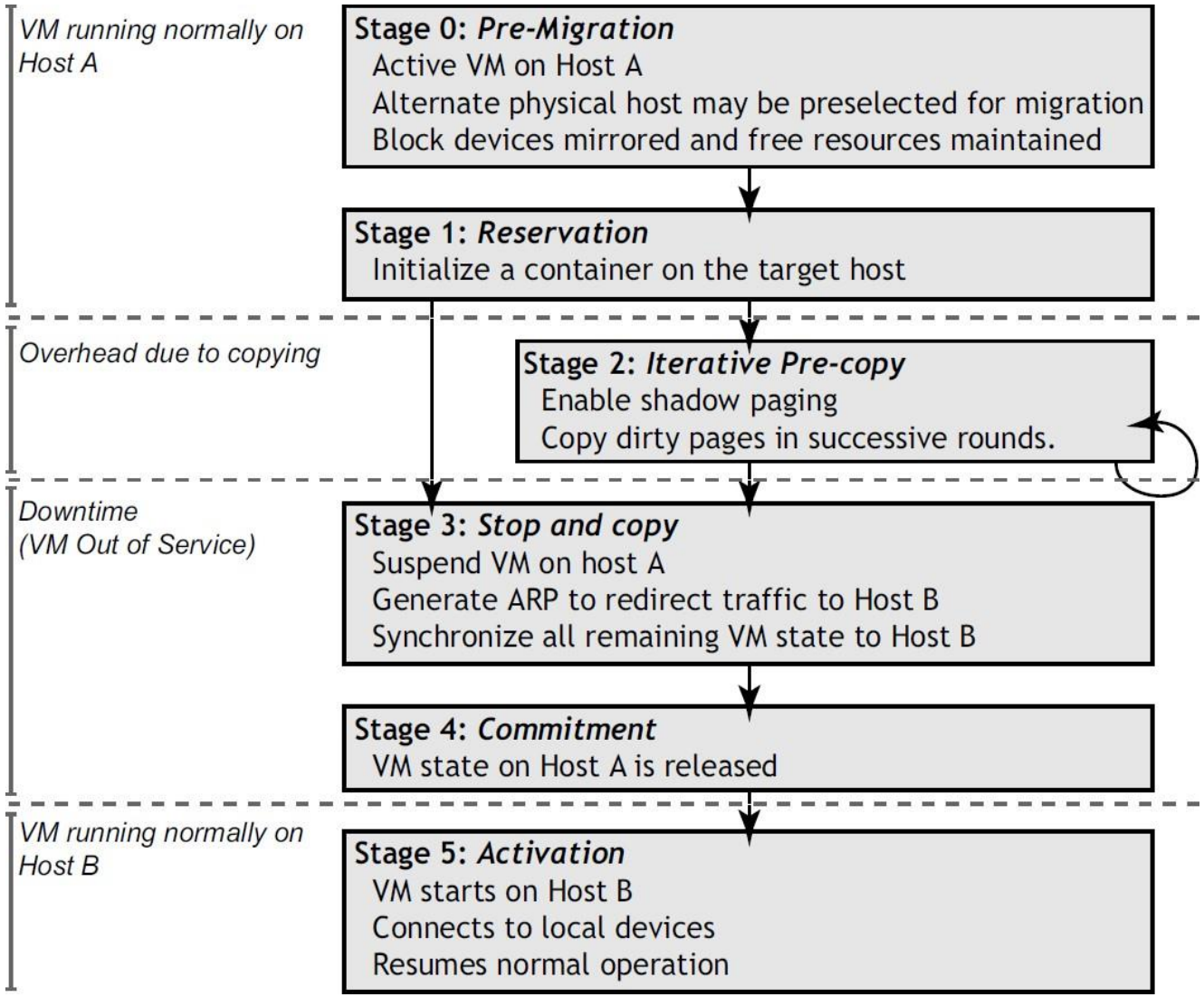
Design-memory migration

Phase	service downtime	migration duration
push	-	-
stop-and-copy	longest	shortest
pull (demand)	shortest	longest

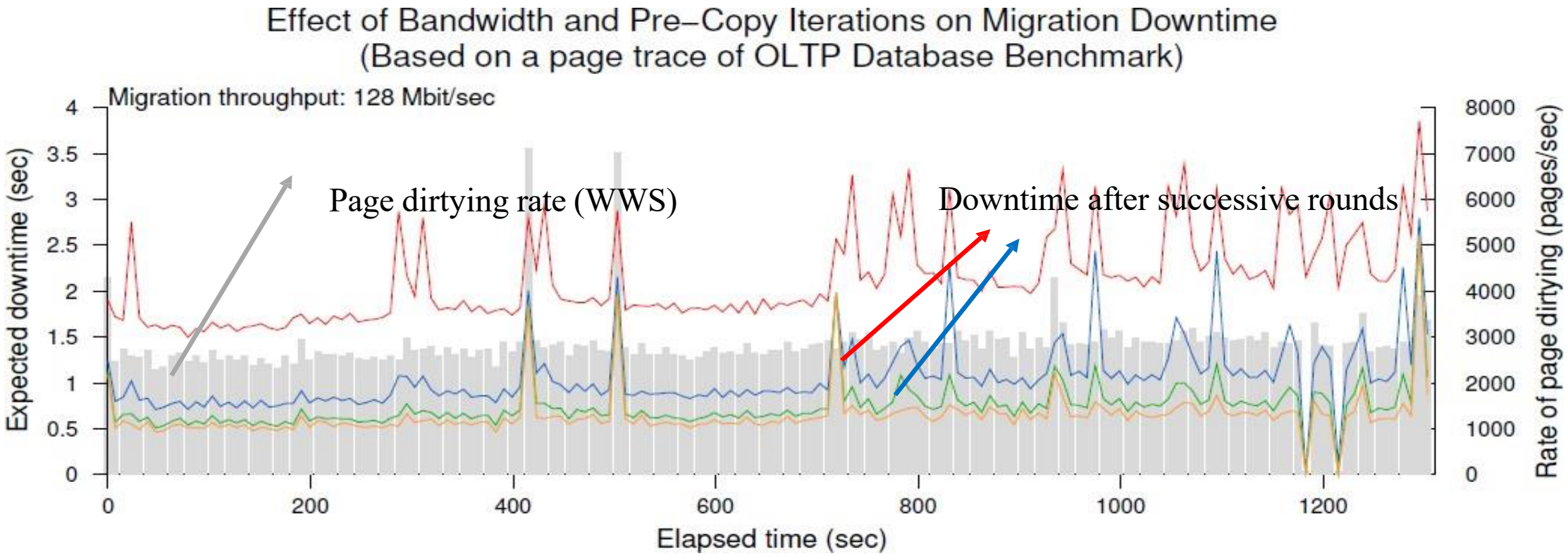
- Careful to avoid service degradation

Pre-copy based live migration

- Iterative pre-copy + stop-and-copy for remaining memory
- First push round copies all pages
- Every round copies pages dirtied in previous round
  - A page maybe copied multiple times
- Writable Working Set (WWS): pages commonly written to
  - WWS will be copied multiple times
  - Finally transferred in stop-and-copy
- **How many rounds?**
- Stop when rate of dirtying > rate of transfer



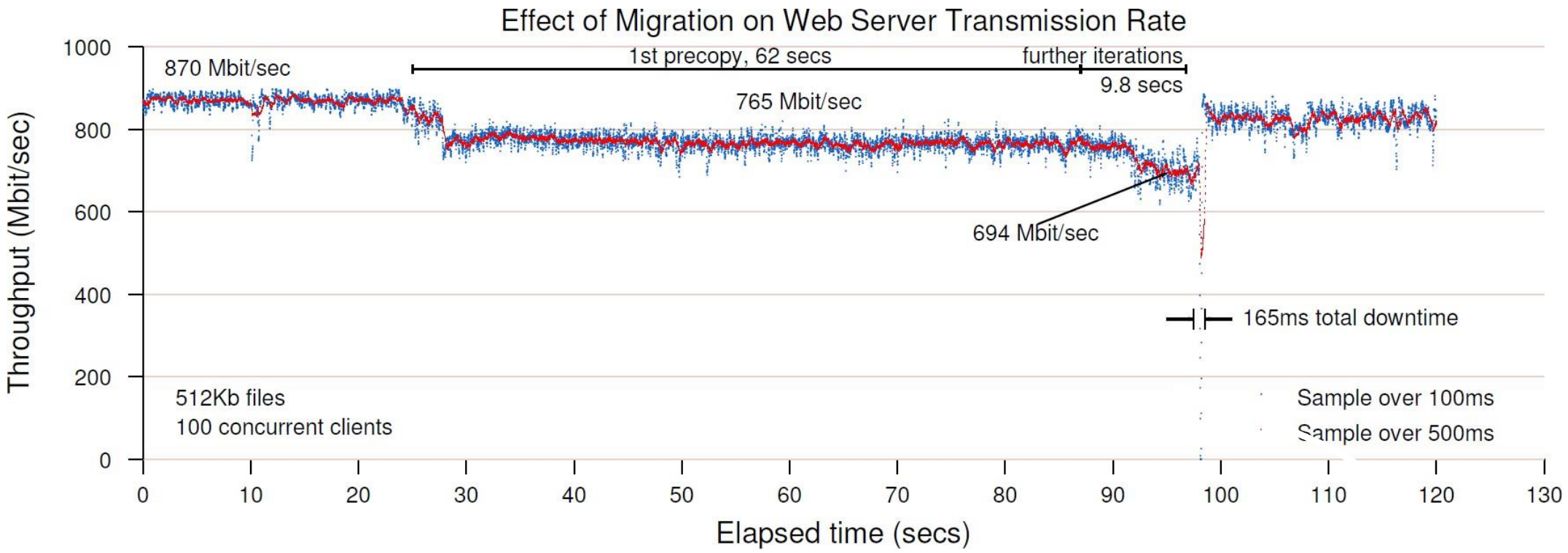
Impact of iterative pre-copy



- If stop-and-copy, 512MB VM, 128 Mbps network, downtime = 32 sec
- With one pre-copy round, downtime goes to 2-3 sec
  - ~1 second for 2 or more rounds

Pre-copy performance

- Downtime: ~100 millisec, total migration time of few tens of seconds
- Worse for memory-intensive applications, better for interactive apps



Implementation-dynamic rate limiting

More network bandwidth, less service downtime !
Less network bandwidth, less impact on running service !
Dynamically adapt the bandwidth limit during each round

-Set a minimum and a maximum bandwidth limit, begin with the minimum limit

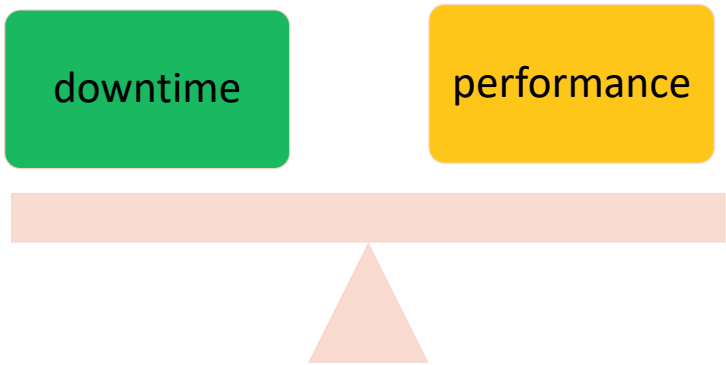
-$bandwidth_{next} = \text{dirty } rate_{\text{current}} + constant\ increment$

-$dirty\ rate_{current} = {}^{dirty\ pages}/_{duration}$

When terminate push, and switch to stop-and-copy ?

-$dirty\ rate_{\text{current}} > bandwidth_{max}$

-$dirty\ pages < threshold$

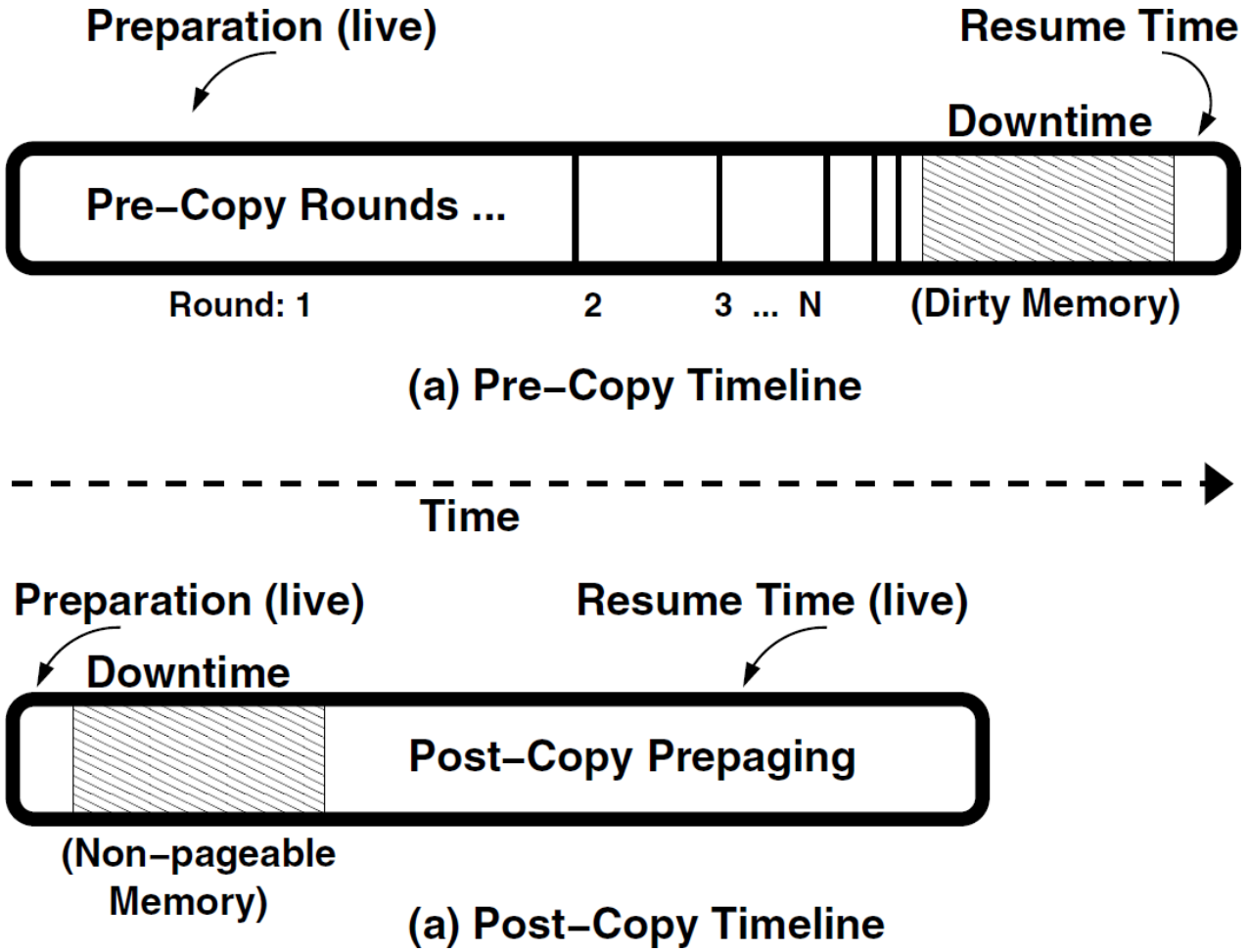


Some optimizations

- Avoid transferring page multiple times
  - Before transmitting page, peek into the current round's dirty bitmap
  - Skip transmission if page is already dirtied in ongoing round
- Move non-interactive processes generating dirty pages to wait queue
  - Execution paused until migration completes
- Free up page cache and other unnecessary pages
  - Reduce memory footprint
  - Much like ballooning

Pre-copy vs Post-copy livemigration

- Avoid multiple transfers of same page as happens in pre-copy
- Prepare target, stop VM, copy CPU context and minimum memory to target
- Start VM at target, pull memory from source via demand paging
  - Memory access at target causes page fault, page fetched from source



Optimizations

- Active pushing: source proactively pushes important pages, in addition to pulling pages via page faults
- Pre-paging: a "bubble" of pages around faulted page and proactively pushed, in anticipation of future accesses
- Dynamic self-ballooning: VM periodically frees up unnecessary memory and gives it back to hypervisor
  - Reduces memory footprint, speeds up page transfer
  - Performed carefully without hurting application performance
  - Can be used to optimize pre-copy migration as well
- Hybrid: one pre-copy round, followed by post copy

Post copy vs pre copy performance

- Longer downtime as compared to pre-copy, but lower total migration time, fewer page transfers, lesser disruption to application

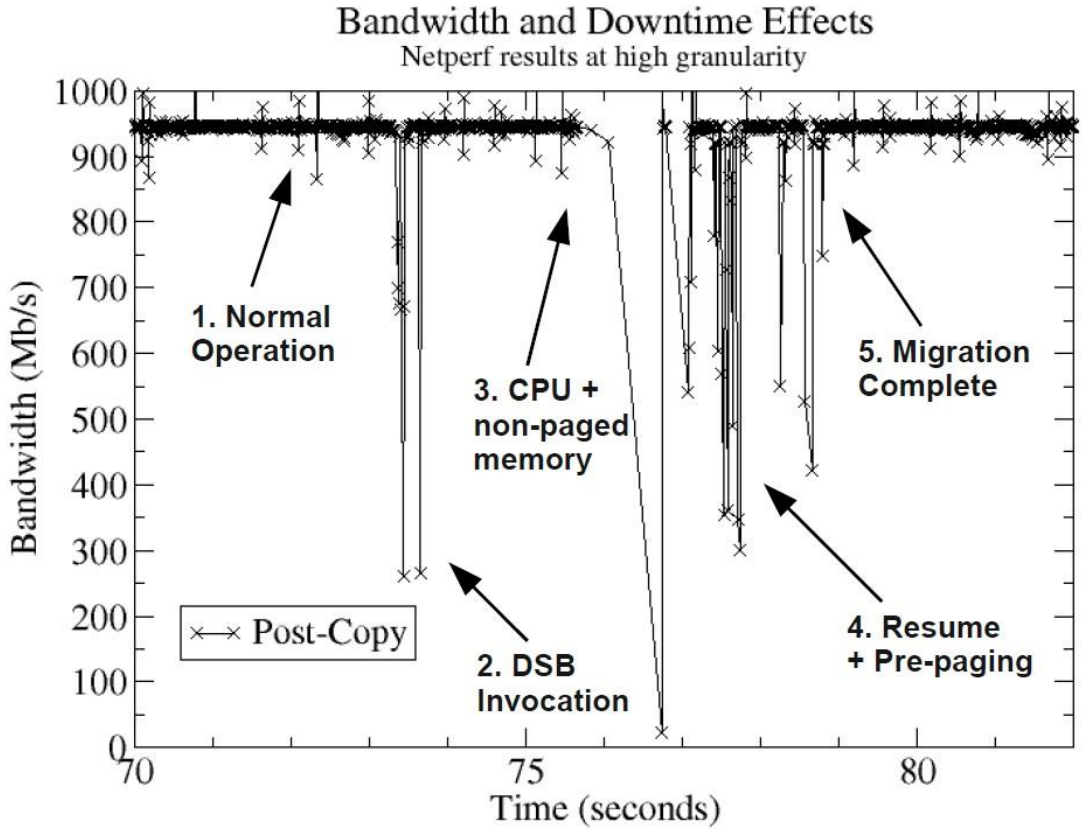


Figure 9. Impact of post-copy on NetPerf bandwidth.

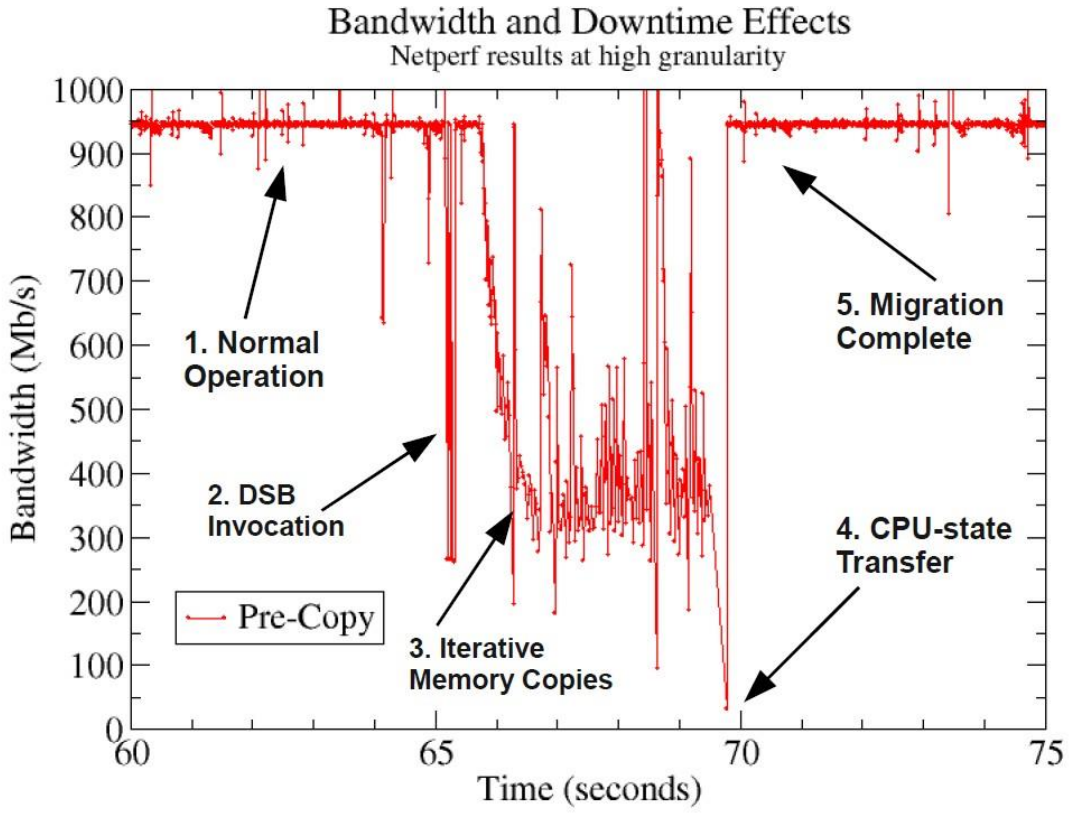
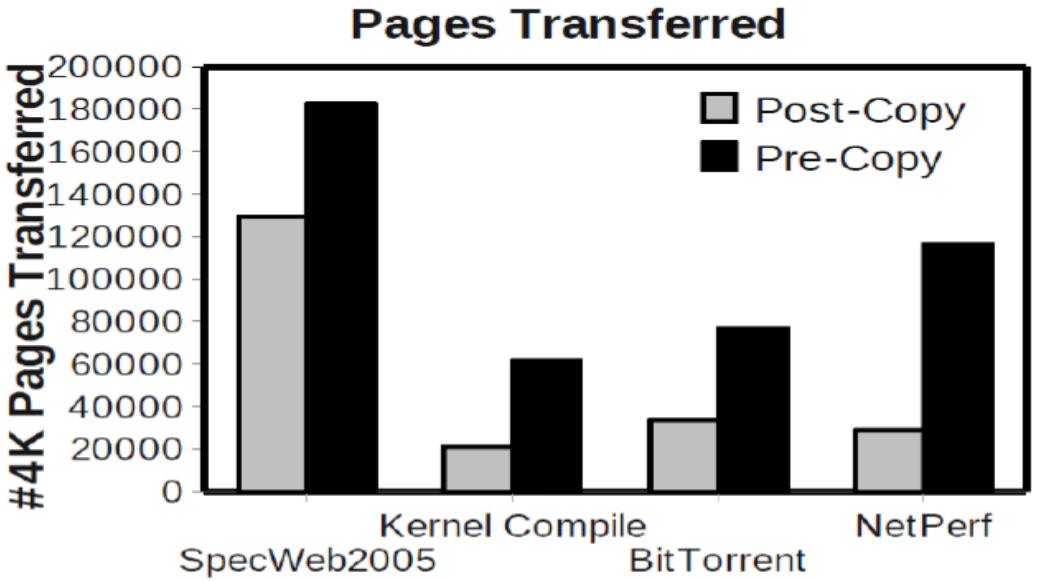
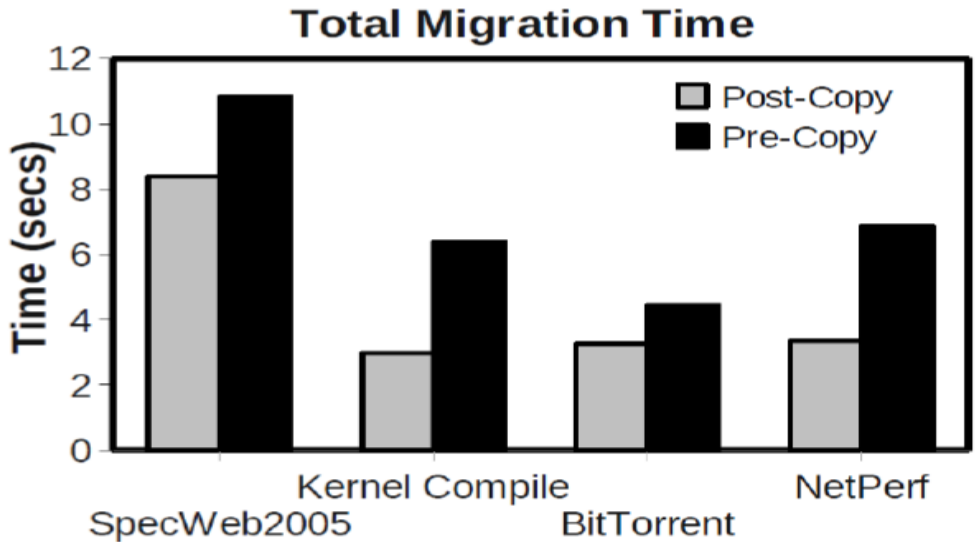
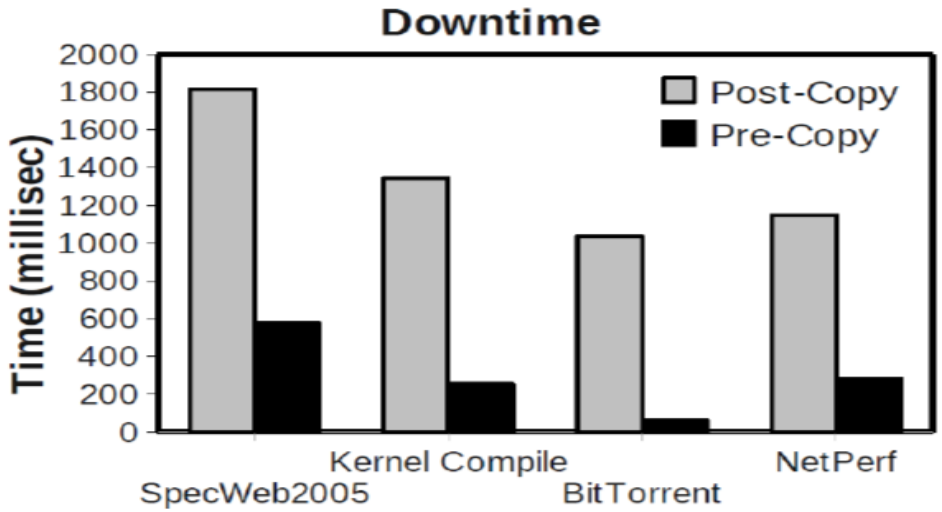


Figure 10. Impact of pre-copy on NetPerf bandwidth.

Post copy vs pre copy performance







What about failures?

- What if **target machine fails during migration**?
- **Pre-copy can simply abort the migration**, restart with another target
  - With pre-copy, **latest state is on source only**, so can recover
- With post copy, **source has stale memory**, target has updated memory
  - If target crashes during post copy, cannot recover application data (unless some replication is performed)

Summary

- VM live migration techniques
  - Iterative pre-copy vs post-copy via demand paging
  - Implementation details on Xen
  - Performance comparison
- Dynamically adapting network-bandwidth
  -Balance service downtime and service performance degradation
- Which is better?
  - Pre-copy suited for interactive application
  - Post copy is better for memory-intensive applications with large WWS
  - Hybrid techniques are also used

IntelliPaat

eict IIT Roorkee

IIT Roorkee

India: +91-7022374614

US: 1-800-216-8930 (TOLL FREE)

support@intellipaat.com

24/7 Chat with Our Course Advisor