Performance Evaluation of Load Sharing Policies with PANTS on a Beowulf Cluster

James Nichols
Mark Claypool

Worcester Polytechnic Institute
Department of Computer Science
Worcester, MA

http://www.cs.wpi.edu/~jnick
http://perform.wpi.edu
Introduction

What is a Beowulf cluster?
- Cluster of inexpensive personal computers networked together via Ethernet
- Typically run the Linux operating system

Load Sharing
- Share load, decreasing response times and increasing overall throughput
- Need for expertise in a particular load distribution mechanism such as PVM or MPI
Introduction

- Load Measurement
  - Typically use CPU as the load metric.
  - What about disk and memory load? Or system events like interrupts and context switches?

- PANTS Application Node Transparency System
  - Removes the need for knowledge about a particular implementation required by some load distribution mechanisms
Contributions

- Propose new load metrics
- Design benchmarks
- Evaluate performance
- There is some benefit to incorporating new types of load metrics into load distributions systems, like PANTS
Outline

- Introduction
- PANTS
- Methodology
- Results
- Conclusions
PANTS

- PANTS Application Node Transparency System
  - Intercepts `exec()` system calls
  - By default uses `/proc` file system to calculate CPU load to classify node as “busy” or “free”

- Any workload which does not generate CPU load will not be distributed

- New load metrics and polices!

  - Early results showed near linear speedup for computationally intensive applications
PANTS Algorithm

- We have implemented a variation of the multi-leader load-balancing algorithm proposed in [FW95]
- A node is elected to be the leader
- Leader keeps track of which machines in the cluster are free
PANTS Algorithm

- A random free node is chosen by the leader and returned to a node upon request.
- Some algorithms use broadcast messages to find free nodes and communicate.
- Busy nodes need to receive and process all of the messages.
- PANTS avoids “busy-machine messages” by sending messages only to the leader multicast address.
PANTS Multicast Communication

Diagram:

- Leader Multicast Address
- A
- B
- Leader
- C
- Free Node Multicast Address
PANTS Implementation

- Two major software components: PANTS daemon and \texttt{prex} (PANTS remote \texttt{execute})
- C-library object intercepts \texttt{execve} for processing by \texttt{prex}
- \texttt{prex} queries the PANTS daemon for a node to execute the process on, the daemon handles load measurement and leader communication
- RSH is used by \texttt{prex} to execute process on remote nodes
Extensions to PANTS

- PANTS is compatible with the distributed interprocess communications package DIPC
- DIPC requires some code modifications, but provides a wide range of IPC primitives

- Modify PANTS daemon configuration by altering `/etc/pantsd.conf`
- Send Unix signals to use new configuration
  - Easily modify thresholds, exponential weighted averaging settings, multicast addresses
  - Wide range of logging options
- New load metrics and policies
WPI’s Beowulf Cluster

- Made possible through equipment grants from Compaq and Alpha Processor, Inc
- Seven 600mhz Alpha machines (EV56)
- Physical memory from 64-512MB
- 128 MB swap space
- PCI Ultra-Wide SCSI hard drives
- 100BaseT Ethernet
- RedHat 7.1, Linux kernel 2.4.18
- Shares files over NFS
Performance Evaluation of Load Sharing Policies with PANTS on a Beowulf Cluster

- Introduction
- PANTS
- Methodology
- Results
- Conclusions
Methodology

- Identified load parameters
- Implemented ways to measure parameters
- Built micro benchmarks which stressed each load metric for testing and verification
- Selected real world benchmark to evaluate performance
## Load Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>usage (%)</td>
</tr>
<tr>
<td>I/O</td>
<td>blocks/sec</td>
</tr>
<tr>
<td>Context switches</td>
<td>switches/sec</td>
</tr>
<tr>
<td>Memory</td>
<td>page operations/sec</td>
</tr>
<tr>
<td>Interrupts</td>
<td>Interrupts/sec</td>
</tr>
</tbody>
</table>

Read from `/proc/stat`
Micro-Benchmarks

- Set of simple benchmarks designed to generate a certain type of workload
- Verification of our load metrics
- Determination of thresholds

- CPU: perform many FLOPS
- I/O: copy large directory and files
- Memory: malloc() a block of memory, copy data structures using mmap()
Application benchmark: Linux kernel compile

- Distributed compilation of the Linux kernel
- Executed by the standard GNU program `make`
- Loads I/O and memory resources
Application Benchmark: Details

- Linux kernel version 2.4.18
- 432 files compiled
- Mean source file size: 19 KB

- Marked \texttt{gcc} compiler binaries migrateable
- Needed to expand relative paths into absolute paths
Application Benchmark

CPU Usage - PANTS Default
Thresholds

- Obtained idle measurements
- Iteratively established thresholds

<table>
<thead>
<tr>
<th>Metric</th>
<th>Idle</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU (%)</td>
<td>0%</td>
<td>95%</td>
</tr>
<tr>
<td>I/O (blocks/sec)</td>
<td>250</td>
<td>1,000</td>
</tr>
<tr>
<td>Context switches (switches/sec)</td>
<td>950</td>
<td>6,000</td>
</tr>
<tr>
<td>Memory (pages/sec)</td>
<td>0</td>
<td>4,000</td>
</tr>
<tr>
<td>Interrupts (interrupts/sec)</td>
<td>103K</td>
<td>115K</td>
</tr>
</tbody>
</table>
Performance Evaluation of Load Sharing Policies with PANTS on a Beowulf Cluster

- Introduction
- PANTS
- Methodology
- Results
- Conclusions
Micro-benchmarks Results

Default Load Metrics

I/O micro benchmark - average load

- CPU (%)
- I/O (100's Disk blocks/sec)
- Memory (100's page faults/sec)
- Interrupts (1000's interrupts/sec)
- Context Switches (100's of switches/sec)
Micro-benchmarks Results

New Load Metrics

I/O micro benchmark - average load

- CPU (%)
- I/O (100's Disk blocks/sec)
- Memory (100's page faults/sec)
- Interrupts (1000's interrupts/sec)
- Context Switches (100's of switches/sec)
Application Benchmark Results

![Bar chart showing CPU metrics]

- Average
- Std Dev
- Max
- Min
- All Metrics

CPU %
## I/O Load

<table>
<thead>
<tr>
<th>Metric</th>
<th>Average</th>
<th>Std Dev</th>
<th>Max</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk B/s</td>
<td>0</td>
<td>50000</td>
<td>100000</td>
<td>150000</td>
</tr>
<tr>
<td></td>
<td>200000</td>
<td>250000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Results: Compile Time
Conclusions

- PANTS has several attractive features:
  - Transparency
  - Reduced busy node communication
  - Fault tolerance
  - Intelligent load distribution decisions

- Achieve better throughput and more balanced load distribution when metrics include I/O, memory, interrupts, and context switches.
Future Work

- Use preemptive migration?
- Include network usage load metric
- Adaptive thresholds
- Heuristic based load distribution
- Migrate certain types of jobs to nodes that perform well when processing certain types of workloads
Questions?

Acknowledgements

Jeffrey Moyer, Kevin Dickson, Chuck Homic, Bryan Villamin, Michael Szelag, David Terry, Jennifer Waite, Seth Chandler, David Finkel, Alpha Processor, Inc. and Compaq Computer Corporation
Performance Evaluation of Load Sharing Policies with PANTS on a Beowulf Cluster

James Nichols
Mark Claypool

Worcester Polytechnic Institute
Department of Computer Science
Worcester, MA

http://www.cs.wpi.edu/~jnick
http://perform.wpi.edu