

Status Update About COLO

(COLO: COarse-grain LOck-stepping Virtual Machines for Non-stop Service)

eddie.dong@intel.com

arei.gonglei@huawei.com

yanghy@cn.fujitsu.com



Agenda

- Background
- Introduction Of COLO
- Current Status
- Performance
- Summary

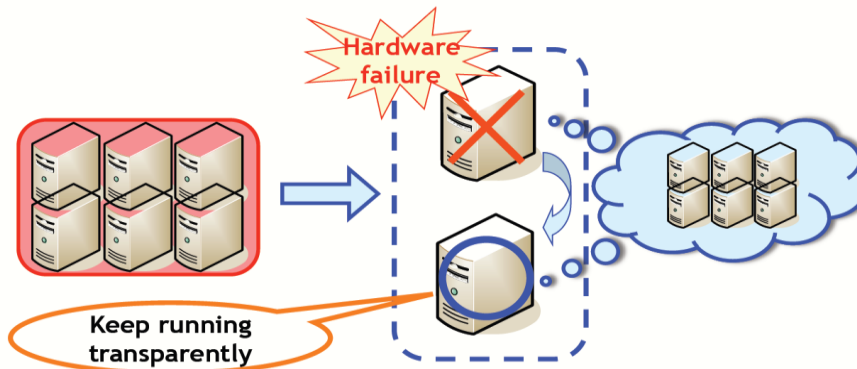


2



What is COLO ?

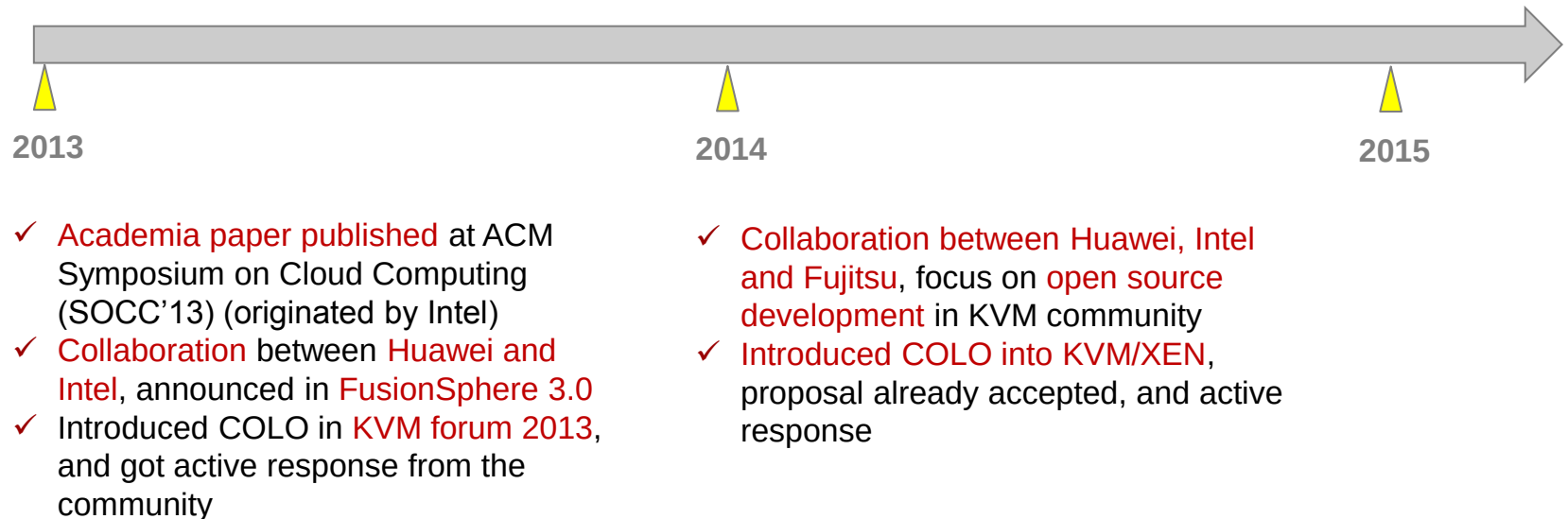
COLO(COarse-grain LOfk-stepping) is an ideal Application-agnostic Solution for **Non-stop service in the cloud.**



- Typical Non-stop Service Requires
 - Expensive hardware for redundancy
 - Extensive software customization
- COLO
 - Cheap Application-agnostic Non-stop Virtual Machine

What Happened ?

Goal: make COLO upstream into KVM/XEN



Agenda

- Background
- Introduction Of COLO
- Current Status
- Performance
- Summary



5



Non-stop Service Solutions

- Hardware Solution
 - Robust Hardware Components
 - Very expensive
 - Still Unpredictable
 - such as alpha-particles in cosmic
- Software Solution
 - Replication
 - Construct backup replicas at run time
 - Failover: backups take over when the primary fails



6



Different Layer of Replication

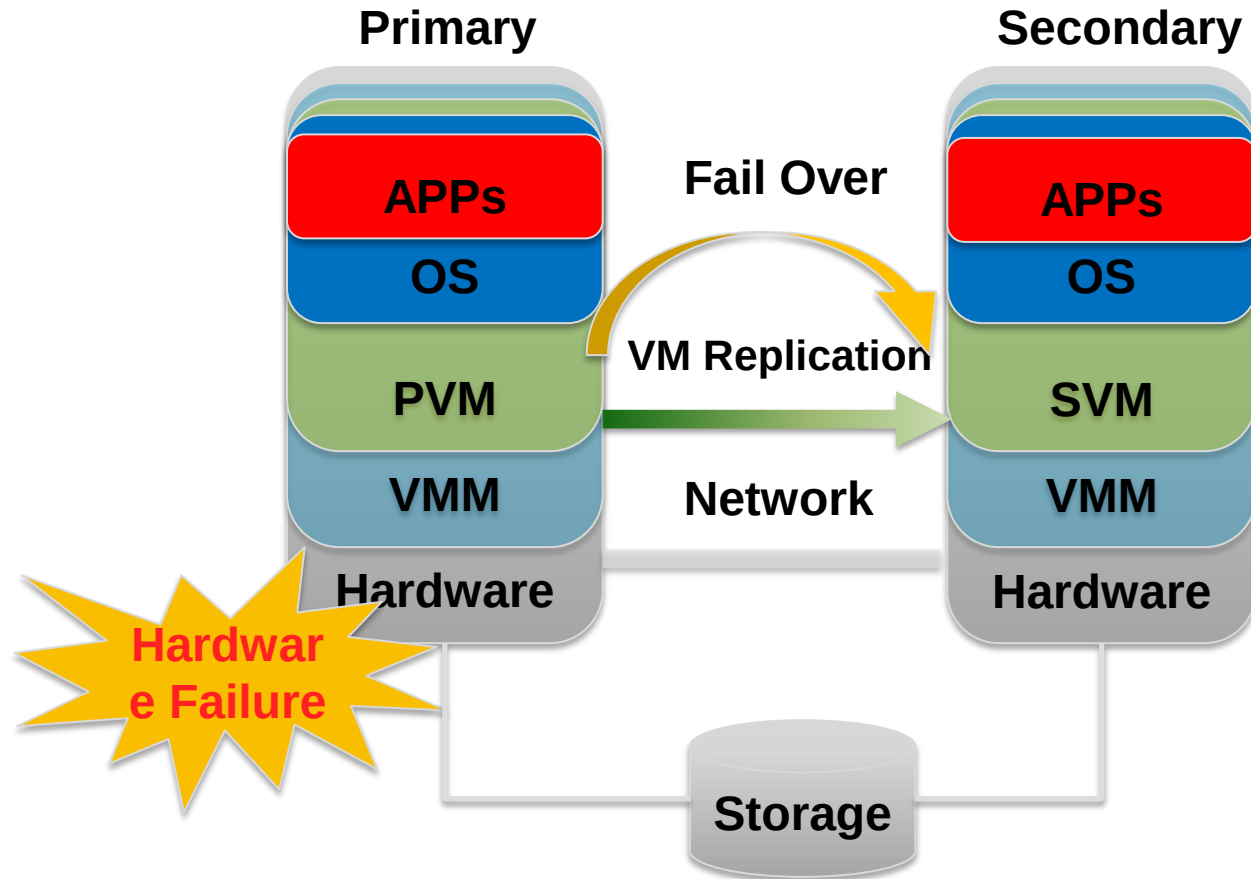
- Application layer
 - Extensive software customization
 - Impractical to legacy software
- OS layer
 - Large complexity, not commoditized
- Virtual machine layer
 - Application and OS agnostic



7



VM Replication



Existing VM Replication Approaches

- Lock-stepping: Replicating per instruction
 - Execute in parallel for deterministic instructions
 - Lock and step for nondeterministic instructions
- Checkpoint: Replicating per epoch
 - Output is buffered within an epoch
 - Exact machine state matching from external observers

Replicating Exact Machine State from PVM to SVM



9



Problems

- Lock-stepping
 - Excessive replication overhead for MP-guest
 - memory access in an MP-guest is nondeterministic
- Periodic Checkpoint
 - Extra network latency
 - Excessive VM checkpoint overhead

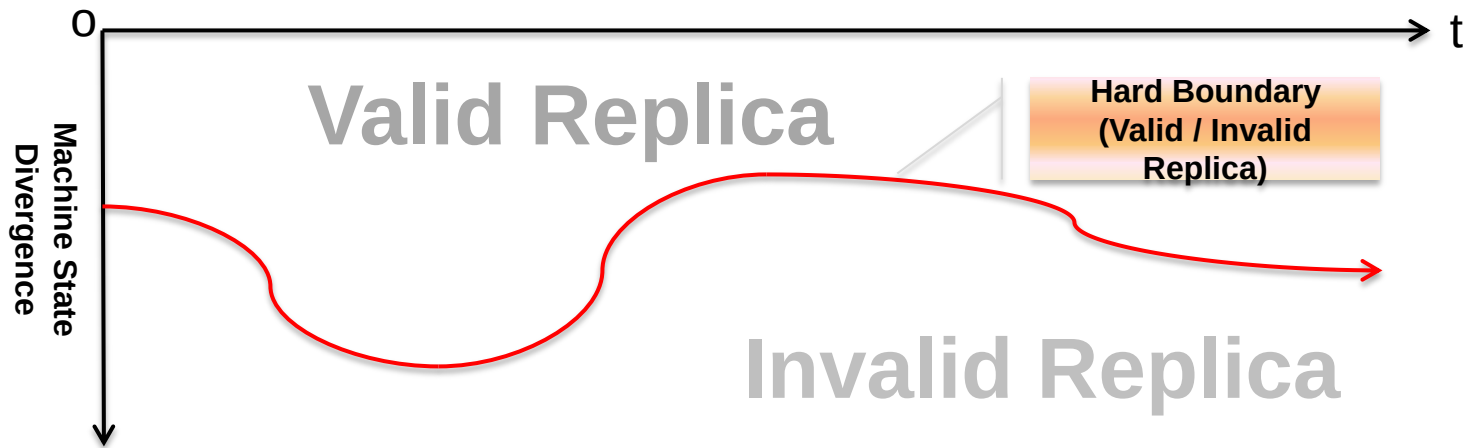


10



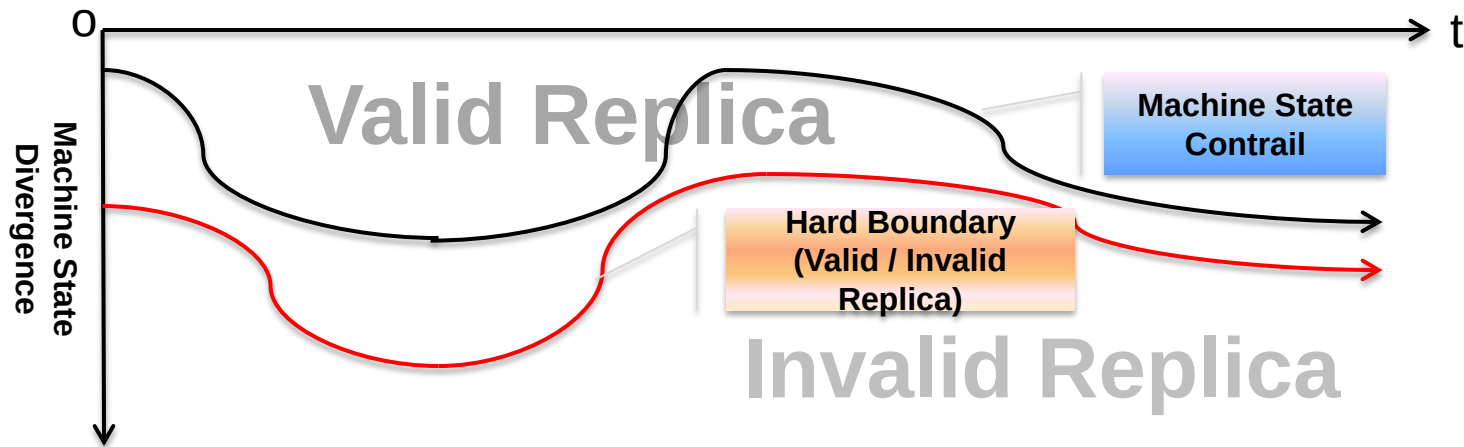
Why Exact Machine State Matching ?

- Valid / Invalid replica (SVM)
 - Be able / unable to take over the service respecting the application semantics
- Exact machine state matching is an overly strong condition



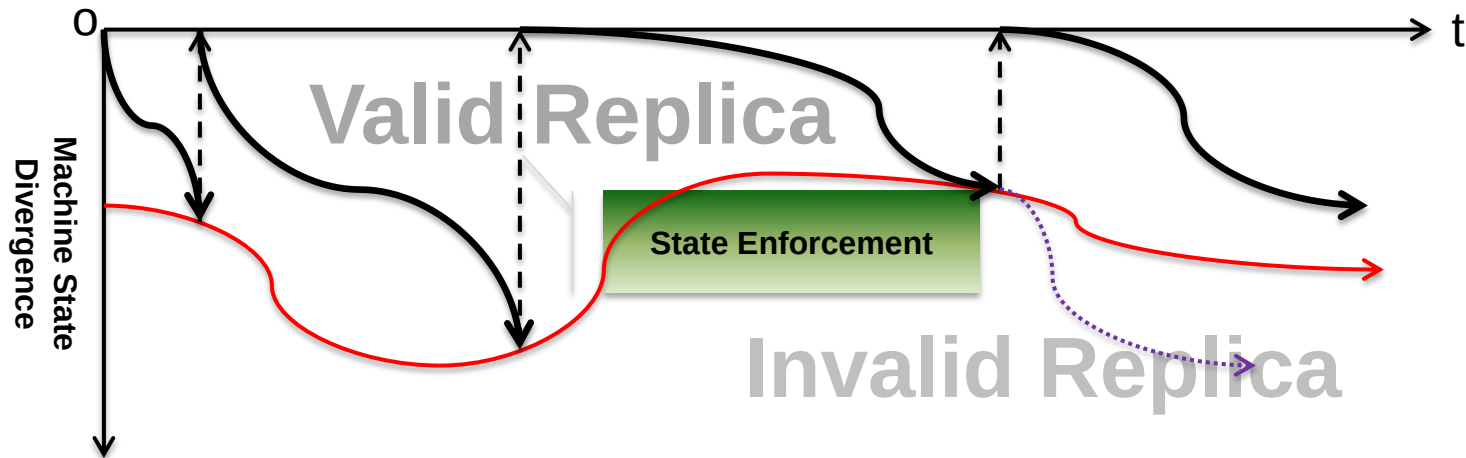
COarse-grain LOck-stepping (COLO)

- Replicating with less machine state matching
 - Executes in parallel, as if the machine state is within the hard boundary
 - Non-stop service system $\longleftrightarrow$ machine state contrail fully within the hard boundary



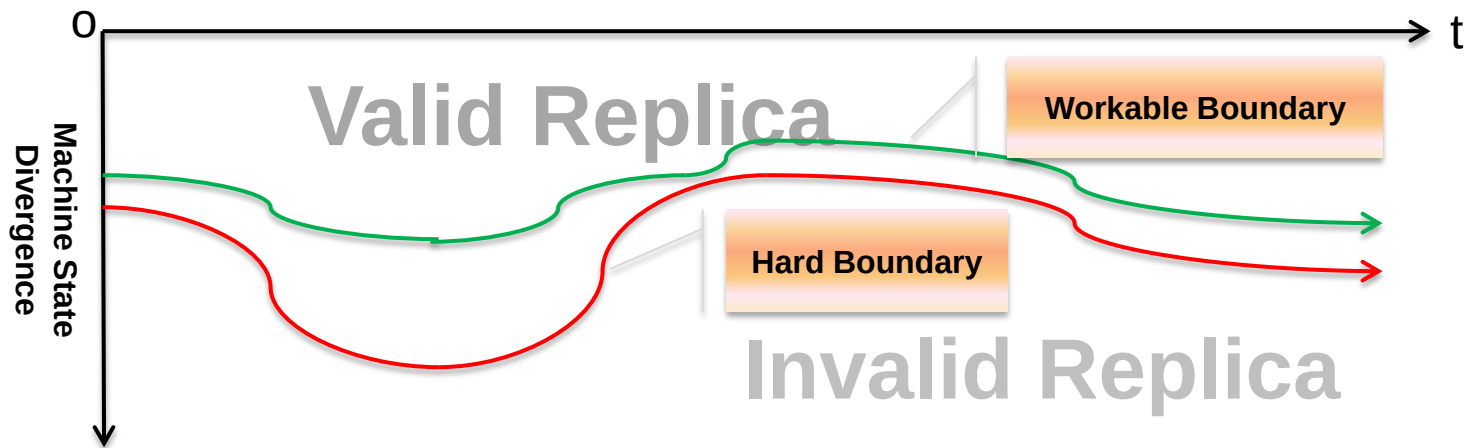
Challenge – 1

- How to guarantee the machine state contrail is fully within the hard boundary
 - The initial SVM state may be identical to the PVM
 - May enforce machine state matching any time



Challenge - 2

- How to identify the boundary ?
 - Impractical to find the exact boundary of 
 - Any boundary  within the hard boundary works



Can We Find a Practical Workable Boundary?



An Example Solution

- A practical solution to COLO
 - Identify the workable boundary (↪) base on the output response
 - The initial SVM state is identical to PVM
 - SVM is a valid replica if and only if it can generate identical responses with PVM
 - SVM and PVM are feed with same inputs
 - Checkpoint if the SVM is no longer a valid replica

Further Explorations Are Required to Identify More Workable Boundary



Why It Works

- Response Model for C/S System

$$R_n = g_n(r_0, r_1, r_2, \dots, r_n, u_0, \dots, u_m)$$

- r_i & u_i are the request and the execution result of a nondeterministic instruction
- Each response packet from the equation is a semantics response

- Successfully failover at k^{th} packet if

$$C = \{R_1^P, \dots, R_k^P, R_{k+1}^S, \dots\} \quad \forall i \leq k, R_i^S = R_i^P$$

(C is the packet series the client received)

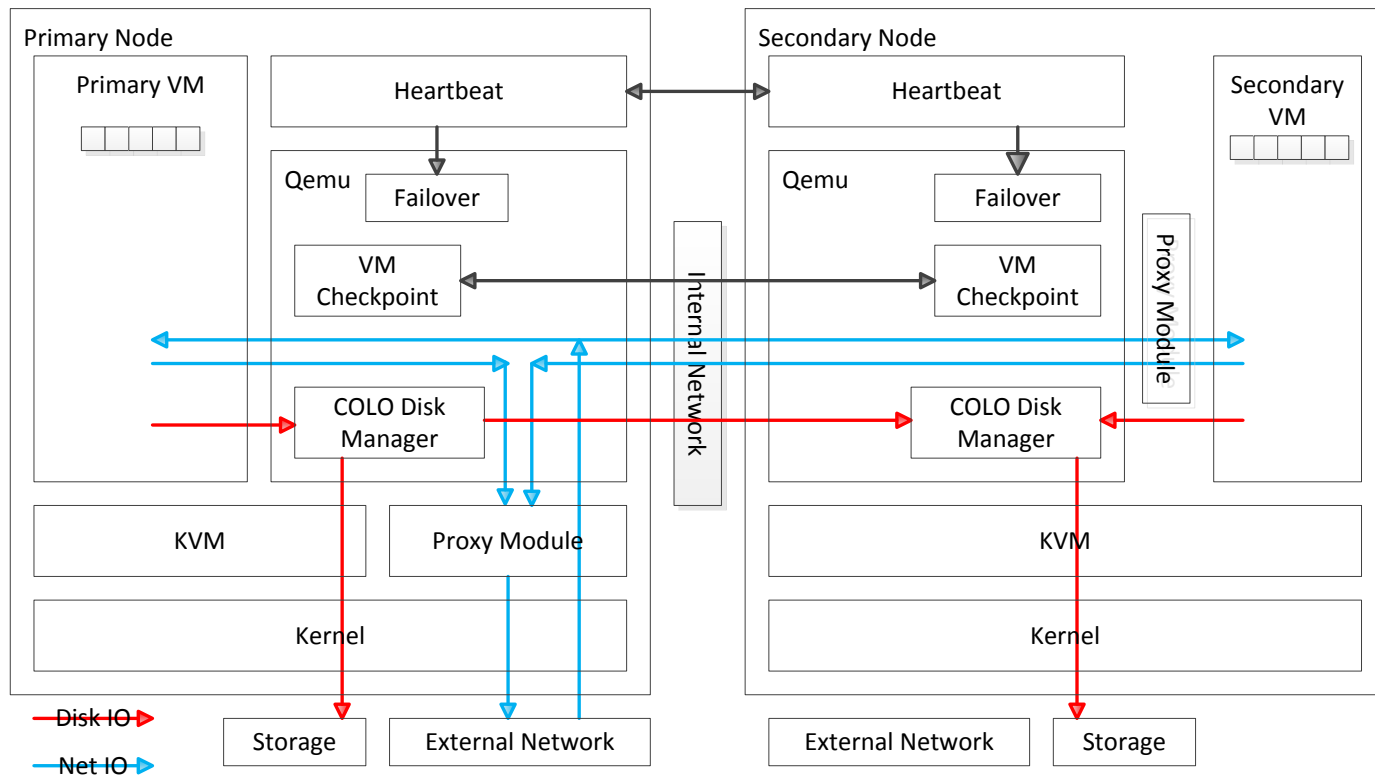


Why Better

- Comparing with Periodic VM checkpoint
 - No buffering-introduced latency
 - Less checkpoint frequency
 - On demand vs. periodic
- Comparing with lock-stepping
 - Eliminate excessive overhead of nondeterministic instruction execution due to MP-guest memory access



Architecture of COLO



Pnode: primary node; PVM: primary VM; Snode: secondary node; SVM: secondary VM

COarse-grain LOck-stepping Virtual Machine for Non-stop Service

Implementations

- Targeting fail-stop failure
 - Most hardware failures are self-corrected in modern server
 - Unrecoverable failures are fail-stop failures
- Base on Xen VM Checkpointing Solution (Remus)
 - Extend Remus passive-checkpointing to support active-checkpointing



Performance Consideration

- The frequency of Checkpoint is critical
 - Highly dependent on the *Output Similarity*, or *Response Similarity*
 - Key Focus is TCP packet!
 - Might be even worse if there are too frequent VM checkpoint
- Response similarity determines the frequency of checkpoint



20

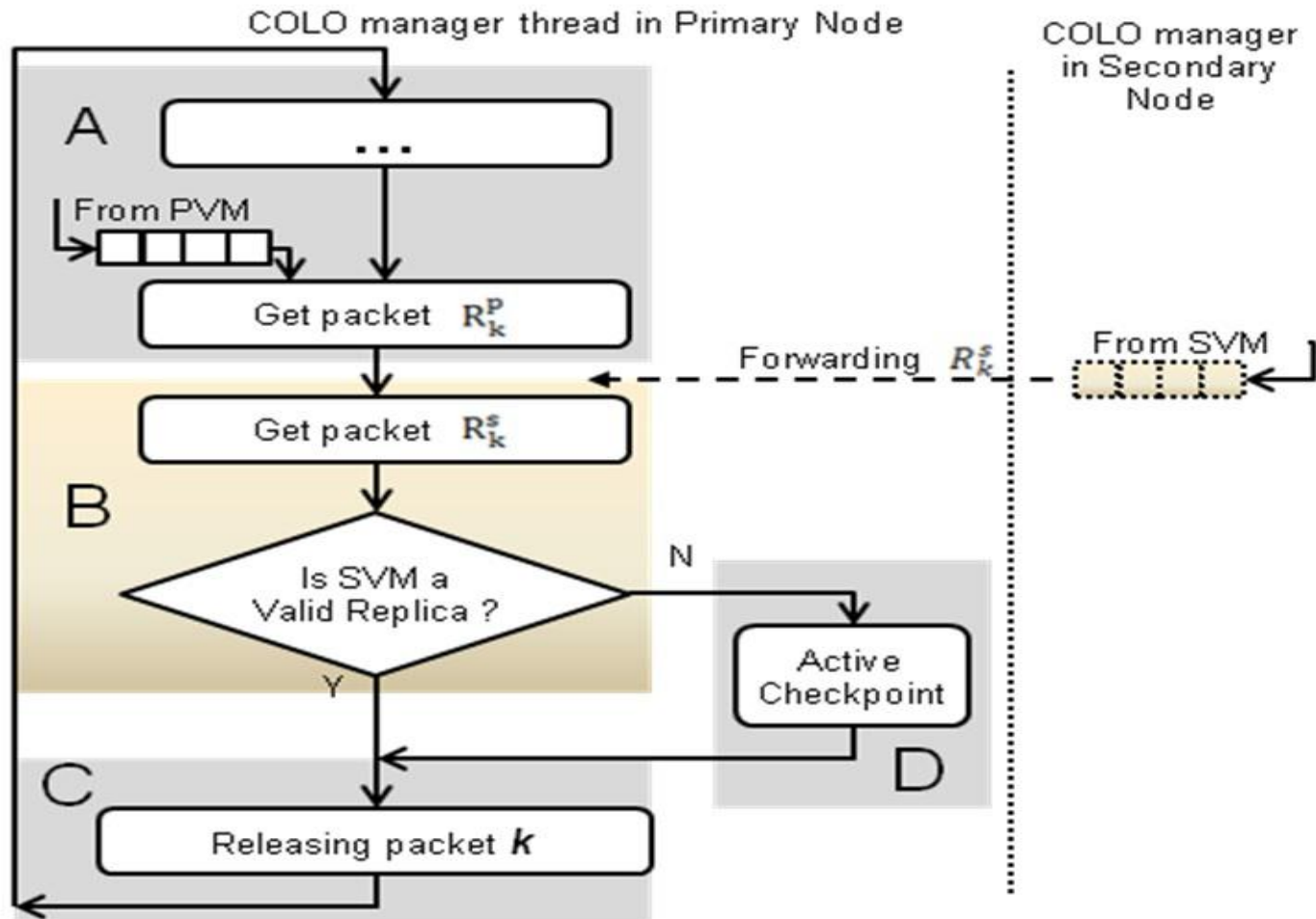


Improving Response Similarity

- Minor Modification to Guest TCP/IP Stack
 - Coarse-grained time stamps
 - Per-Connection comparison
 - Highly-deterministic ACK mechanism
 - Deterministic segmentation with Nagle algorithm
 - Coarse-grained notification Window Size



Failover Mode



Device State Lock-Stepping

- Local Storage
 - View as external interaction
 - Consider the write operations as responses to external observer → lead to increased checkpointing frequency
 - View as internal interaction
 - Write operation result is part of internal machine state
 - may not lead to immediate VM checkpointing
 - Need to forward and compare storage write operations
- Remote Storage
 - Shared access: Rely on the remote storage system
 - Exclusive access: Same policy with local storage



Agenda

- Background
- Introduction Of COLO
- **Current Status**
- Performance
- Summary



24



Current Status

- Academia paper published at ACM Symposium on Cloud Computing (SOCC'13)
 - “COLO: COarse-grained LOck-stepping Virtual Machines for Non-stop Service”
 - <http://www.socc2013.org/home/program>
 - Refer to the paper for technical details
- Industry announcement
 - Huawei FusionSphere uses COLO
 - http://e.huawei.com/en/news/global/2013/hw_308817
- Wiki pages
 - COLO on Xen:
 - http://wiki.xen.org/wiki/COLO_-_Coarse_Grain_Lock_Stepping
 - COLO on Qemu/KVM:
 - <http://wiki.qemu.org/Features/COLO>



Upstream Status

- COLO-Frame: patch series v8 has been posted on QEMU maillist.
 - <http://lists.nongnu.org/archive/html/qemu-devel/2015-07/msg05713.html>
 - [PATCH COLO-Frame v8 00/34] COarse-grain LOck-stepping(COLO) Virtual Machines for Non-stop Service (FT)
- Block replication: patch series v8 has been posted on QEMU maillist.
 - <http://lists.nongnu.org/archive/html/qemu-devel/2015-07/msg01585.html>
 - [PATCH COLO-BLOCK v8 00/18] Block replication for continuous checkpoints
- COLO-proxy in QEMU: poc been posted on QEMU maillist.
 - <http://lists.nongnu.org/archive/html/qemu-devel/2015-07/msg04069.html>
 - The netfilter for QEMU which could be useful for colo-proxy been posted v6
 - <http://lists.nongnu.org/archive/html/qemu-devel/2015-08/msg00883.html>
- Patches for Xen are sent to the mailinglist (v8)
 - <http://lists.xenproject.org/archives/html/xen-devel/2015-07/msg02911.html>
 - [PATCH v8 --for 4.6 COLO 00/25] COarse-grain LOck-stepping Virtual Machines for Non-stop Service



Agenda

- Background
- Introduction Of COLO
- Current Status
- **Performance**
- Summary



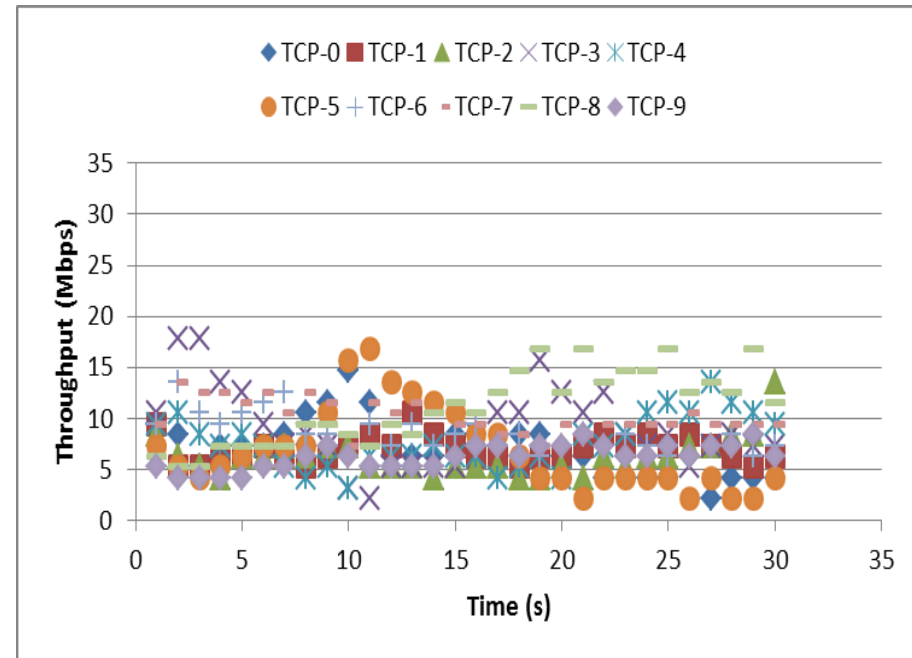
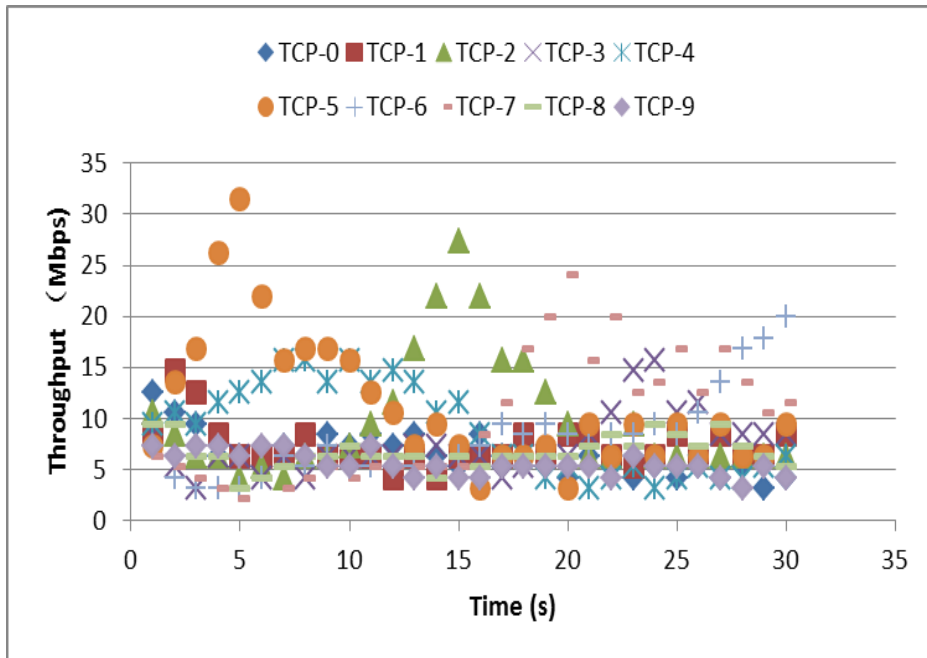
Configurations & Benchmarking

- Hardware
 - 2.7 GHz 8-core Xeon processor, 128 GB Ram
 - 1 Gbps external conn, 10 Gbps internal conn
- Software
 - Guest: RHEL5U5 with 2 GB memory, PV disk/NIC
 - Dom0: RHEL6U1 (kernel updated to 3.2)
 - Hypervisor: Xen 4.1, 10 MB log buffer for disk write operations
- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.



Impact of TCP/IP Modification

- Concurrent TCP connections in COLO native TCP/IP stack in WAN
 - Total BW: 80 Mbps, stddev: 4.4
- Concurrent TCP connections in COLO modified TCP/IP stack in WAN
 - Total BW: 80 Mbps, stddev: 3.1

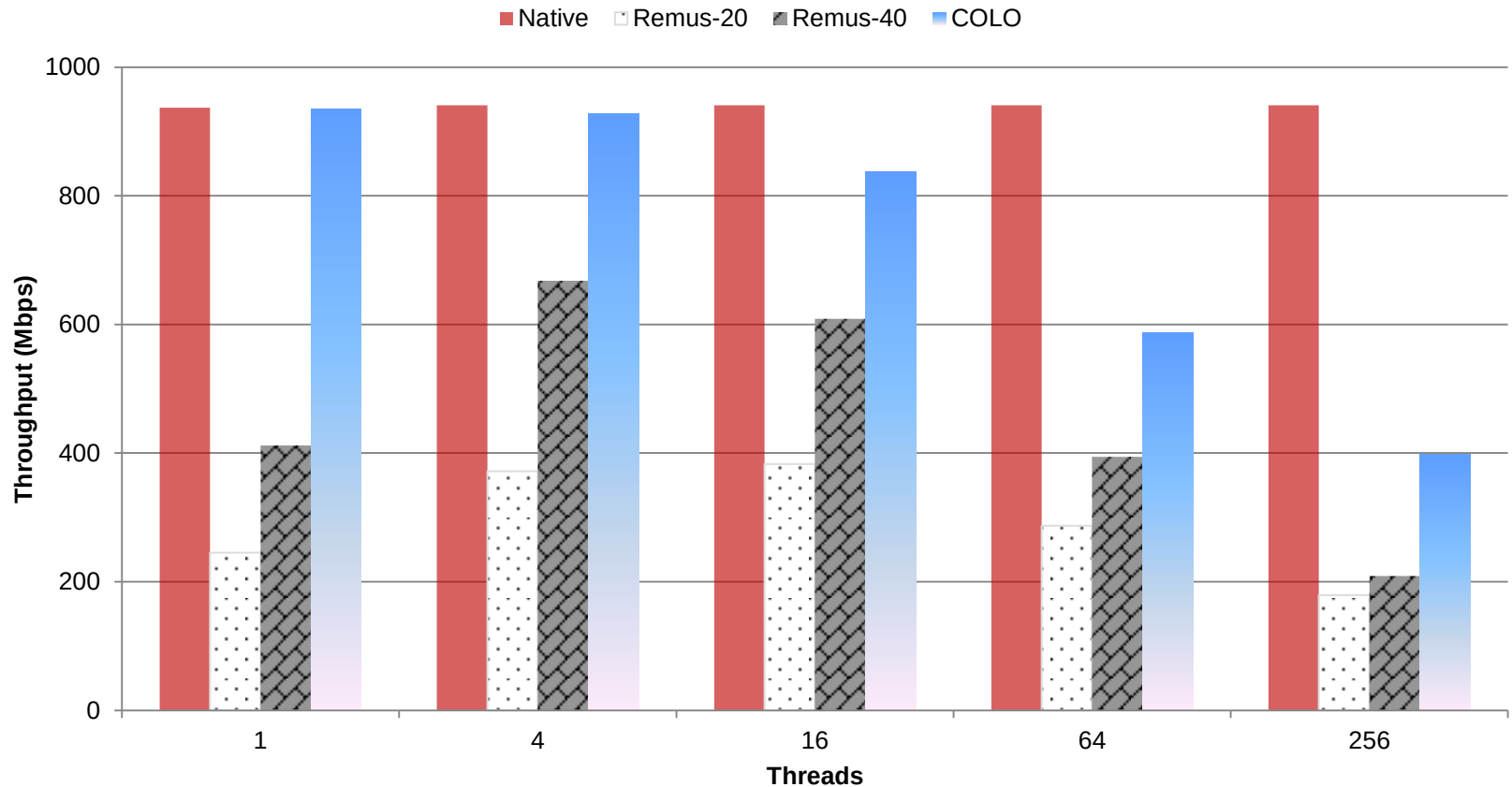


Source: Intel

For more complete information about performance and benchmark results, visit www.intel.com/benchmarks



Web Server Performance - Web Bench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

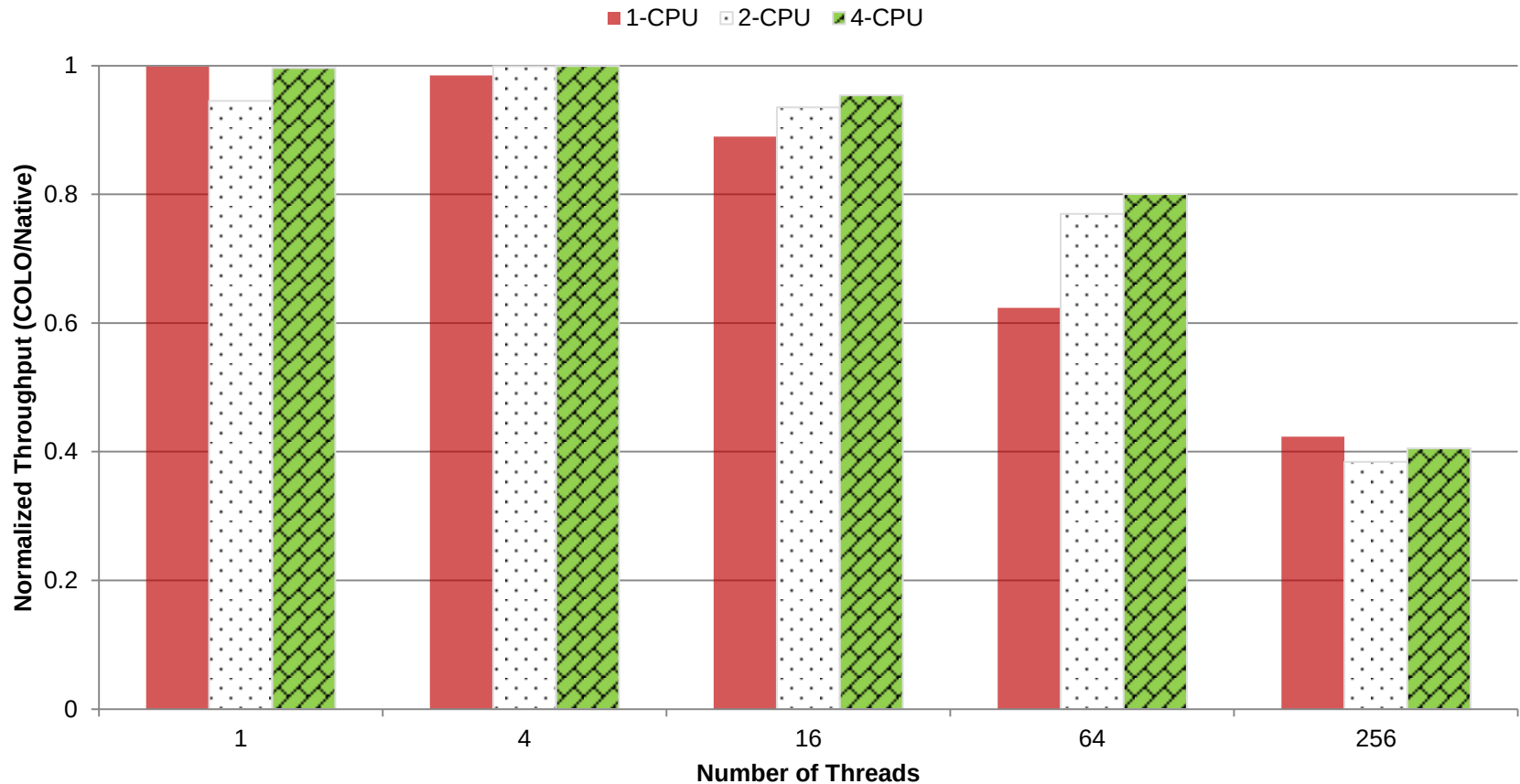
Source: Intel



30



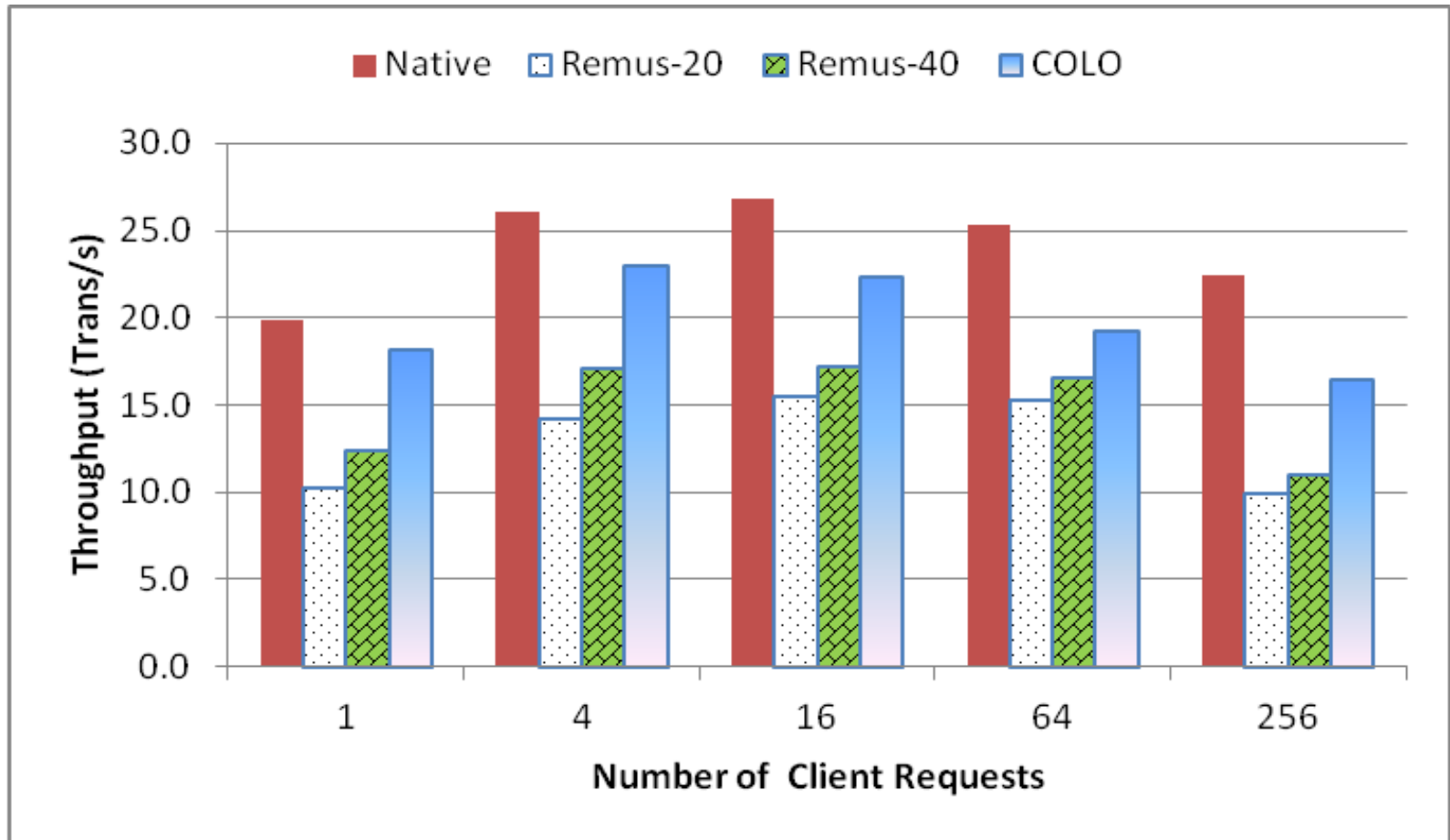
Web Server Performance - Web Bench (MP)



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

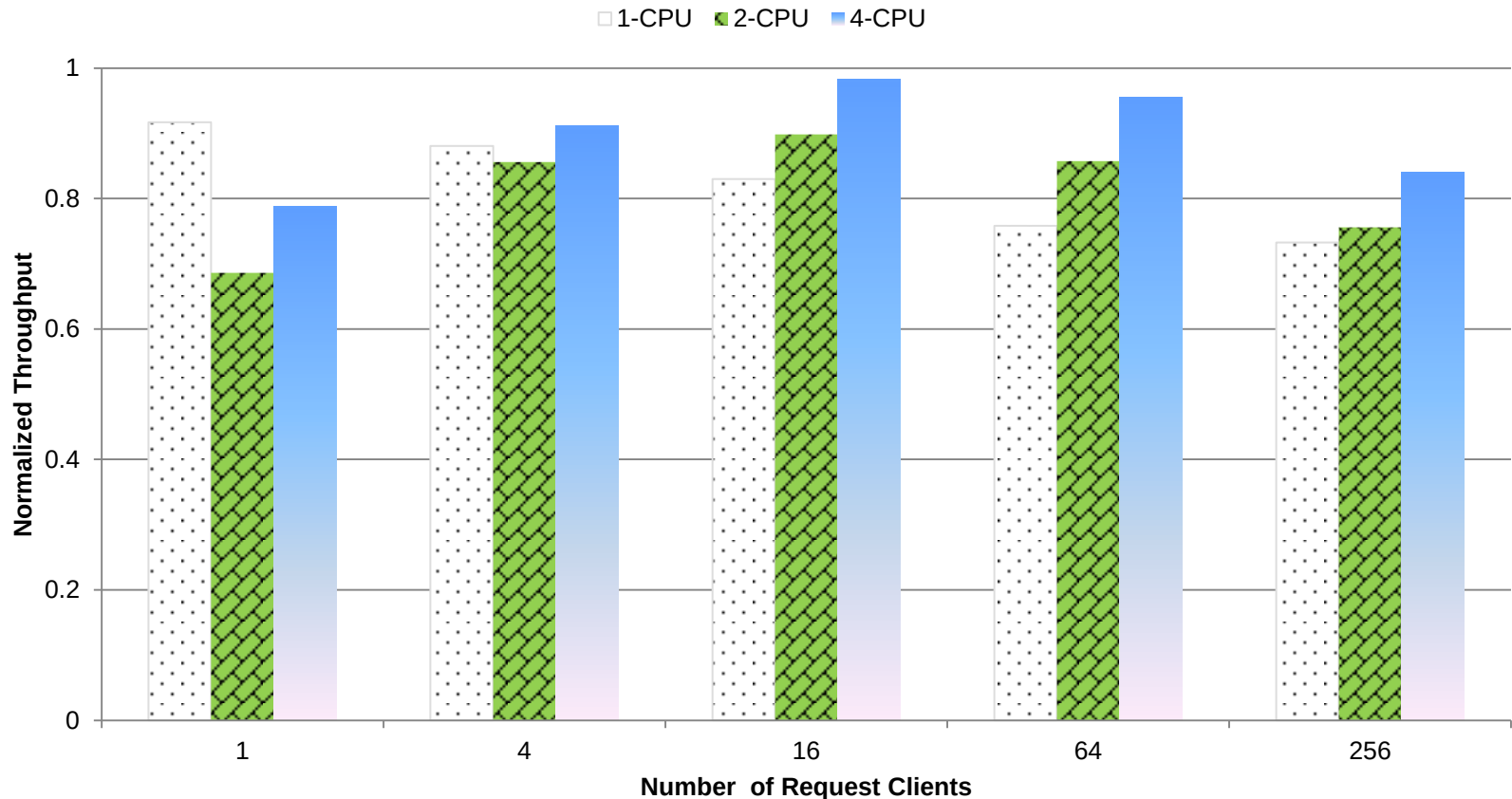
PostgreSQL Performance - Pgbench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

PostgreSQL Performance - Pgbench (MP)



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

Agenda

- Background
- Introduction Of COLO
- Current Status
- Performance
- Summary



34



Summary

- COLO can achieve native performance for CPU-intensive workload
- COLO is MP-neutral, and outperforms Remus by 69% in Web Bench, and 46% in Pgbench, respectively
- Next steps
 - Fix based on reviewing comments
 - Optimize performance
 - Anything else?



35



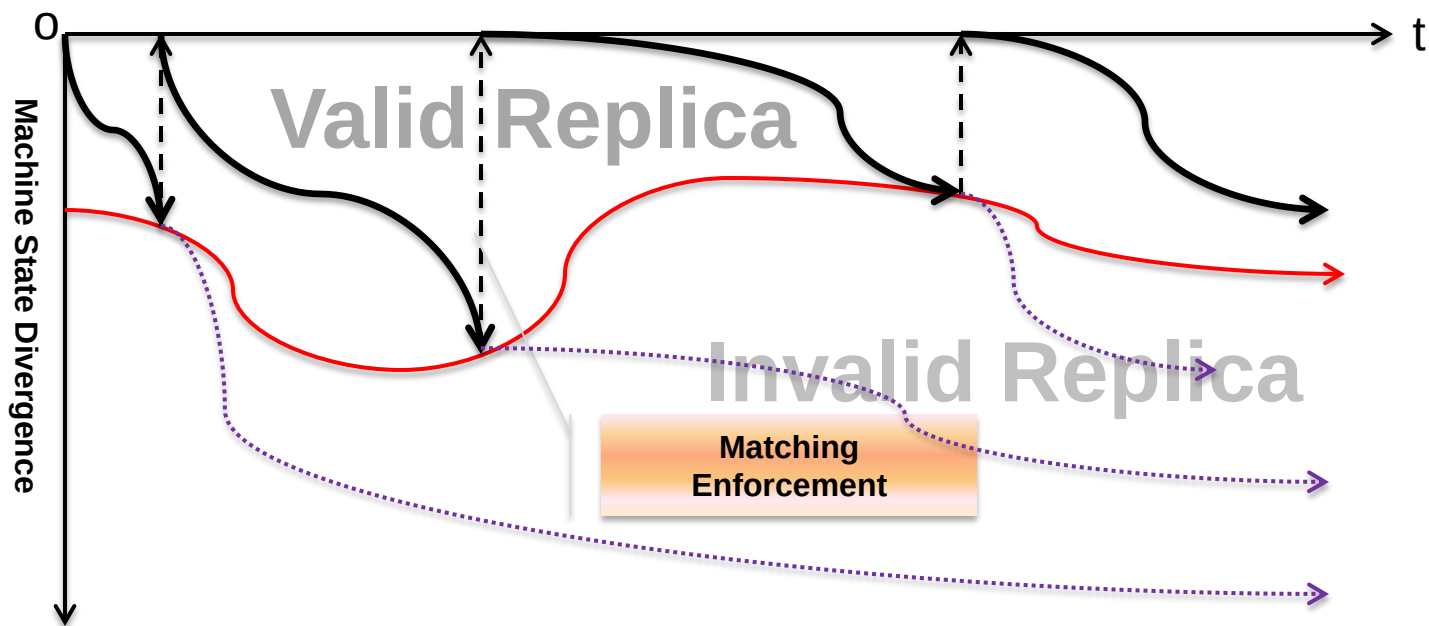
Backups



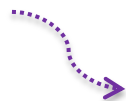
36



Machine State Evolvment in COLO



Boundary



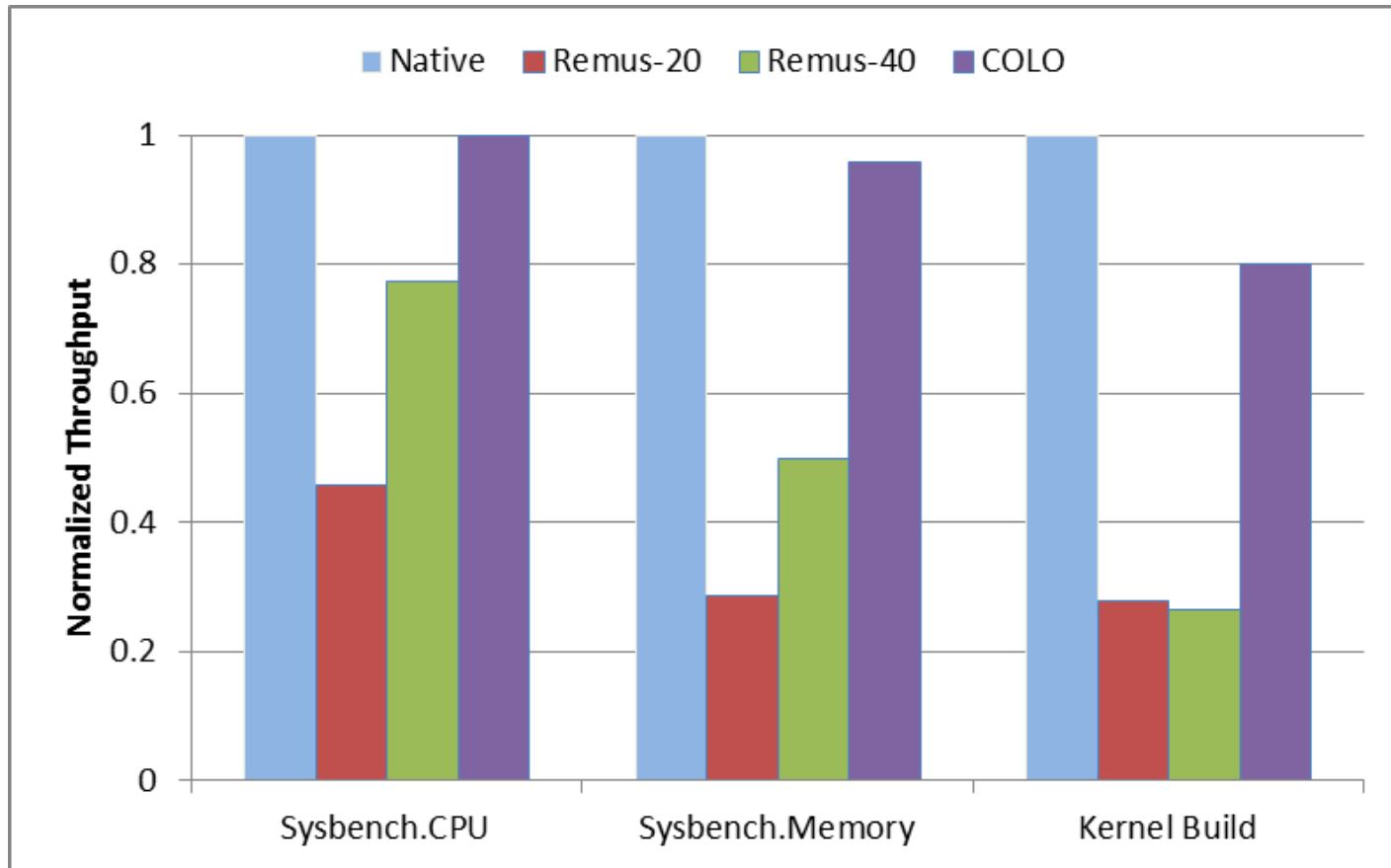
Machine State
Evolvment w/o matching
enforcement



Machine State
Evolvment in COLO



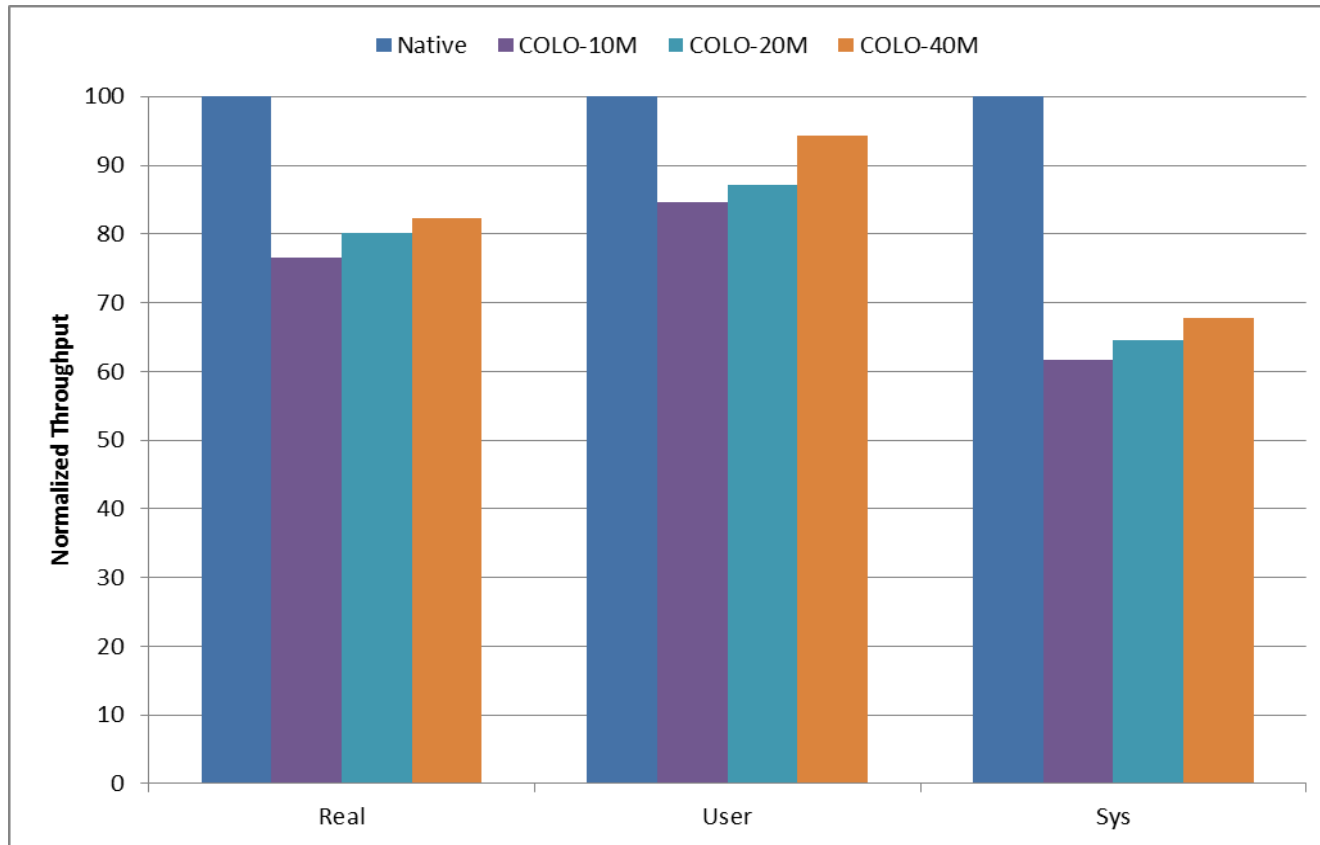
Sysbench & Kernel Build



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

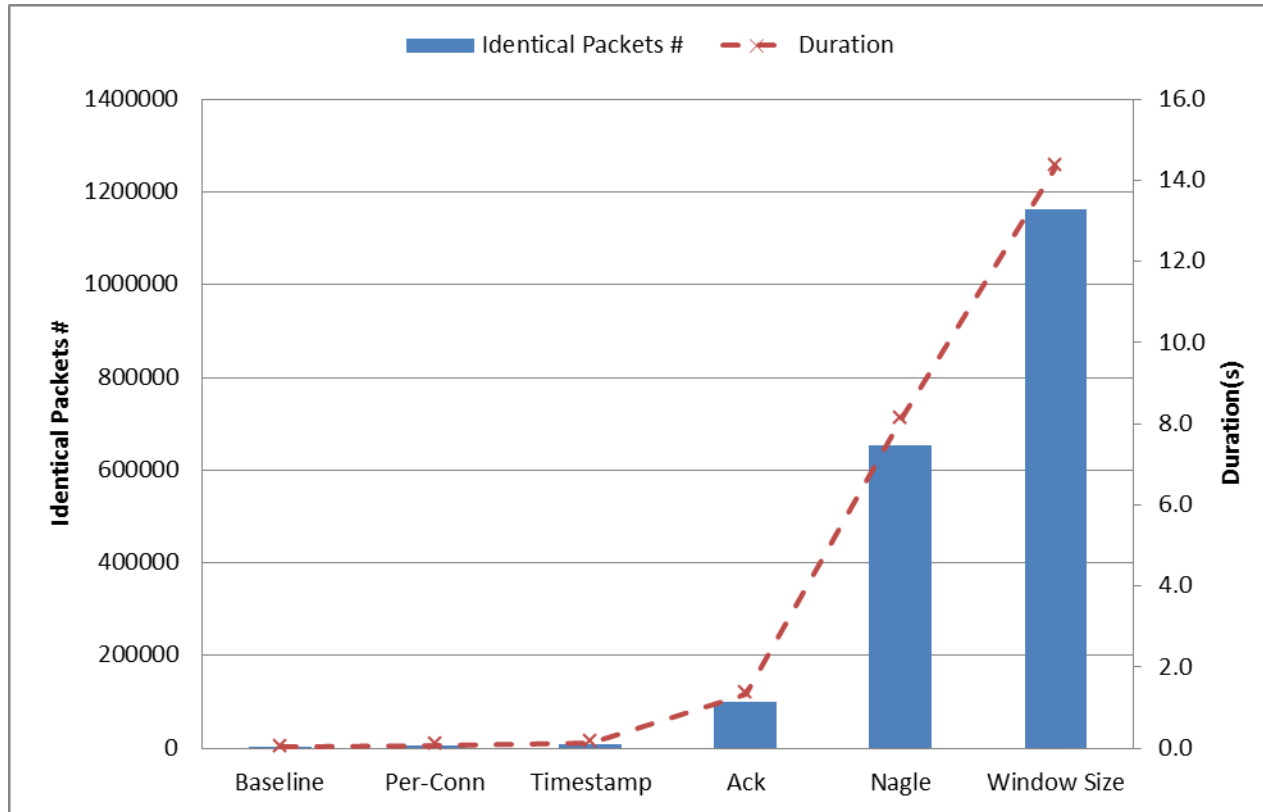
Kernel Build vs. Log Buffer Size



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

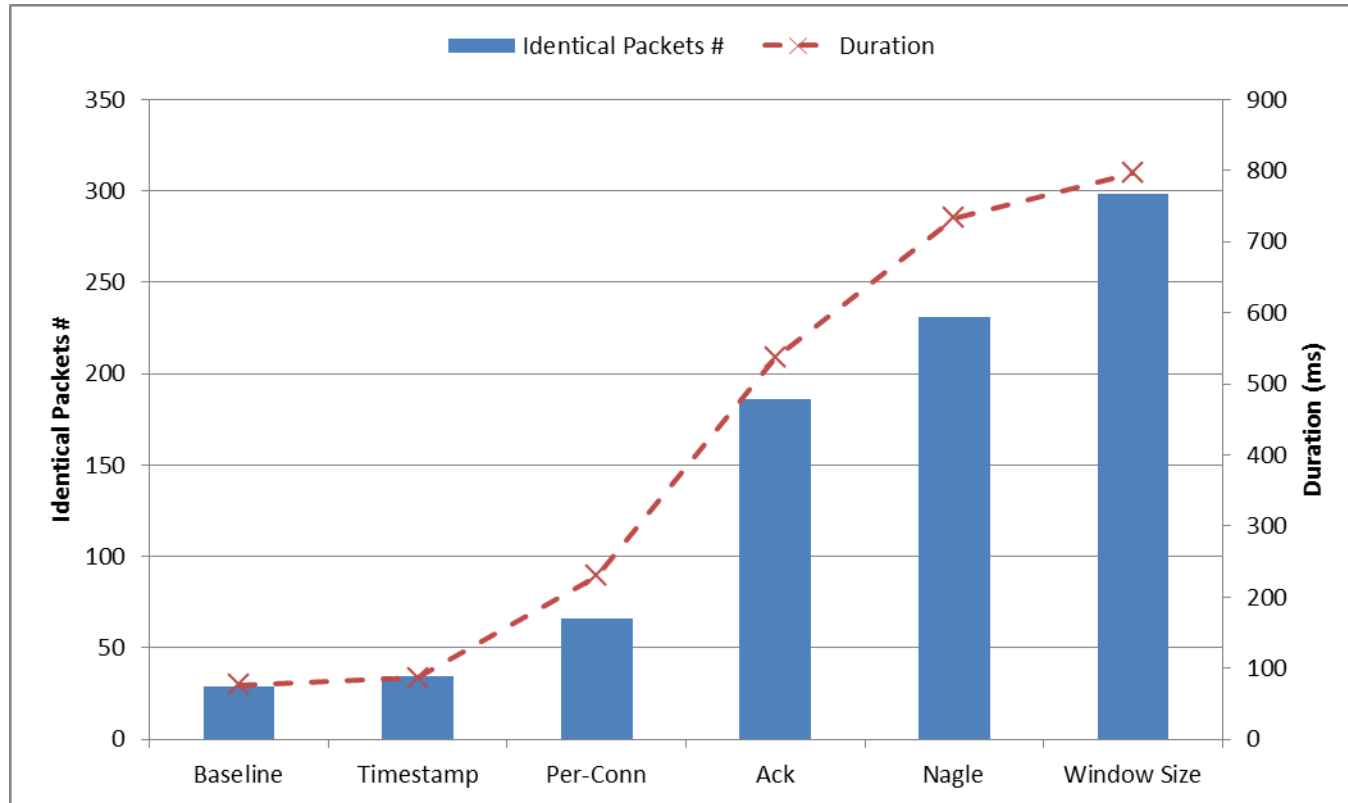
Output Similarity of Web Server – Web Bench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

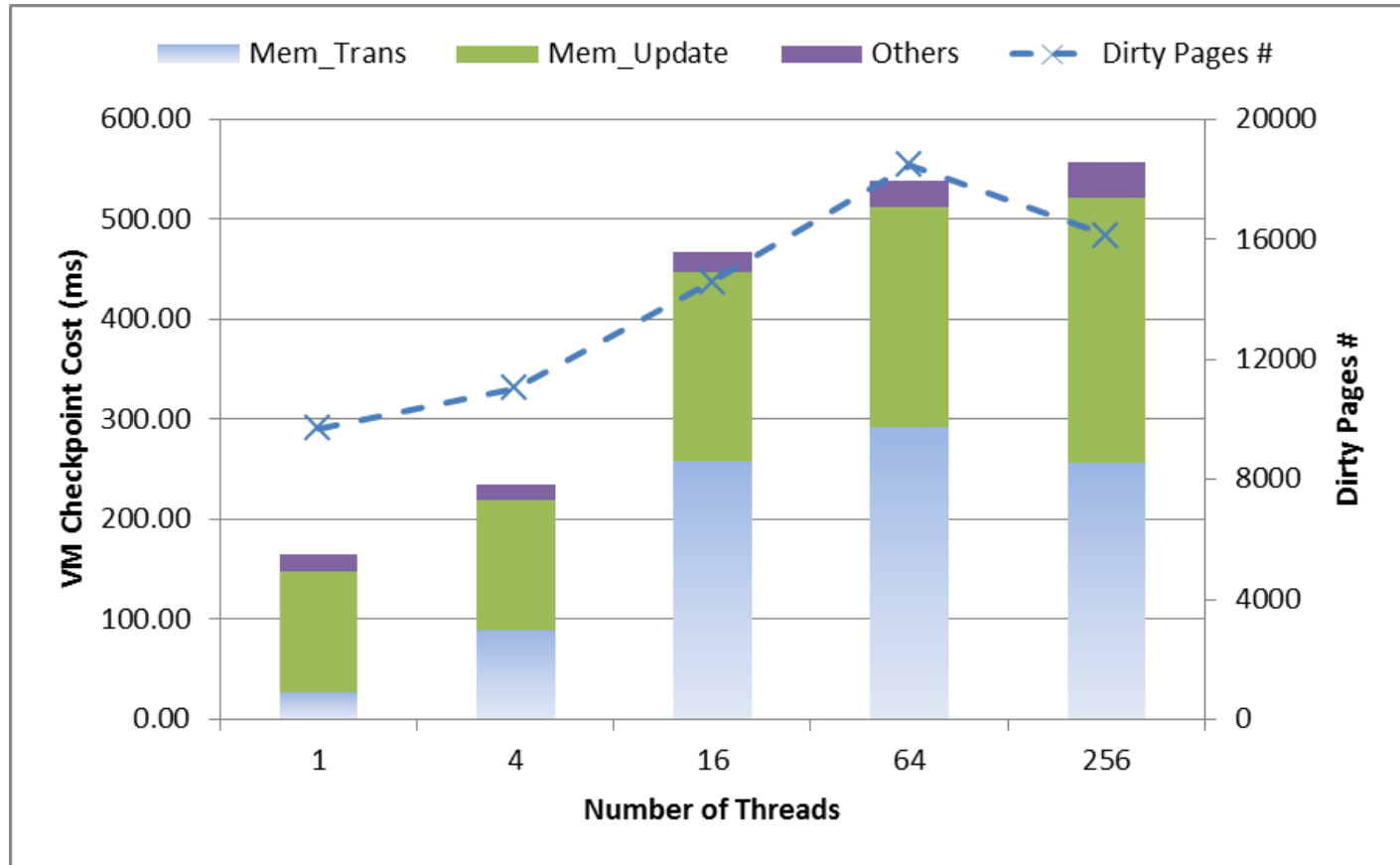
Output Similarity of PostgreSQL - Pgbench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

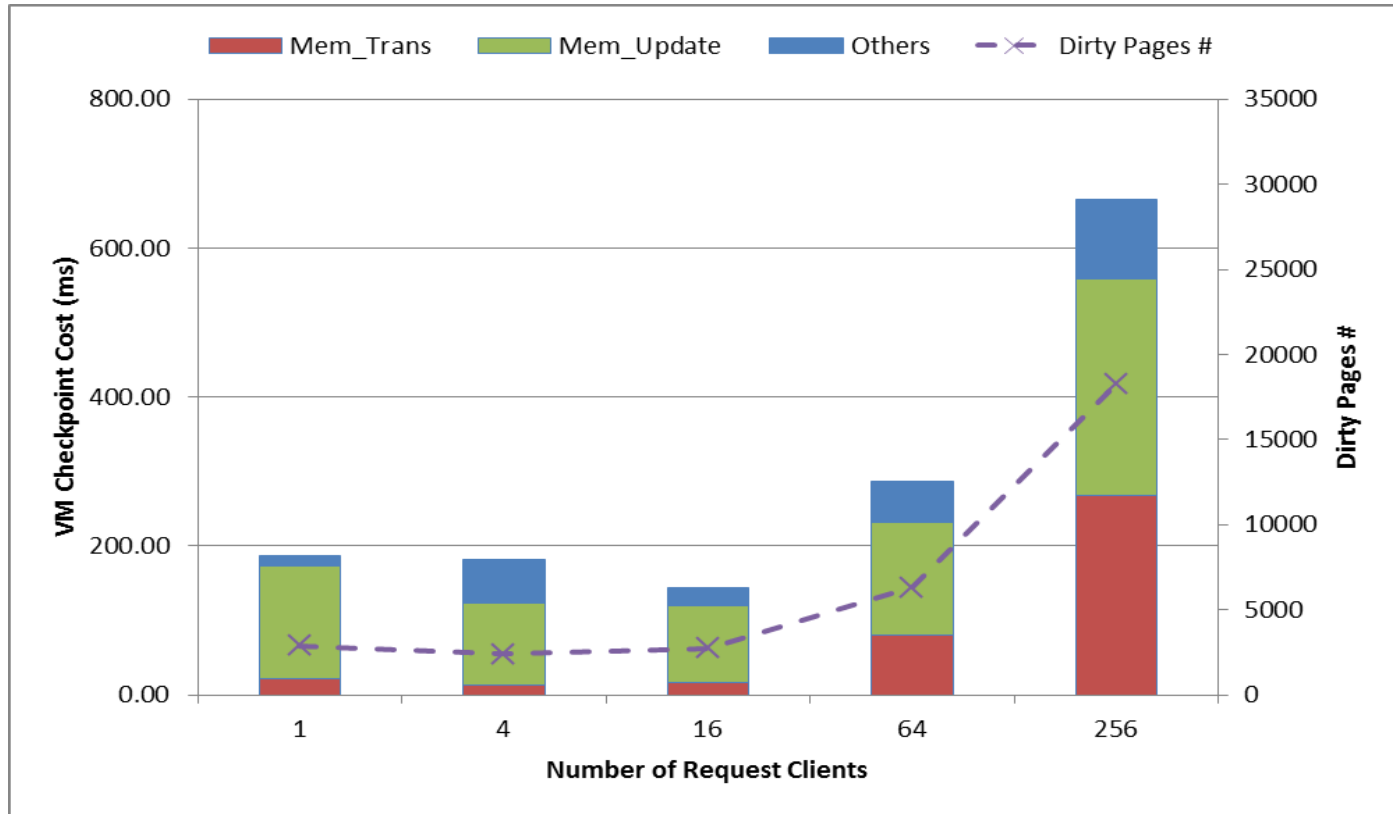
VM Checkpoint Cost – Web Bench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

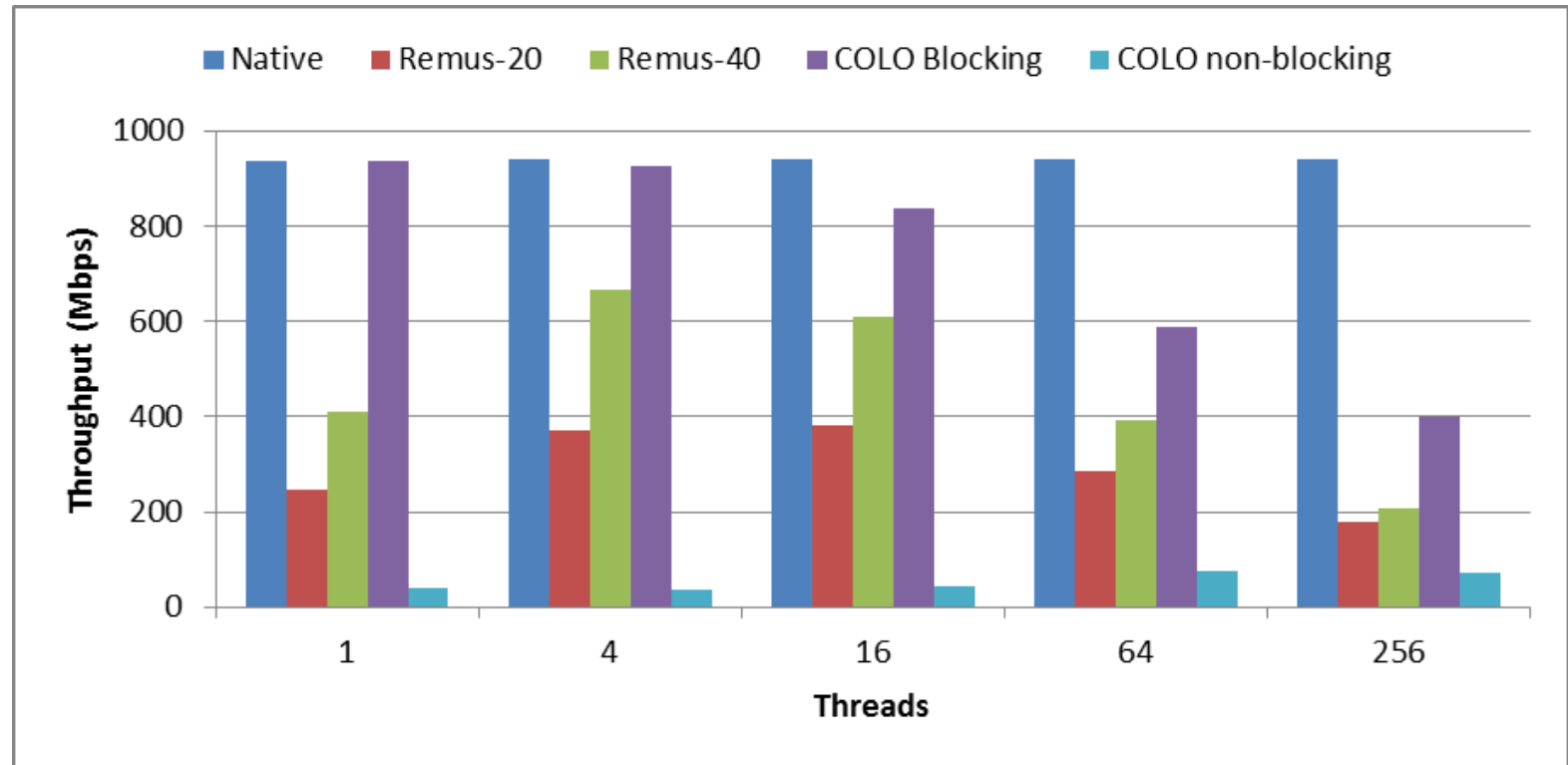
VM Checkpoint Cost – Pgbench



For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel

Limitations



Performance Degradation from nonblock Sending in Web Bench

For more complete information about performance and benchmark results, visit www.intel.com/benchmarks

Source: Intel