Super-Dense Servers:
An Energy-efficient Approach to Large-scale Server Clusters

Charles Lefurgy

IBM Research, Austin

Outline

- **Problem**
  - Internet data centers use a lot of energy
- **Opportunity**
  - Load-varying applications
  - Servers can be power-managed
- **Solution**
  - Hardware: Dense server blades
    - Design decisions
    - Software support
  - Software: Power-Aware Request Distribution
    - Framework for cluster energy studies
    - Adapt cluster resources to workload

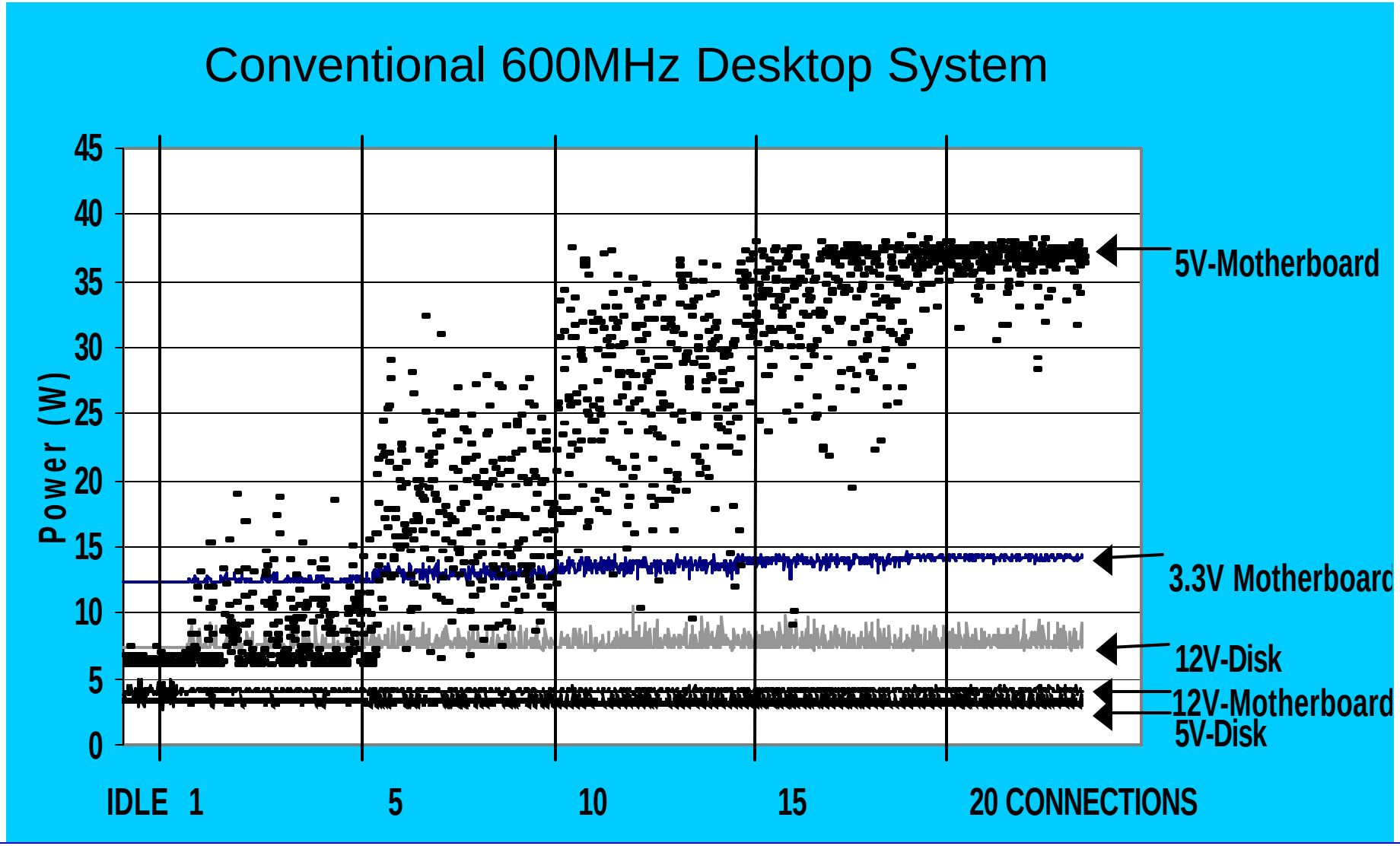
Motivation

- **Internet Data Centers**
  - 25% of operation cost are for energy and cooling
  - Anecdotal evidence that customer's racks are power-limited
  - Source: Jennifer Mitchell-Jackson's thesis at http://enduse.lbl.gov/Info/datacenterreport.pdf
    See also http://www.repp.org/articles/static/1/binaries/data_centers_report.pdf
- **Power consumption affects cooling and backup power generation requirements, as well as reliability**
  - Higher power means greater investments in sophisticated racks, air conditioning and power generation infrastructure
  - Excessive heat may cause intermittent failures
  - Power draw may become problem for utilities
- **Mobile servers, e.g. aircraft, ships, military applications**

Web server energy

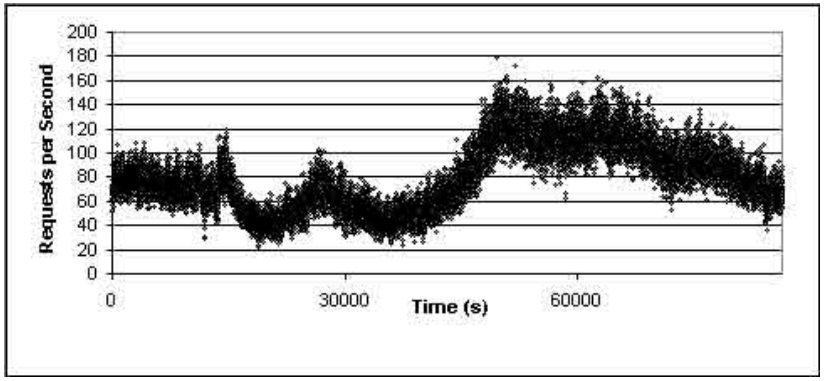
- **Sites are built with extra capacity**
  - To handle load spikes
  - To handle failures
- **This causes clusters to be underutilized**
  - Nagano 1998 Winter Olympics
    - Average load was 25% of peak encountered
  - Wimbledon 1999
    - Average load was 11% of peak encountered
- **Workload varies dramatically**
  - Time of day
  - Time of year
  - Application type
- **This is an opportunity for power management!**

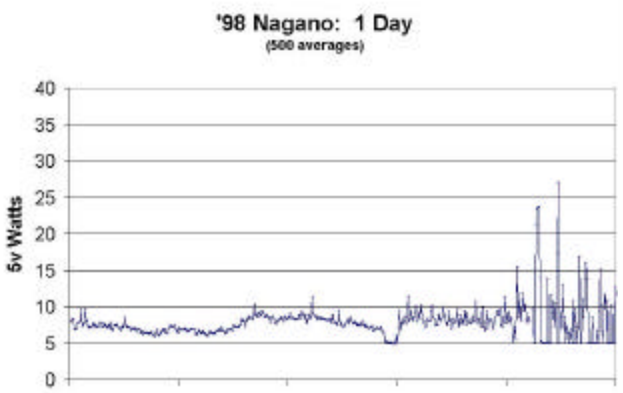
Where does the energy go?

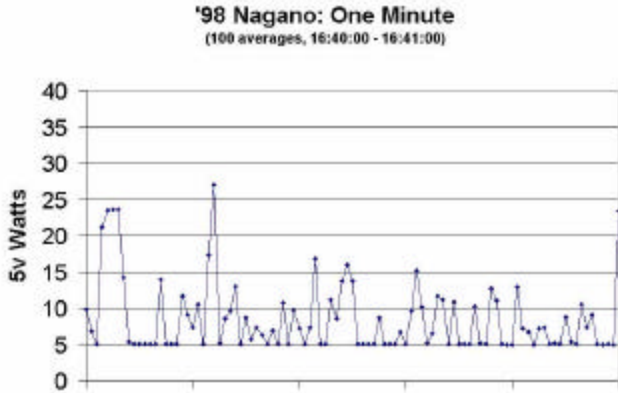


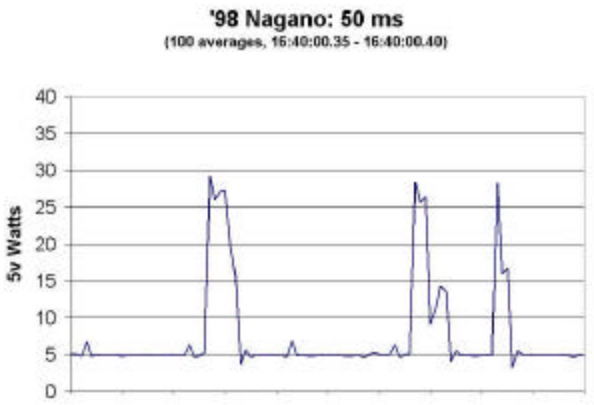
When can power be saved?

- **1998 Nagano Winter Olympics**



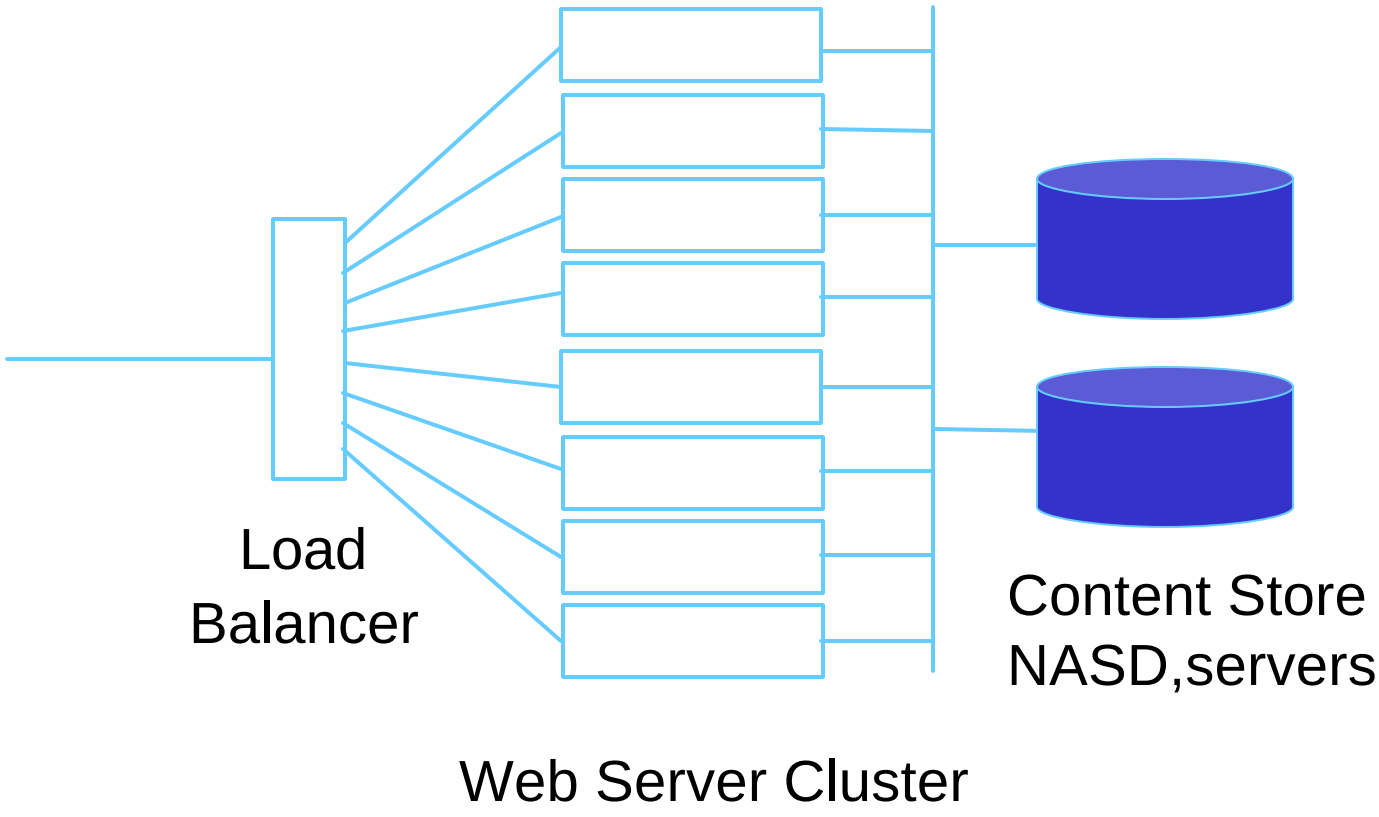






Server cluster

Set of computers used as a single system

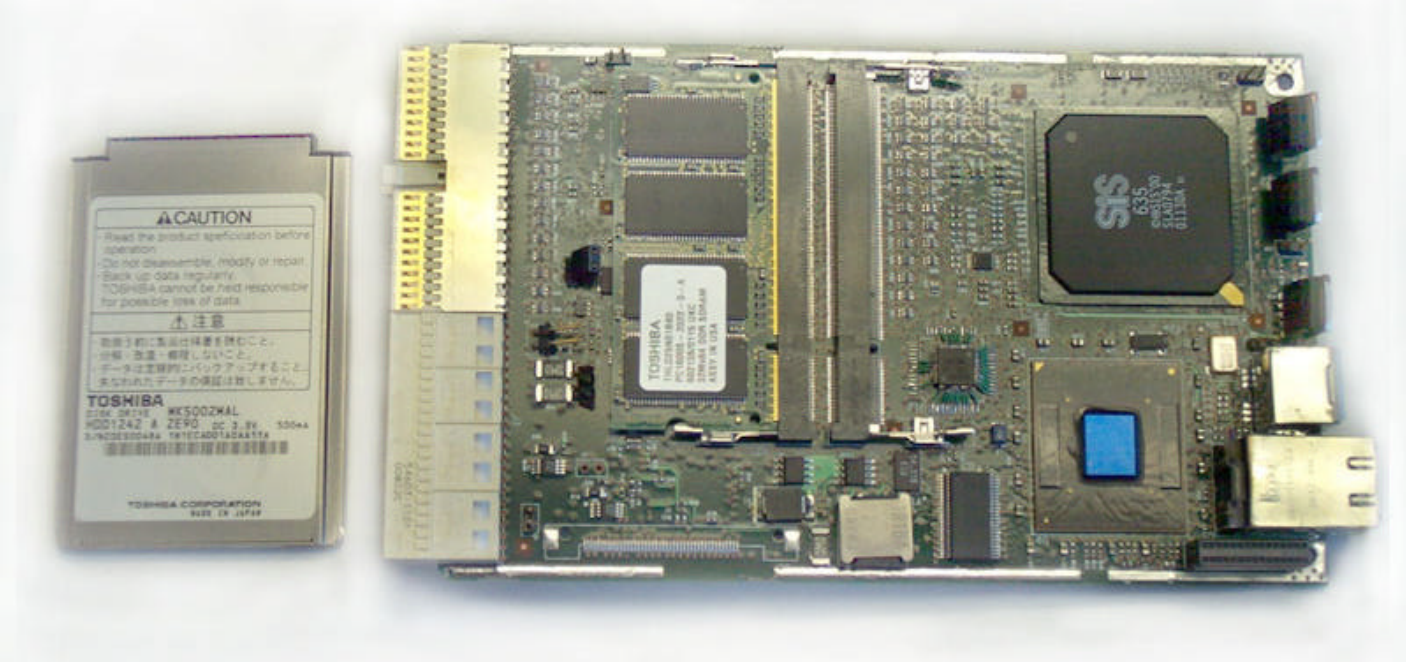


The SDS hypothesis

- **Use embedded processors for general purpose servers**
  - Army of turtles approach
  - Take advantage of $P \propto cfv^2$
  - Embedded processors use less speculation. Waste less energy.
  - Low-power + high integration = high density
  - Focus on MIPS / m^3 / Watt
- **Use blade form factor for high density**
- **Use blades in tier 1 and tier 2 of web site**
  - Parallelism in requests is a match for having many slow blades
  - Tier 3 (database) has too much synchronization and works better on symmetric multiprocessors.

SDS blade

- **1 x86 ULV SpeedStep 500/300 MHz**
- **512 MB SODIMM (256 MB with disk)**
- **2 100-Mb Ethernet ports**
- **1 Toshiba 1.8” IDE 5GB HDD**
- **No keyboard, video, mouse**



Blade Power Budget

Worst case power (Watts)

Processor	**6.402**
SODIMM 256MB	**1.000**
Voltage Regulator	**0.005**
North/South/Ethernet	**1.980**
Ethernet PHY	**0.660**
LPC Flash Memory	**0.033**
EEPROM	**0.007**
PCI to PCI Bridge	**0.173**
Supervisory Processor	**0.330**
Ethernet Controller	**0.743**
Voltage Monitor - I2C	**0.008**
Clock Generator	**0.693**
Disk	**1.485**
90% efficient power supply	**1.352**
Total	**14.871**

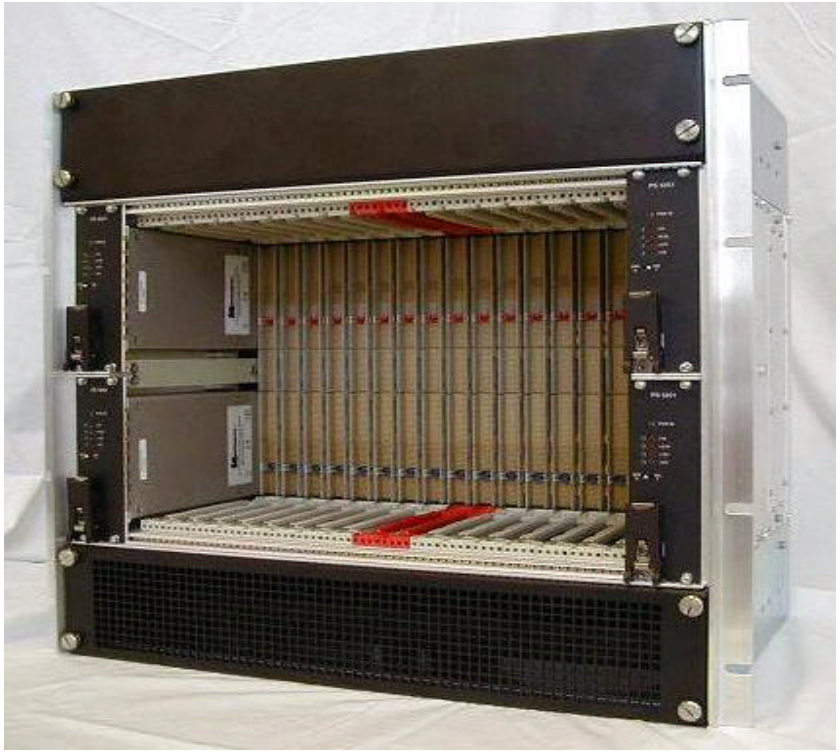
Bladed Servers

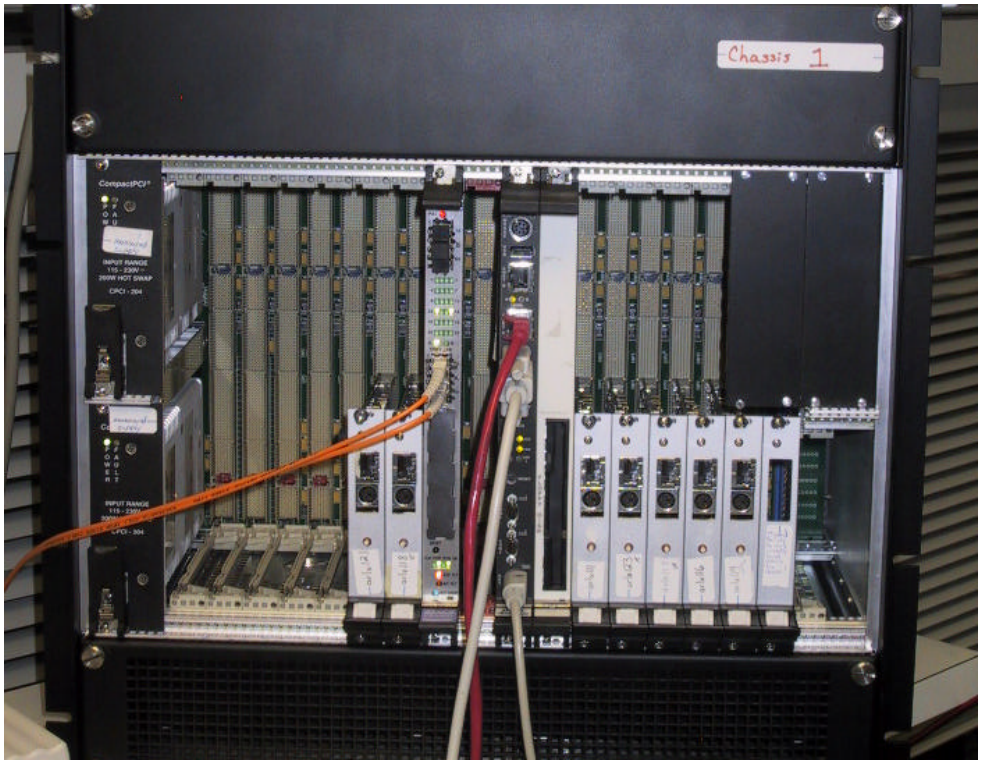
- **Blade: Board that plugs into chassis (backplane)**
- **Advantages**
  - Less cabling (this is important!)
  - Potentially lower space, power, cooling needs
  - Mix and match: server, storage, network, etc.
  - Blade cluster offers finer power management and system control

- **Disadvantage**
  - Current data centers may not be able to cope with higher energy density

Blade enclosure

- **Industry standard CompactPCI enclosure. 6U high.**
- **Network blade: network switching**
- **Server blade: processor + memory**
- **System management blade with disk**





Rack comparison

	SDS Cluster	Conventional
CPUs	360	42
CPUs/U	8.57	1
Processor speed	180 GHz (x-86) (500 MHz each)	101 GHz (x-86) (2.4 GHZ each)
Main memory	184 GB	168 GB
Ethernet	71.4 Gb/s	84 Gb/s
L2 cache	92 MB	42 MB
I/O buses	360	42

Software for SDS

- **Linux Diskless Server Architecture**
  - Single system image for all blades
  - Boot from management blade disk
  - Blades are diskless and boot in 20 seconds
- **Ethernet block device**
  - High performance swap
  - Serving web content
- **Blade management across I2C bus**
  - H8 microcontroller on blades acts as power switch
- **Console over Ethernet**
- **Power-Aware Request Distribution**
  - Quick boot time reduces “idle” power

Evaluation of SDS Cluster

	8 blade SDS cluster	IBM x330
CPUs	8	1
Processor speed	300 MHz each	1.2 GHz
Main memory	256 MB each	2 GB
Ethernet	100 Mb/s each	1 Gb/s
L2 cache	512 KB each	512 KB

- IBM x330 is what was available when SDS was designed
- Use same total memory
- Conservative cluster configuration in other aspects.
- Blades could only use 300 MHz and 256 MB at time of evaluation
- Use a modified TPC-W benchmark (fit images in memory of x330)

<tpc-w> results

	8 blade SDS cluster	IBM x330
WIPS	**117**	**68**
Power	**104.1 W**	**101.7 W**
WIPs/Watt	**1.12**	**0.67**

- **Benchmark is CPU bound**
- **Blades provide 1.7x performance for similar power level**
- **2x CPU frequency of blades helps them win**

- **This is a conservative result**
  - Fixing NIC interrupts, using 500 MHz, and using 512 MB would improve energy-efficiency of blades

Summary of SDS blades

- **Lesson: blades are viable deployment alternative for edge and application servers**
- **1.7x better performance for CPU-bound <tpc-w> workload at same energy cost**
- **Performance is worse for applications in which blades "band" together to provide a single cluster image**
  - SpecWeb99 requires a lot of memory
  - Blades each have less memory and they cannot share their memories
  - Traditional SMP server are better here
- **Heterogeneous deployments are required until memory density improves**

PARD

Power-Aware Request Distribution

PARD

- **PARD: a method of scheduling requests among servers so that energy consumption is minimized while maintaining a particular level of performance**

- **Goals**
  - Save energy in a web cluster
  - Do not impact response latency

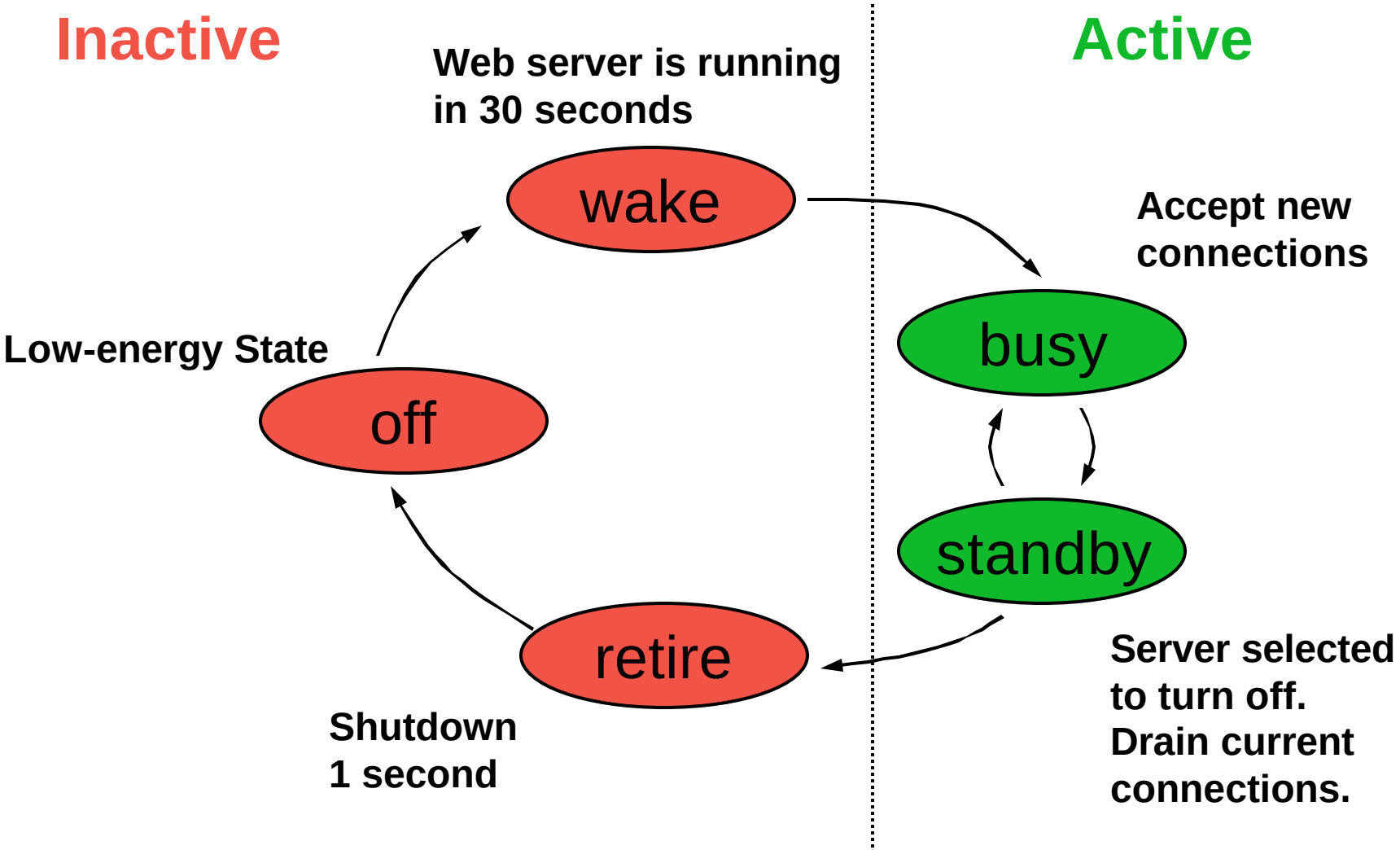
- **Solution: consolidate work onto fewer servers**
  - Turn off inactive servers
    - Idle servers use a lot of energy
  - Cost: small increase in response latency

- **Mechanism**
  - Monitor the cluster
  - Use load balancing

Pitfalls in measuring energy for clusters

- **Not measuring total system energy**
  - Slowing down the system to save processor power is not useful if other components are on longer and use more idle energy
- **Not scaling workload to the system**
  - Improving an inefficient machine overstates the results!
- **Poor metrics**
  - Running the benchmark again and reporting the energy savings is not enough. How was performance impacted?
  - Ignoring response time in web benchmarks
- **Poor benchmarks**
  - Cluster workloads for energy-efficiency have used manufactured benchmarks
- **No idea if results are "good enough". What is the limit of the method being evaluated?**

Life of a blade web server



4 dimensions of problem

- **Energy savings**
- **Quality of service (performance)**
- **System characteristics**
- **Workload characteristics**

- **When any two are fixed, there is a trade-off between the other two**
- **All must be reported to understand results**
  - Often, only the first two are used

System Characteristics

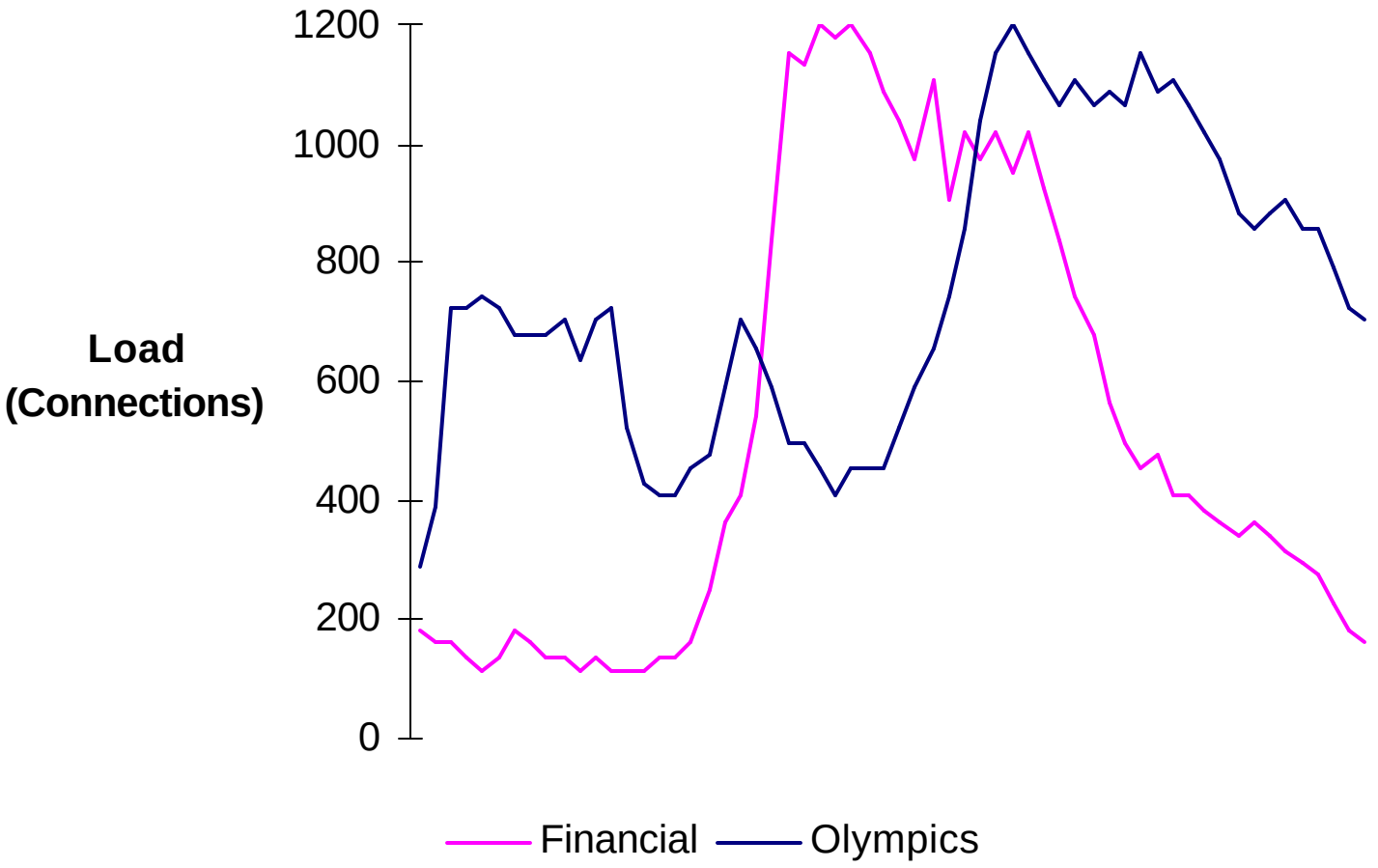
- **Cluster unit**
  - For example, a complete server
- **Immunity to overload**
  - At what point does a server overload and die?
- **System energy consumption**
  - Idle power, peak power
- **Startup and shutdown delay of cluster units**
  - Is this unit used by other units?
- **Ability to migrate requests**
  - Free up servers to turn them off

Workload characteristics

- **Workload unit**
  - For example, a connection. Differs by connection type.
- **Load profile**
  - The instantaneous load and a required minimum QoS.
  - “machine utilization ratio” corresponds to “load ratio”
- **Rate of change in workload**
  - Can machine respond to spikes quickly enough?

Workloads

- **Adapt TPC-W to fit load-varying web trace**



Test environment

- **Web cluster**
  - 8 SDS blades running Apache
  - Linux Virtual Server (LVS) does request distribution
  - <tpc-w> benchmark

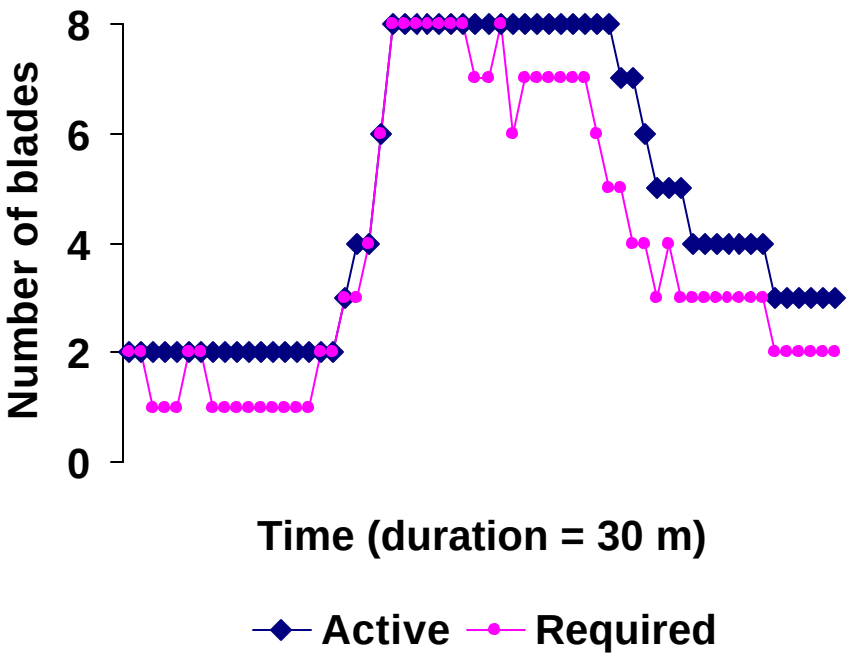
- **Energy monitoring**
  - National Instruments data acquisition equipment and Lab View
  - Send data to LVS director
  - Measure “wall power” of cluster

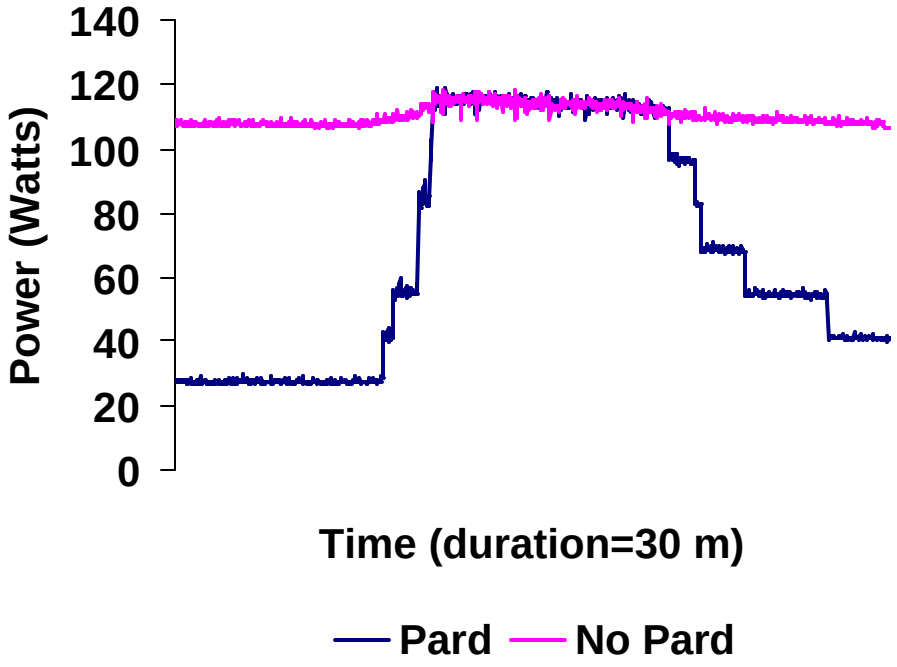
- **Cluster utilization**
  - Each server periodically collects statistics
  - /proc/stat, netstat: CPU, disk, network stats
  - Send data to LVS director

- **Linux Virtual Server**
  - Modified to use monitoring to drive request distribution policies
  - Use “Least Connections” scheduling policy

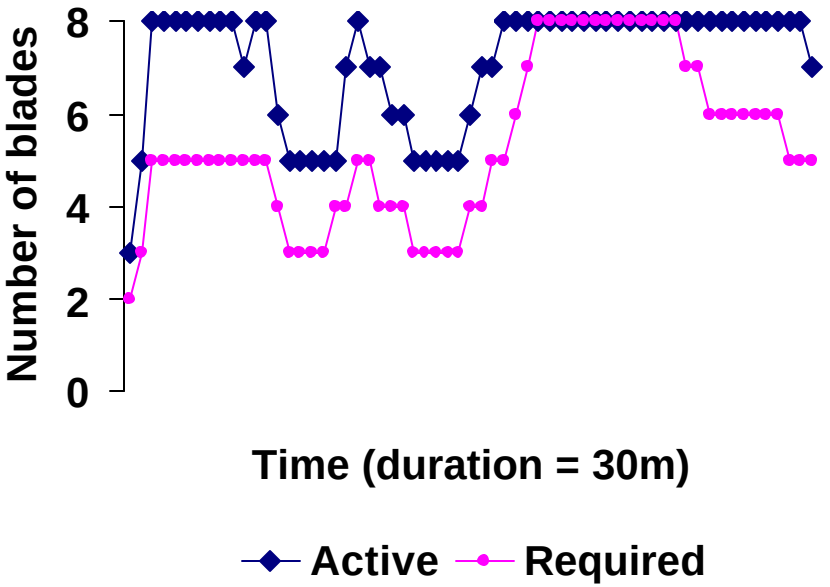
Simple threshold model

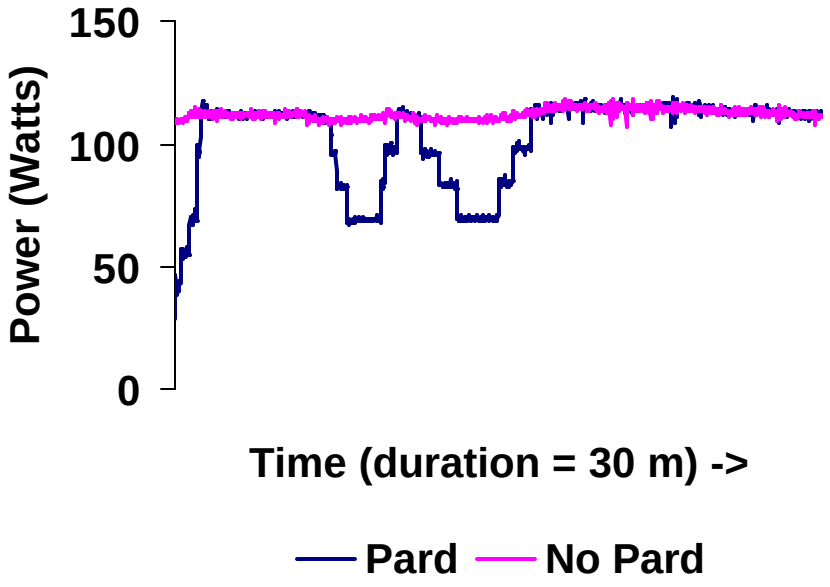
- **Given a workload, always turn on enough servers so that the performance goal is met**
  - Assign a threshold to each machine for # connections it can hold before the next server is turned on. Based on maximum load acceleration.
  - Turn on another machine when load balancing can no longer be used to avoid putting a machine over its threshold
  - Use future knowledge to set threshold. This is to estimate the limit of technique.





Financial workload





- **Maximum rate of change in trace is higher than Olympics**
  - Use a lower threshold to turn blades on earlier
- **When rate of change is lower (other parts of trace)**
  - Blades turn on earlier than they really should
  - This is why “active” is much higher than “required” for most of trace
  - We can do better by modifying the threshold during the workload

Simple threshold results

	Olympics	Financial
Active blades avg.	**4.76**	**7.2**
Inactive blades	**40%**	**10%**
Energy savings measured	**38%**	**10%**
Savings possible from workload activity (limit)	**51%**	**32%**

Improving results

	Olympics	Financial
Active blades avg.	**4.76 (4.07)**	**7.2 (5.57)**
Inactive blades	**40% (49%)**	**10% (30%)**
Energy savings measured	**38% (45%)**	**10% (26%)**
Savings possible from workload activity (limit)	**51%**	**32%**

- **Change blade threshold as workload changes**
  - Requires knowledge of workload change at every point in time
  - Appropriate for predictable, cyclic workloads

Summary of PARD

- **4 dimensions of problem: energy, QOS, system characteristics, workload characteristics**
- **Developed method of adapting an industry standard workload to study power management**
- **Energy savings closely track with utilization of machine**
- **Future work**
  - Apply to non-connection workloads
  - Address non-cyclic workloads (spikes)
  - Extend model to multiple-power states (voltage scaling processors)

People

Pat Bohrer
Bishop Brock
Mootaz Elnozahy
Wes Felter
Jessie Gonzalez
Tom Keller
Mike Kistler
Ravi Kokku
Charles Lefurgy
Akihiko Miyoshi
Thanos Papathanasion
Jim Phelan
Karthick Rajamani
Ram Rajamony
Freeman Rawson
Alison Smith
Bruce Smith
Eric Van Hensbergen