A Distributed Device Paradigm for Commodity Applications

Enrique Cauich, John Duselis, Richert Wang and Isaac Scherson
Dept. of Computer Science – Systems

Donald Bren School of Information and Computer Sciences
University of California, Irvine
Irvine, CA 92697
Observation (Motivation)

In most modern (corporate) networked system installations, a large number of computers are underutilized.

Powerful machines are devoted to little work and/or a few non-demanding tasks, while other processing computers (used for data analysis) may require more resources than normally implemented.
Remote high performance components/devices can be aggregated and presented to a system as local virtual devices.

These virtual devices can deliver better performance than local physical devices by using the network links as a virtual system bus.

Applications using virtual devices do not require any code modifications.
Novel Paradigm’s Approach

Conventional Library Level Implementation (e.g. PVM)

Device Level Aggregation/Driver Level Implementation
Device Level Aggregation/Driver Level Implementation
Resource Virtualization and Aggregation

• Resource Virtualization is divided in two phases
  – Lender side:
    • Implements a Component Abstraction Layer (CAL).
    • Provides basic Building Blocks
  – Borrower side:
    • Implements a Device Abstraction Layer (DAL).
    • Creates a specific virtual device satisfying the needs of the application using the Building Blocks

• Resource Aggregation is done at the borrower side
  – Resource Discovery
  – Resource Management
  – Resource Consolidator
Virtualization Paradigm

Virtual Device

- Driver Interface
- Consolidator
- Resource Discovery
- Resource Manager

Virtual Components

- Virtualizer
- Software
- Hardware

Borrower Virtualization

Aggregation

Lender Virtualization
Virtual Device Architecture

- Exploits parallelism
- Avoids Data hazards
- Maintains coherency
Two Experimental Case Studies

• High Performance Distributed RamDisk

  Improves I/O intensive (Disk) applications’ performance by using idle semiconductor memory resources in the network.

• Distributed Graphics Device

  A Codec device that enhances the system with ray tracing capabilities by adding virtualized idle processor and graphics devices.
Distributed RamDisk

- Harnessing available resources from under-utilized systems in the network.
- Memory is an important resource.
- Generally used as temporary storage to speed up memory references.

Improves I/O intensive (Disk) applications performance by using idle memory resources in the network.

Storage bandwidth is increased and storage latency is decreased.
Distributed RamDisk Read Throughput in MB/s
Distributed RamDisk Write Throughput in MB/s
Distributed RamDisk Comparison using 8 outstanding requests
Accordion Effect

TCP Send Buffer

Repeated TCP Segments ACKs

TCP Segment Retransmission

Retransmitted TCP Segments ACK

Next TCP Segment

Resource Lender

Resource Borrower

TCP Receive Buffer

TCP Receive Buffer
TCP receive window size

Performance Correlation in a 48KB size Window and 4KB request size

- Throughput in MB/s
- Latency in centiseconds
- Retransmission percentage in hundredths of a percent

Chart showing the correlation between the number of outstanding requests and performance metrics.
TCP receive window size

Performance Correlation in a 40KB size Window and 64KB request size

Graph showing performance correlation with number of outstanding requests:
- Throughput in MB/s
- Latency in centiseconds
- Retransmission percentage in hundredths of a percent
TCP receive window size (Cont.)

Distributed RamDisk throughput while using 1, 2 and 3 of lenders

Throughput in MB/s

1 KB 2 KB 4 KB 8 KB 16 KB 32 KB 64 KB

8

1 KB 2 KB 4 KB 8 KB 16 KB 32 KB 64 KB

16

1 KB 2 KB 4 KB 8 KB 16 KB 32 KB 64 KB

32

1 Memory Lender
64 KB TCP RcvWnd

2 Memory Lenders
32 KB TCP RcvWnd

3 Memory Lenders
24 KB TCP RcvWnd
Database Applications’ Performance Improvement

Given:

- a distributed RamDisk 1 borrower/3 lenders configuration interconnected by a Gigabit switch
- SQLServer 2008
- TPC-H benchmark

Conjecture: a database application can improve its performance by storing index files in a distributed RamDisk. The database does not require any code modification.
Database Performance Improvement

Time spent by the database performing the index search

Request Size:

<table>
<thead>
<tr>
<th>Size</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 KB</td>
<td>35</td>
</tr>
<tr>
<td>128 KB</td>
<td>17</td>
</tr>
<tr>
<td>192 KB</td>
<td>13</td>
</tr>
<tr>
<td>256 KB</td>
<td>17</td>
</tr>
<tr>
<td>512 KB</td>
<td>10</td>
</tr>
</tbody>
</table>
Image Processing Applications’ Performance Improvement

Given:

• a distributed RamDisk 1 borrower/3 lenders configuration interconnected by a Gigabit switch
• Adobe Photoshop CS2

Conjecture: A image processing application can improve its performance by using the distributed RamDisk as scratch space.
Image Processing Performance Improvement

Gaussian Filter Execution Time while Using a Distributed RamDisk and WD10 disk

Seconds

<table>
<thead>
<tr>
<th>Images</th>
<th>1 Pixel</th>
<th>5 Pixels</th>
<th>250 Pixels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image 4</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

WD10
Distributed RamDisk
Second Case Study: A Distributed Codec Device

![Diagram of a distributed codec device system](image)

- **Win32 Application**
- **Windows Image Component**
- **Codec Device**
- **High Speed Network**
- **Graphics Hardware/Software**
- **Virtual HAL**
- **CPU**
- **Graphics Card**
Second Case Study: A Distributed Codec Device

• Consolidator partitions the request.

• Resource manager calculates the suitability of each virtual component
Codec Results

Image decodification time using different number of lenders and different load distribution mechanisms

- **Intelligent Distribution**
- **Naïve Distribution**
- **Borrower/Lender Homogeneous**
Conclusions

• A new level of abstraction, namely the Device Level, and a new level of implementation, namely the Driver Level, are added to the virtualization taxonomy.

• The Device Abstraction Level increases system performance capacity by aggregating idle networked resources and binding them as a single device.

• The Driver Implementation Level provides support for legacy applications by avoiding code modifications.