

Resource Models: Cycle Stealing

- Last Time
 - » What is needed for Dynamic Adaptive Applications?
 - Resource-aware Applications; understand their performance
 - » VGrADS Project
 - Simpler Model: vgDL, Resource Performance Expectations, enable Separation of Concerns
- Today
 - » General Resource Management Problem
 - » Resource Models: Cycle Stealing
 - » Sharing Resources in Utilities
- Reminders/Announcements
 - » none

CSE225 – Lecture #9

Today's Readings

- Litzkow, Livny, Mutka, Condor: A Hunter of Idle Workstations, Proceedings of the 8th International Conference on Distributed Computing Systems, June 1988, San Jose, California.
- A. Chien, S. Marlin, and S. Elbert, Resource Management in the Entropia Desktop Grid System, in Grid Resource Management, Editors: Weglarz, Nabryzski, Schopf, and Stroinski, Kluwer Press, 2003.
- Artur Andrzejak, Martin Arlitt, Jerry Rolia , Bounding the Resource Savings of Utility Computing Models, Hewlett-Packard Technical Report Number 339, 2002, HPL-2002-339.

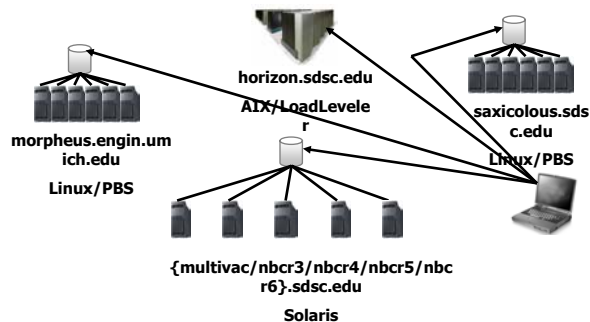
CSE225 – Lecture #9

Resource Management Problem

- Application Perspective:
 - » Given my application, find and bind an appropriate (“best”, “acceptable”, “best below acceptable”) set of resources
 - » => Optimize for application quality or performance
- System Perspective:
 - » For a set of resources, identify a set of applications which make good use of the resources (“best”, “acceptable”, “high utilization”, etc.)
 - » => Optimize for system utilization or “total value”
- Both of these problems require description of the resource needs/use. Both require collection of dynamic resource information.
- The application and system views are in conflict in many cases.

CSE225 – Lecture #9

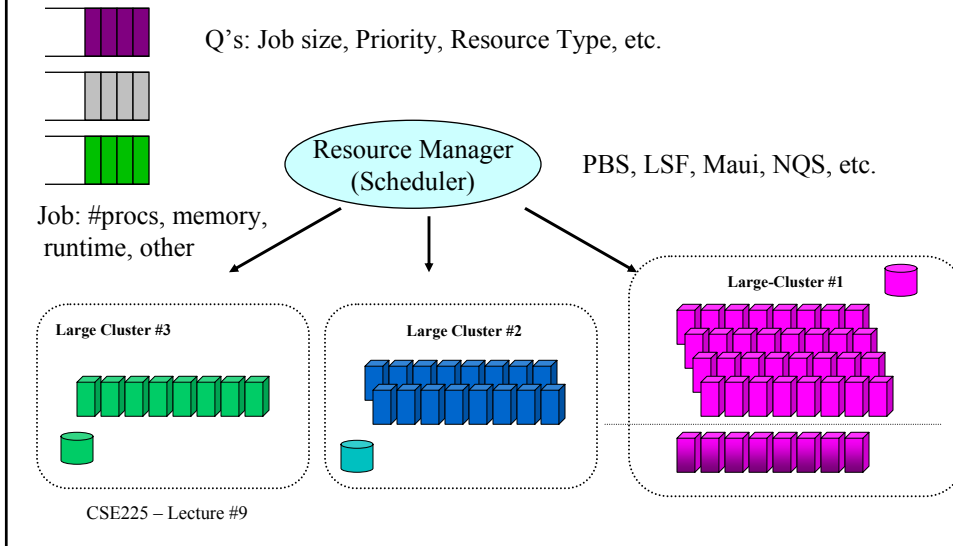
Where Should I Run parts of my Application?



- For performance (runtime), interactive response (turnaround)
- For reliability, variability, cost, etc.
- Networks matter as well as end resources

CSE225 – Lecture #9

Local Resource Management



Resource Management Models

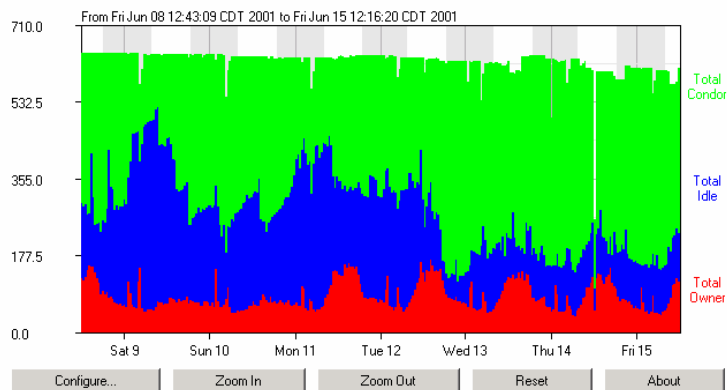
- “Cycle Stealing”
 - » Volatile, Asynchronous Preemption **TODAY**
- Dedicated Resource Scheduling
 - » Batch schedulers, dedicated resources, advanced reservation
- Time-sharing
 - » Slice schedulers, proportional share, guaranteed/best effort
- Objective: Understand implications for distributed application performance and motivations/advantages of various models (reach, local)

A “Cycle Stealing” Resource Model

- Application can use the Resource
- Right to use it may be terminated without notice
 - » User activity (keyboard, mouse, screensaver)
 - » Other higher priority activity (memory, cpu contention, any user activity)
 - » Inappropriate behavior (read file, opened network port, etc.)
 - » Resource failure (turned it off! Unplugged network, etc.)
- Why do this?
 - » Increased resource reach of the system
 - » Access large quantity of resources

CSE225 – Lecture #9

External Application View



- Asynchronously
 - » Submit jobs into matchmaking/batch system
 - » Jobs run when resources are available (competition, etc.)
 - » Eventually get notice for job completion

CSE225 – Lecture #9

Internal Resource View

- Highly optimized and controlled matchmaking process
- Resource policies are strictly observed
- Resources get well utilized (matching assumed to find all possible opportunities to run)
- Key flexibility is in the ability to migrate and passivate applications

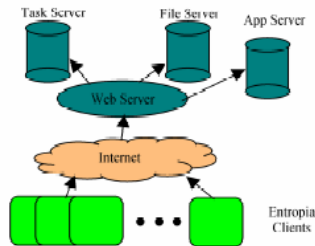
CSE225 – Lecture #9

Discussion

- What applications can be run in this model?
 - » Not interactive/not real-time, no deadline, partitionable, EP, not continuous
- How do applications deal with heterogeneity?
 - » Prohibit parts you can't deal with, deal with the rest
- How is the statistical behavior of the resource exposed to the application?
 - » Change in throughput, mostly masked
- How would I use this for – completing studies for a research paper? Doing a financial model ahead of a deadline?
 - » Implement a priority system? Designate resources in class-ads
 - » Ensure plentiful resources, limit access
- What is a denial of service attack in this system?
 - » Anything that consumes a LOT of resources

CSE225 – Lecture #9

A Modern Desktop Grid Architecture



Database of
All Attributes,
All History,
All resources

All scheduling can use
all this information

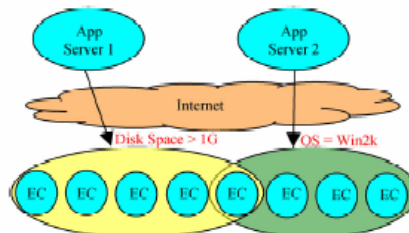
Examples:

- first available
- sufficient memory
- not a .com machine
- not a athlon processor
- not more than 100 cpus for this job
- ...

- Entropia Desktop Grid System
 - » Designed late 1999
 - » Modern DB, Web 3-tier architecture
 - » Centralized scheduling and control
 - » Designed to achieve turnaround
 - application quality of service
 - » Designed to have explicitly controllable scheduling priorities

CSE225 – Lecture #9

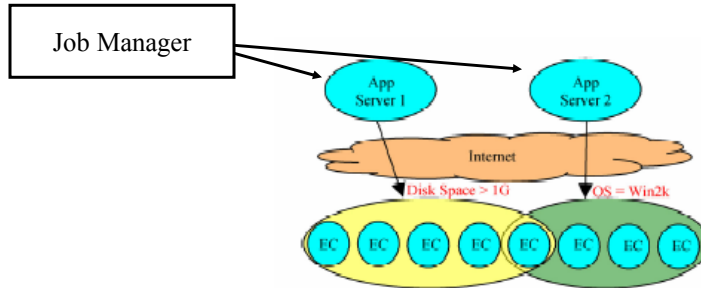
Entropia Resource Database and Scheduling Model: Pools



- Organized Resource Pools
- Homogeneity out of Heterogeneity (with scale)
- Scalable resource management
- Applications are Scheduled on a Single pool (at this level)
- Essentially PBS/Maui/LSF functionality within pools (you'll hear about these)

CSE225 – Lecture #9

Job Management and Prioritization



- Notion: Collections of subjobs (tasks) as an application
- Schedules applications relative to each other based on a PRIORITY and deadline
- Job process monitoring
- Ensures progress, reasonable completion

CSE225 – Lecture #9

The screenshot shows the Entropia web interface. At the top, there is a navigation bar with tabs for 'Jobs', 'Stage', 'Submit', 'Monitor', 'Results', 'Summary', 'Grid', and 'Security'. The 'Jobs' tab is active. Below the navigation bar, there are several sections:

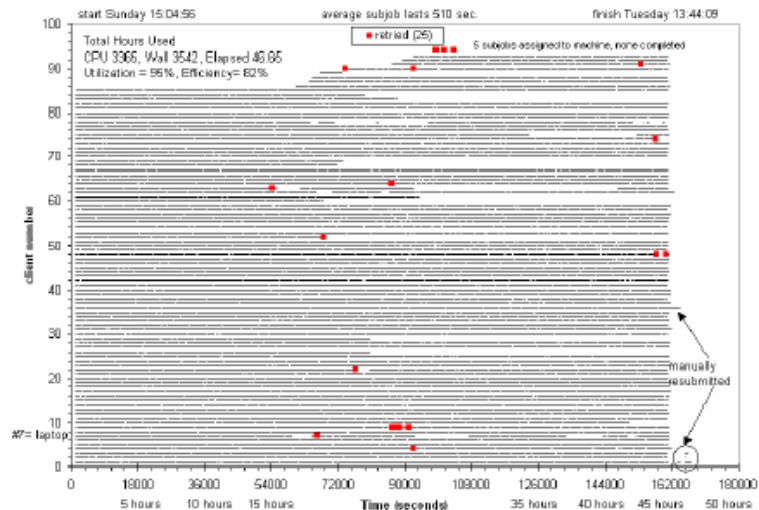
- Summary:** Shows 'All', 'Production', and 'Test' filters.
- Jobs:** A table showing 'Currently Running' and 'Completed Today' jobs.

Application	Jobs	Subjobs
DOCK	0	0
GOLD	0	0
BLAST	1	1
Completed Today	0	0
- Clients:** Shows 'Total CPU Power' as 1.992188 GFLOPS and 'Total Number of Clients' as 4. A breakdown shows 4 Active, 0 Idle, 0 Disconnected, and 0 Needs Attention clients.
- Recent 10 Running Jobs:** A table listing recent jobs with columns for Job Name, User Name, Application, # Subjobs, Status, % Done, Submit Time, and Avg Exec.

Job Name	User Name	Application	# Subjobs	Status	% Done	Submit Time	Avg Exec
BLAST Job	root	BLAST	10	Assigned	90	2/12/2002 4:27:29 PM	0
DCGT Test Job	root	DCGrid Tester	300	Assigning	0	2/12/2002 4:28:49 PM	0
DCGT Test Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:40:38 PM	0
DCGT Test Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:42:10 PM	0
DCGT Test Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:43:18 PM	0
DCGT Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:45:22 PM	0
DCGT Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:45:30 PM	0
DCGT Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:45:30 PM	0
DCGT Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 7:47:19 PM	0
DCGT Job	root	DCGrid Tester	20	ReadyForStarting	0	2/13/2002 8:50:51 PM	0

At the bottom of the page, there is a footer with '© 2002 Entropia, Inc.' and navigation links for 'Stage', 'Submit', 'Monitor', 'Results', 'Summary', 'Grid', 'Security', 'Logout', and 'Help'.

End Application Performance Characteristics



Observations

- Large scale resources can be managed with large quantities of static and historical information
 - » Feed into scheduling, management, other policies and actions
- Heterogeneity in large scale systems can be made into homogeneity thru scale
 - » How many configurations are there really? And matching problems made simple
 - » Locality is the big caveat here
- Application-oriented “Job” management is critical
 - » Must be coordinated with underlying resource management
 - » Priority and admission techniques used in Entropia work well
 - » Doesn't work in an “open” system

Sharing Resources: What efficiency gains?

Resource Descriptions and Usage

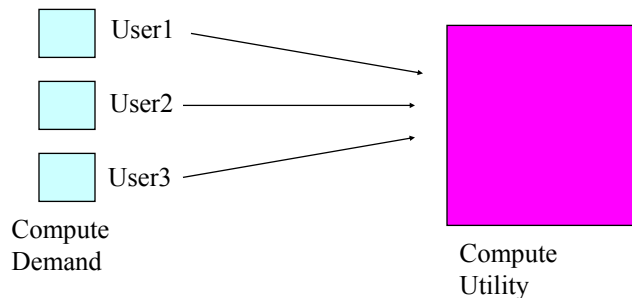
- All previous resource descriptions indicated desired attributes
- Description -> selection -> binding
- So, how much should I ask for?
 - » Resources as fractional quantities
 - » If I overask? If I underask?
- How much am I likely to use?
 - » Understanding application behavior
 - » How stable is this?

A Resource Specification

```
[
  Type = "Job";
  Owner = "roy";
  Universe = "Standard";
  Requirements = (other.OpSys == "Linux" && other.DiskSpace >
  140M);
  Rank = (other.DiskSpace > 300M ? 10 : 1);
  ClusterID = 12314;
  CPUSpeed = 350;
  JobID = 0;
  Env = "";
  ...
]
```

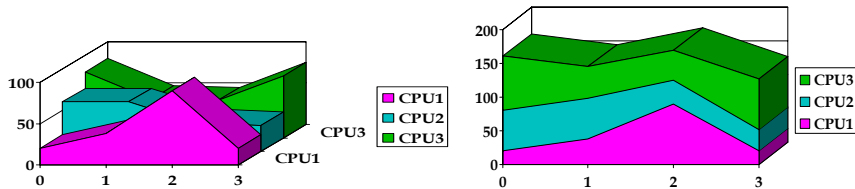
CSE225 – Lecture #9

A Computing Utility



- Compute users sharing a divisible resource
- Resource descriptions => division of the resource

Two Models of Division



- Non-work Conserving (full server utility) and Work Conserving (shared utility)

CSE225 – Lecture #9

Key Questions

- Non-Work Conserving: How much waste?
- Work Conserving: How much savings? (and how much “failure”)

CSE225 – Lecture #9

Discussion

- What implications for resource requests?
 - » Work-conserving: might want to ask for a little more or ask for little (to get in) and then use a lot; might design application to be more robust to variations in resource
 - » Non-work conserving: don't over-ask, incentive to smooth out resource consumption, and ask for less
- What implications for resource utilities?
 - » Opportunity to overbook (but conflict with bill-for-use)
 - » Want customers with flexible (not rigid) requirements (or dictatable requirements)
 - » Penalize people who overask (or charge for asking); Penalize people who overuse
 - » Complementary customer use (maximal statistical multiplexing benefit)
- How typical are these results? How broad the impact?
 - » Very widespread, the one anomaly is supercomputing users

CSE225 – Lecture #9

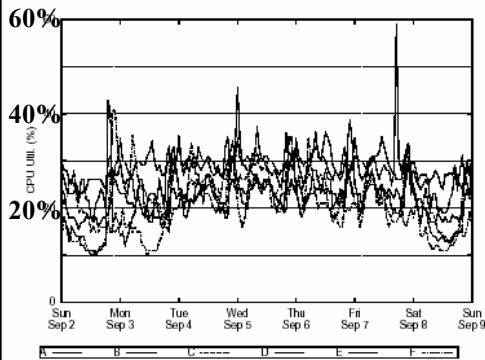
Study of Data Centers

- CPU Utilization, 5 minute averages
 - » MP – over all the cpu's in a server
- 1000 servers
- 6-7 weeks of data
- Late 2001

- => some interesting “holes”

CSE225 – Lecture #9

How Busy?



- 6 data centers
- 80th percentile (cpu's below the curve)
- Lots of "idle" resources
- Caveats:
 - » Large variability
 - » 5 minute averages (long?)

Figure 2: 80-percentile of CPU utilization Servers of Six Data Centers

CSE225 – Lecture #9

Drilling Down, a Single Center

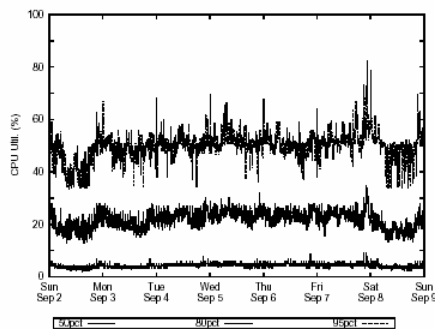


Figure 3: Detailed CPU Utilization for Data Center A

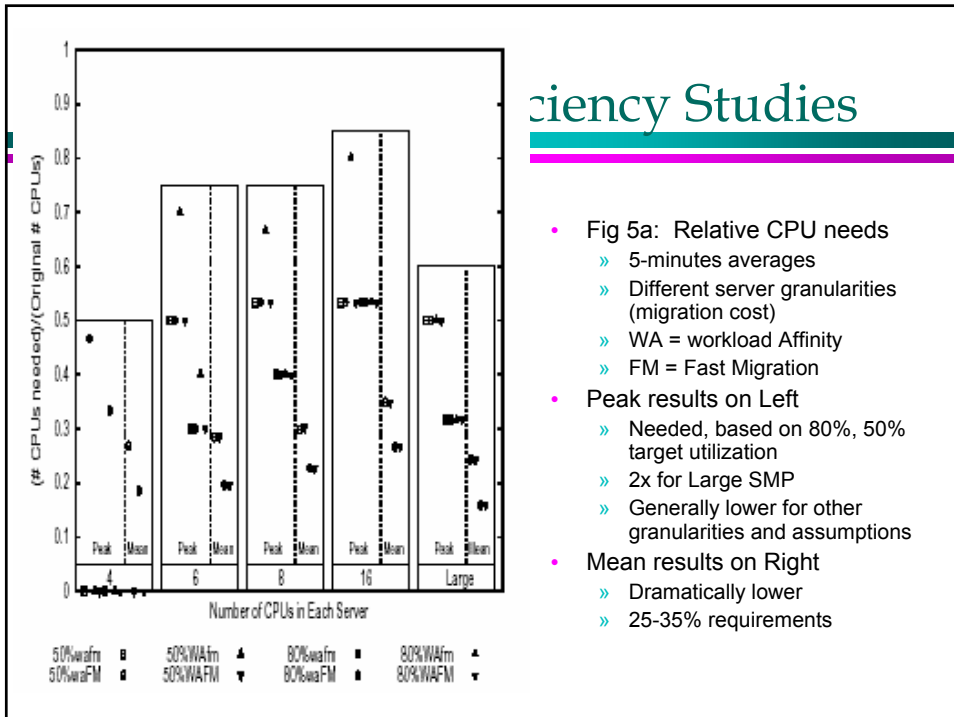
- Single Data Center
- 50th, 80th, 95th percentile
 - » Specific CPU's can shift across
- => Over 1/2 the CPU's less than 10% load!
- => 80% of CPU's have less than 20% load
 - » You thought desktops were idle???
- Only ~5% of CPU's have greater than 60% utilization

CSE225 – Lecture #9

Translation to a Grid Context

- IT Administrators have allocated Servers to Applications
 - » Resource specification => Resource Binding
- What is the relationship of their specification to resource consumption?
 - » Performance? Cost?

CSE225 – Lecture #9



- Fig 5a: Relative CPU needs
 - » 5-minutes averages
 - » Different server granularities (migration cost)
 - » WA = workload Affinity
 - » FM = Fast Migration
- Peak results on Left
 - » Needed, based on 80%, 50% target utilization
 - » 2x for Large SMP
 - » Generally lower for other granularities and assumptions
- Mean results on Right
 - » Dramatically lower
 - » 25-35% requirements

Discussion

- Limitations –
 - » MP cost not same as discrete servers; fewer processors; cost of ownership
 - » With higher CPU utilization, memory hierarchy may perform worse
 - » Correlated failures?
- How much waste in whole server / non-work conserving utilities?
 - » Huge waste, 50% - 75%
- How much benefit if we build shared utilities?
 - » Complement huge waste, potentially large
 - » How much cost if applications don't get their requirements satisfied
 - » May need intelligent load shedding
- How definitive are these measurements?
 - » Measurement may not correspond to driving design requirements for capacity
 - » Interesting aggregation, combining wide range of requirements and applications
 - » Doesn't capture other dimensions of resource limits (IO, network, user, ?)
 - » 5-minute intervals seem long, are they representative
 - » 6-weeks may not be enough data
 - » Need way to simulate rare events...

CSE225 – Lecture #9

Discussion: Open Questions and Ideas

- So, how volatile are desktop systems? And what implications does this have?
- Can these systems be a proving ground for “extreme” distributed or grid computing?
 - » High asynchronous failure rates
 - » Complex resource use policies
- Limited application structures make good use of these resources; can these be broadened?
 - » Large compute
 - » Idempotent (retryable) computations
- What role can desktop resources play in an overall grid?

CSE225 – Lecture #9

Summary

- Cycle Stealing Resource Model
 - » Large Reach, “High Throughput Computing”
 - » Massive Heterogeneity, complex resource behavior
- Mature systems Implement
 - » Resource Pooling: Homogeneity from Scale
 - » Sophisticated Scheduling based on static stats; can ensure Application SLA
 - » Large-scale resource, unreliable for incorporation into distributed apps
- Resource specifications and Resource Use
 - » Resource specifications and analogy to full server utility
 - » Low Utilization for expensive servers!
- Implications for Utilities and Resource Efficiency
 - » Combining based on averages yields 4x + !
 - » Combining based on actual usage in each interval with a target of 80% yields 2x