15-712:
Advanced Operating Systems & Distributed Systems

Virtual Machines I

Prof. Phillip Gibbons

Spring 2020, Lecture 19
“Disco: Running Commodity Operating Systems on Scalable Multiprocessors”

Edouard Bugnion, Scott Devine, Kinshuk Govil, Mendel Rosenblum 1997

- **Edouard Bugnion** (Stanford MS/PhD, EPFL)
  - Co-founded VMware 1998, Chief Architect until 2004

- **Scott Devine** (Stanford PhD, VMware)
  - Co-founded VMware, still there!

- **Kinshuk Govil** (Stanford PhD, LBL)
  - Industry: VMware (non-founder), Google, LBL

- **Mendel Rosenblum** (Stanford)
  - Co-founded VMware, Chief Scientist until 2008
  - NAE, AAAS, etc
“Disco: Running Commodity Operating Systems on Scalable Multiprocessors”

Edouard Bugnion, Scott Devine, Kinshuk Govil, Mendel Rosenblum 1997

SigOps HoF citation (2008):

None given 😞
Scalable Shared Memory Multiprocessors

Cache-coherent Non-uniform Memory Access (CC-NUMA)
Challenge: Providing System Software for Innovative Hardware

• Scalable shared memory multiprocessors of the mid-1990s

“OS developers shoulder much of the blame for the inability to deliver on the promises of these machines.”

• System software is millions of lines of code

“Late, incompatible, and possibly even buggy system software can significantly impact the success of such machines”
Architecture of DISCO

Virtual machine monitor (VMM), aka Hypervisor
Advantages of VMM Approach

• No redesign of system software to new hardware

• Can keep running conventional OS & legacy apps;
  Can stage new apps

• Can have multiple OS, including application-specialized OS

• Fine-grain resource sharing: e.g., move memory between VMs

• Dynamic load balance of virtual processors on physical machines

• Simpler VM becomes unit of scalability & of fault containment

• Can hide NUMA, via dynamic page migration/replication
Challenges Facing VMs

• **Time/Memory/Disk Overheads**
  – Interposition overheads on instructions, e.g., I/O requests
  – Slow SW emulation of privileged instructions
  – Additional memory for multiple OS, multiple apps, multiple file system buffer caches
  – Additional disk space for multiple file systems

• **Resource Management**
  – Decisions made w/o the high-level knowledge an OS would have, e.g., busy-waiting / idle loop vs. useful instructions, inactive page vs. active page

• **Communication & Sharing are difficult**
Disco’s Interface

• Processors
  – MIPS R10000
  – Load/store on special addresses for fast enable/disable of interrupts

• Physical Memory
  – Contiguous physical addr space starting at address 0
  – (Nearly) Uniform memory access

• I/O Devices
  – Disks, network interfaces, interrupt timers, clock, console
Disco’s Implementation

- **Virtual CPUs**
  - Direct execution + Interposition/Emulation

- **Virtual Physical Memory**
  - Level of indirection. Steps to minimize extra TLB miss overhead

- **NUMA Memory Management**
  - Dynamic page migration/replication to reduce NUMA impact

- **Virtual I/O Devices**
  - Intercept via special device drivers
Discussion: Summary Question #1

- **State the 3 most important things the paper says.** These could be some combination of their motivations, observations, interesting parts of the design, or clever parts of their implementation.
Copy-on-Write Disks

Not used for disks containing user files: Instead, only a single VM can mount the disk at any given time
Transparent Sharing of Pages over NFS
Performance on Simulated HW

Disco overhead is only 3%-16%
• **Describe the paper's single most glaring deficiency.** Every paper has some fault. Perhaps an experiment was poorly designed or the main idea had a narrow scope or applicability.
Breakdown by Kernel Service

Table 2. Overhead breakdown for the top kernel services of the pmake workload

<table>
<thead>
<tr>
<th>OS Service</th>
<th>Time</th>
<th>Count</th>
<th>Avg time</th>
<th>Slowdown</th>
<th>Relative Execution Time on Disco</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IRIX 5.3 – 32 bit – 4KB pages</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEMAND-ZERO</td>
<td>30%</td>
<td>4328</td>
<td>21 μs</td>
<td>1.42</td>
<td>0.43 0.21 0.16 0.47 0.16</td>
</tr>
<tr>
<td>QUICK-FAULT</td>
<td>10%</td>
<td>5745</td>
<td>5 μs</td>
<td>3.17</td>
<td>1.27 0.80 0.56 0.00 0.53</td>
</tr>
<tr>
<td>open</td>
<td>9%</td>
<td>667</td>
<td>42 μs</td>
<td>1.63</td>
<td>1.16 0.08 0.06 0.02 0.30</td>
</tr>
<tr>
<td>UTLB-MISS</td>
<td>7%</td>
<td>630 K</td>
<td>0.035 μs</td>
<td>1.35</td>
<td>0.07 1.22 0.05 0.00 0.02</td>
</tr>
<tr>
<td>write</td>
<td>6%</td>
<td>1610</td>
<td>12 μs</td>
<td>2.14</td>
<td>1.01 0.24 0.21 0.31 0.17</td>
</tr>
<tr>
<td>read</td>
<td>6%</td>
<td>733</td>
<td>23 μs</td>
<td>1.53</td>
<td>1.10 0.13 0.09 0.01 0.20</td>
</tr>
<tr>
<td>execve</td>
<td>6%</td>
<td>42</td>
<td>437 μs</td>
<td>1.60</td>
<td>0.97 0.03 0.05 0.17 0.40</td>
</tr>
<tr>
<td></td>
<td>IRIX 6.2 – 64 bit – 16KB pages</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEMAND-ZERO</td>
<td>27%</td>
<td>1134</td>
<td>57 μs</td>
<td>1.01</td>
<td>0.17 0.05 0.10 0.68 0.01</td>
</tr>
<tr>
<td>execve</td>
<td>16%</td>
<td>42</td>
<td>608 μs</td>
<td>1.30</td>
<td>0.81 0.01 0.03 0.33 0.11</td>
</tr>
<tr>
<td>open</td>
<td>10%</td>
<td>667</td>
<td>37 μs</td>
<td>1.37</td>
<td>1.17 0.00 0.08 0.02 0.11</td>
</tr>
<tr>
<td>read</td>
<td>6%</td>
<td>733</td>
<td>21 μs</td>
<td>1.32</td>
<td>1.13 0.00 0.12 0.00 0.07</td>
</tr>
<tr>
<td>write</td>
<td>6%</td>
<td>1640</td>
<td>9 μs</td>
<td>1.71</td>
<td>0.98 0.00 0.31 0.35 0.06</td>
</tr>
<tr>
<td>COPY-ON-WRITE</td>
<td>5%</td>
<td>120</td>
<td>94 μs</td>
<td>1.16</td>
<td>0.59 0.03 0.06 0.41 0.06</td>
</tr>
<tr>
<td>QUICK-FAULT</td>
<td>5%</td>
<td>2196</td>
<td>5 μs</td>
<td>2.83</td>
<td>1.28 0.49 0.89 0.00 0.16</td>
</tr>
</tbody>
</table>

Kernel+Disco is 2.15x slower on IRIX 5.3, 1.62x slower on IRIX 6.2
Data Sharing between VMs

Significant savings from V (no sharing) to M (Disco’s sharing)
Workload Scalability

Disco scales well. Enabling specialized OS is important.
Benefits of Page Migration/Replication

UMA = IRIX w/ all local memory

Disco improves local hits by 62%-70%, performance within 26%-40% of UMA

stall times

% of cache misses satisfied locally
Performance on Real Hardware

Table 3. Origin200 Execution Time

<table>
<thead>
<tr>
<th>Breakdown</th>
<th>Pmake</th>
<th>Engineering</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IRIX (sec.)</td>
<td>Disco (sec.)</td>
<td>Ratio</td>
</tr>
<tr>
<td>User</td>
<td>11.3</td>
<td>11.7</td>
<td>1.03</td>
</tr>
<tr>
<td>Kernel</td>
<td>5.9</td>
<td>9.6</td>
<td>1.62</td>
</tr>
<tr>
<td>Idle</td>
<td>13.1</td>
<td>11.4</td>
<td>0.87</td>
</tr>
<tr>
<td>Total</td>
<td>30.3</td>
<td>32.7</td>
<td>1.08</td>
</tr>
<tr>
<td></td>
<td>IRIX (sec.)</td>
<td>Disco (sec.)</td>
<td>Ratio</td>
</tr>
<tr>
<td></td>
<td>65.2</td>
<td>69.7</td>
<td>1.07</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>0.2</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Discos overhead is only 7%-8%
Monday’s Class

Kernels & Virtual Machines (V)

“Memory Resource Management in VMware ESX Server”

Carl A. Waldspurger 2002