The Entropia Machine for Desktop Grids

Brad Calder, Andrew Chien, Ju Wang, Don Yang

Oct 28,2004

Presented by: Dongyu Liu

Grid Introduction

- *Grid computing*, which is defined as "coordinated resource sharing and problem solving in large, multi-institutional virtual organization." [Foster and Kesselman 1999]. More specifically, a Grid is an infrastructure for globally sharing compute-intensive resources such as supercomputers or computational clusters.

Current grid technology

- Grid technology efforts are now focused around the Global Grid forum and the Globus project .
- A computing grid can be seen and used as a single, transparent computer. A user logs in, starts jobs, moves files, and receives results in a standard way.

How grid system works?

- The grid system can sustain multiple teraflops by aggregating unused capacity from millions of computers.
- Aggregating following resources:
  CPU: 300Mhz -3.0Ghz
  high-speed local network: 100Mbps to 1G
  main memory: 256M-1G
  large disks: 60G-150G

What application Grid suit for?

- Large scale of computational problems:

Data mining

Molecular interaction

Financial modeling

DNA sequence analysis

Etc.

Entropia Introduction

- http://www.entropia.com/

The entropia DC Grid has been deployed at over 12 industry sites, and have been used for 50 applications.

The majority of machines run windows x86 machines. Specifically most operating system are NT,2000 and XP. (Compatibility)

Entropia Introduction

- Three components:

Desktop Control

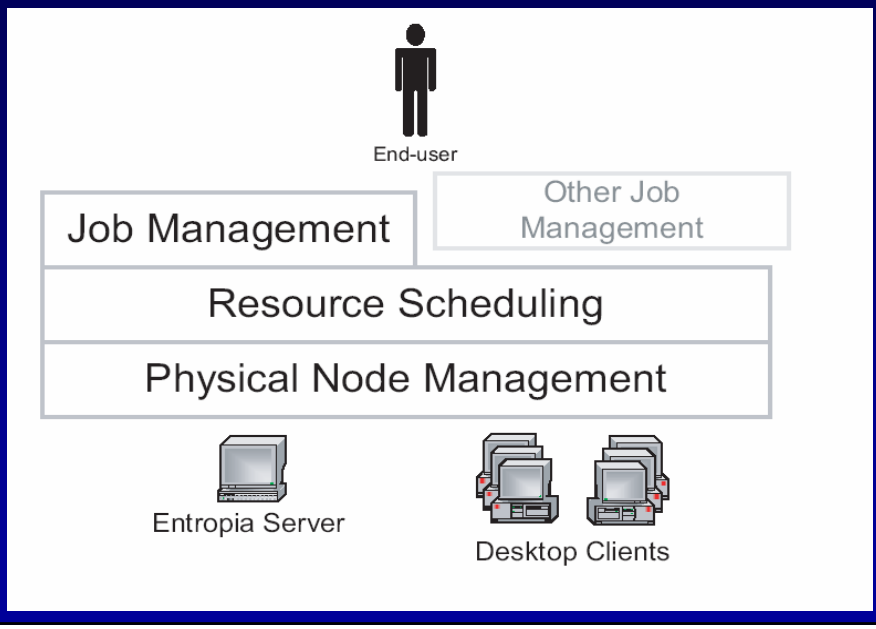
Sandbox

Application Security

Desktop grid solution features

- Efficiency: harvest unused cycles efficiently
- Robustness: finish jobs with minimal failures.
- Scalability: upward and downward with reasonable efforts.
- Manageability: No incremental human efforts
- Ease of Application Integration: support all kinds of applications

Entropia architecture



Entropia system architecture overview

- Layered architecture:

raw desktop resource → a single logical resource.

Physical Node Management: low level reliability and provide resource and application management.

Resource Scheduling: accept unit of computation and match them and schedule them to execute.

Job Management: decompose job, manage the progress the job and aggregate the results of sub-jobs.

How a job flows through system

- 1. Authenticate DCGrid user to system
- 2. User submit a job.
- 3. Job is broken into sub-jobs.
- 4. The related binaries are wrapped in EVM and created “sandboxed” version.
- 5. Submit the sub-job to the resource scheduler.

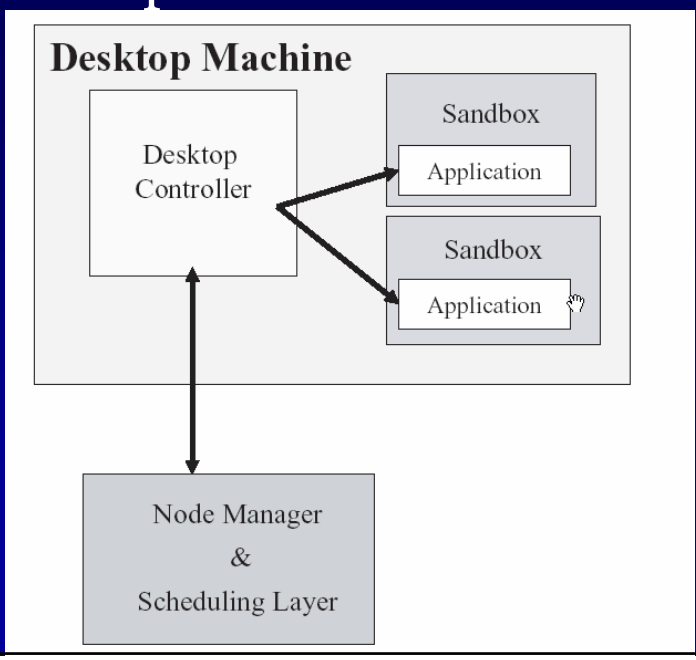
How a job flows through system (cont.)

- 6. Run the sub-job on a desktop machine.
- 7. User can check sub-job status.
- 8. Process output files to analyze the results when sub-job is completed

Desktop Grid Virtual Machine Requirement

- Desktop Security: protect the integrity of computer resources.
- Application Security: protect the integrity of the distributed computation
- Unobtrusiveness: not to interfere with the primary use of the desktops. (killing an application's processes)

Entropia Virtual Machine



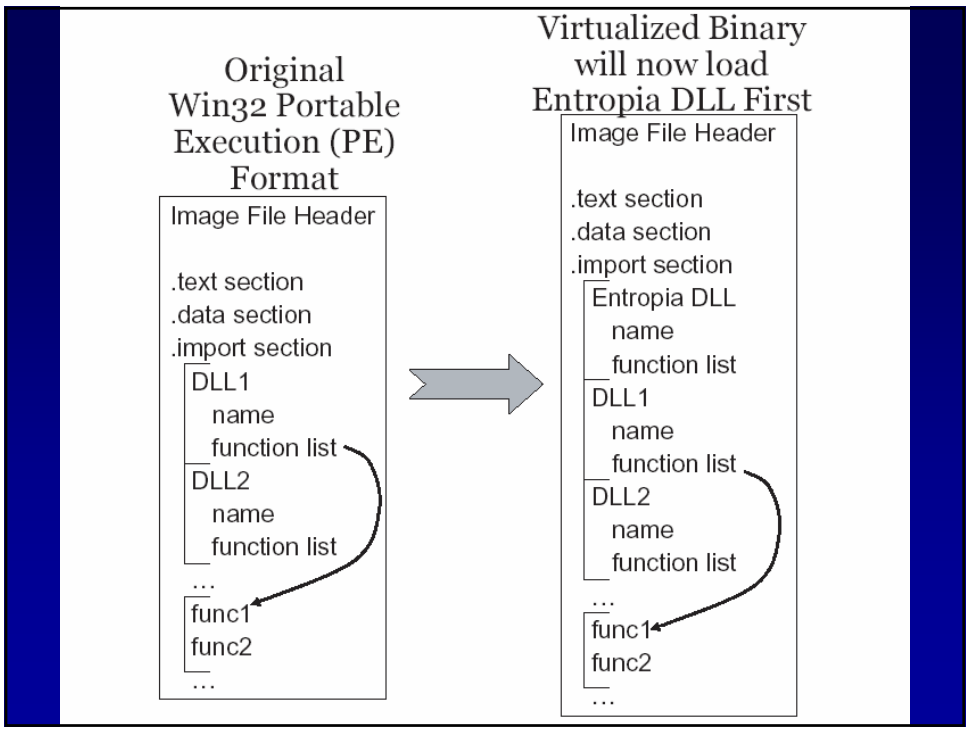
Enabling Applications

- To contain the application inside the sandbox, application wrapping is used to insert a mediation layer.
- Application integration: “sandboxing”
- The following are mediated: file system access, network communication, registry access , process control and memory access.

VM wrapping

The wrapping of the binary is achieved by rewriting the import table of the binary, so that Entropia vm.dll is the first dll in the list.

Dll_main inside vm.dll is executed before any non-system code for the program’s execution and encase all binary and dlls.



Sandbox execution

- The goal of sandbox is to control the application's interaction with the operating system, and to virtualize some of the operating system components.
- Access file system, registry and graphical user interface.
- The pc state isn't changed.

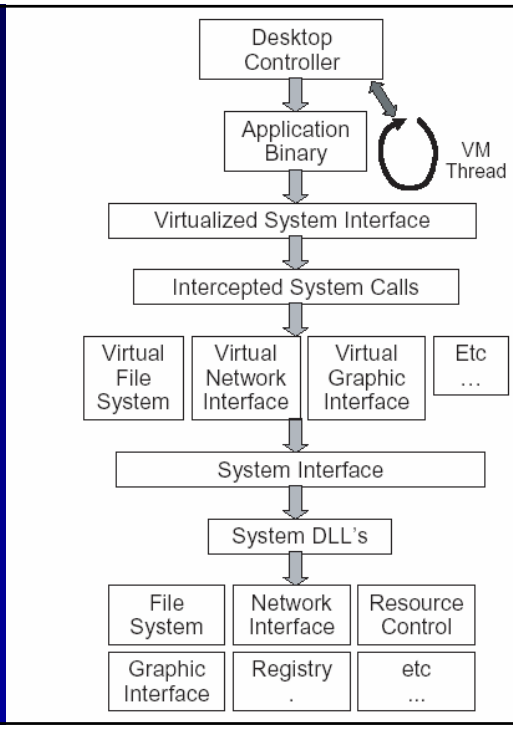
Mapping of file and directory

- To guarantee that application read and write in the sandbox, we have to map the file and directory structure.
- Application may believe that it is reading or writing to the directory C:\Program Files\when in fact it is writing to a sandbox directory deep within the Entropia software installation (e.g., C:\Entropia\root\C:\Program Files\).

Create an OS Interception Layer

We need to intercept low level Windows API calls in NT.dll, User32.dll, GDI32.dll and Kernel32.dll

- In addition binary and any libraries loaded are scanned.



Virtualized Components

- File virtualization
- GUI virtualization
- Registry virtualization
- Network virtualization
- Thread and Process Control (fork bomb)

Desktop Control

- EVM Portal Thread and Process Control
- Enforcing Resource Limit.
- Dealing with resource problem.

Application security

- File encryption

The system uses 3DES encryption and the block size is 8 bytes or 64 bytes.

- Detecting Application Tampering

When machine was rebooted from linux, a configuration file will keep the checksum of binary and data file.

Performance Evaluation

- The Entropia DC Grid has been deployed at over 12 industry sites, and have been used for over 50 applications.
- Demonstrate the behavior of four Entropia applications:

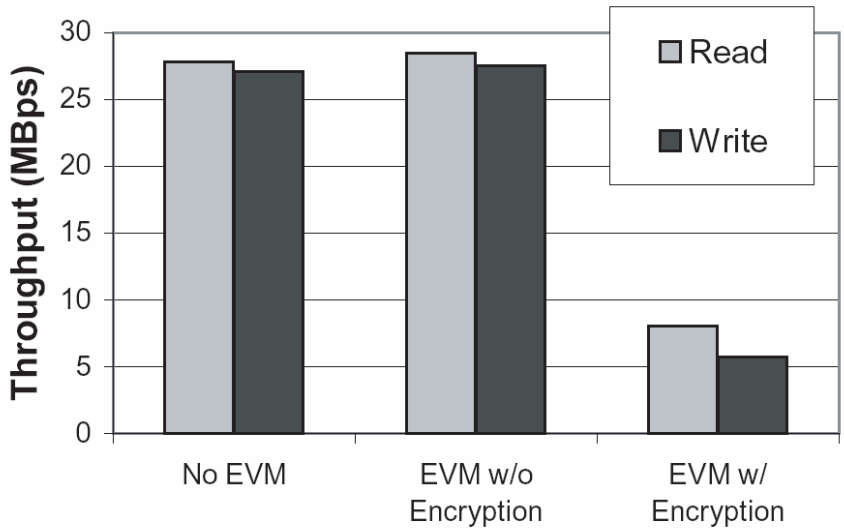
Virtual Screening: DEPLOY,FRED and MOE,

Sequence Analysis: HmmerSearch

Performance Evaluation

Application	Average Running Time (seconds)	Average Bytes Written (kb)	Average Bytes Read (kb)	Sum Bytes I/O / Run Time	Sum Bytes (I/O) NonVmized / Vmized	Run Time NonVmized / Vmized
DLPOLY	1917.9	1622.0	1019.0	1.4		
DLPOLY w/ EVM	1924.9	2396.0	1801.0	2.2	1.6	1.0
FRED	244.1	838.0	552.0	5.7		
FRED w/ EVM	242.6	907.0	603.0	6.2	1.1	1.0
HmmSearch	5683.5	0.1	269357.0	47.5		
HmmSearch w/ EVM	5763.4	0.1	269357.0	46.9	1.0	1.0
MOE	839.6	77.0	2335.0	2.9		
MOE w EVM	889.1	853.0	2409.0	3.7	1.4	1.1

Encryption performance



Related work

- Web-base general purpose vm
  Java Virtual Machine (JVM)
  .Net

Difference with Entropia VM

Related work (Cont'd)

- Another class of virtual machine systems is VMware and Terra. They allow multiple operating systems to run concurrently.
- VMware:
- Terra:
- Denali
- Virtuoso
- NGSCB

Summary

- A detailed requirement for desktop grid virtual machine environment.
- How the EVM used binary rewriting and desktop controller to enforce unobtrusive behavior.
- This approach provides security for the application and their data.

Characterizing and Evaluating Desktop Grids: An Empirical Study

- Derrick Kondo†, Michela Taufer† §, Charles Brooks III §, Henri Casanova†‡, Andrew A. Chien†

† University of California at San Diego

‡ San Diego Supercomputer Center

§ The Scripps Research Institute

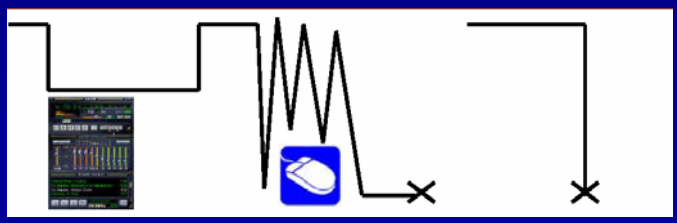
Desktop Grid Background

- Set of (shared) network-connected resources
- High computational power at low cost
  - Reuse existing infrastructure of resources
- Successful deployment of compute-intensive applications
  - E.g. SETI@home, folding@home, fightaids@home
- Computing platform
  - Internet
  - Enterprise

Desktop Grid Resources

- Resources are extremely heterogeneous
  - –E.g. in terms of CPU, memory, disk space, network connectivity, OS
- Resources are volatile

- **CPU Availability**



Goal & Approach

- Determine the utility of desktop grids for high throughput, task parallel applications
  - Develop performance model
  - Quantify utility in terms of cluster equivalence
- Measurements of resource availability

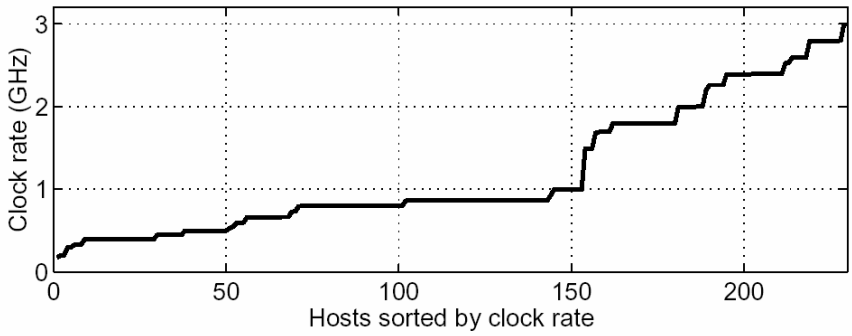
Related Work

- Monitored CPU availability [Wolski99, Wolski99+, Dinda98, Boloskyoo, Arpaci95]
  - Difficult to determine effect on desktop grid behavior
    - OS idiosyncrasies
    - Ignores keyboard/mouse activity
      - E.g. hard to infer task failures

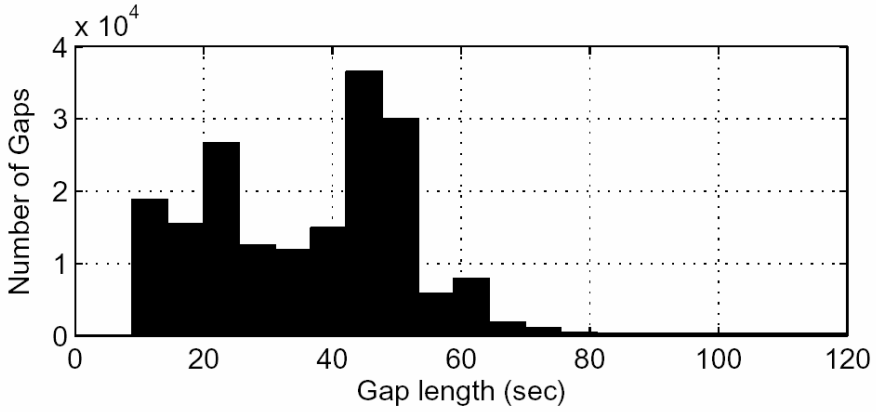
Method

- ***Intrusive*** measurements on ***Entropia*** desktop grid system
  - –Fixed time-length tasks
  - –Every 10 seconds the program writes the number operations completed to file
  - –Output files assembled to produce a CPU availability trace
    - Interpolated gaps due to system overhead
  - –220 machines at SDSC
  - –Cumulative measurement period: 1 month

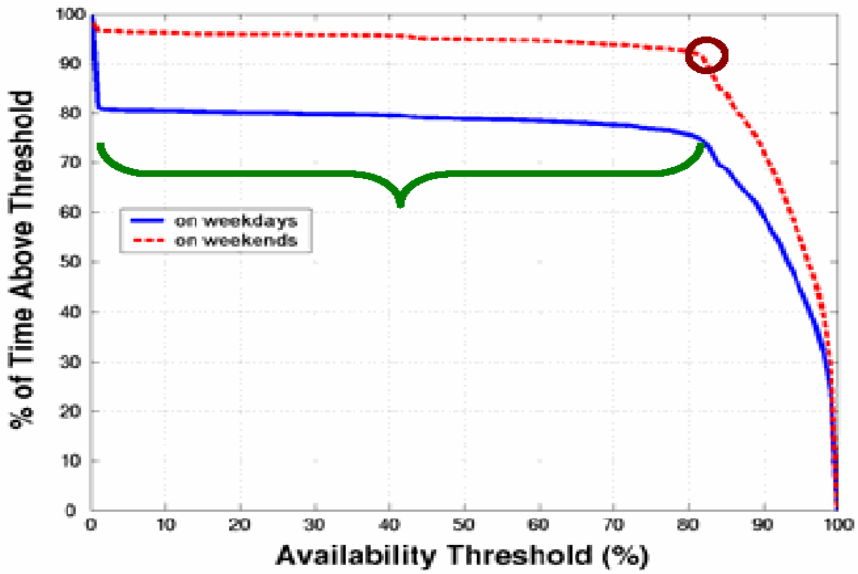
Host clock rate values for all hosts



Length distribution of small gaps



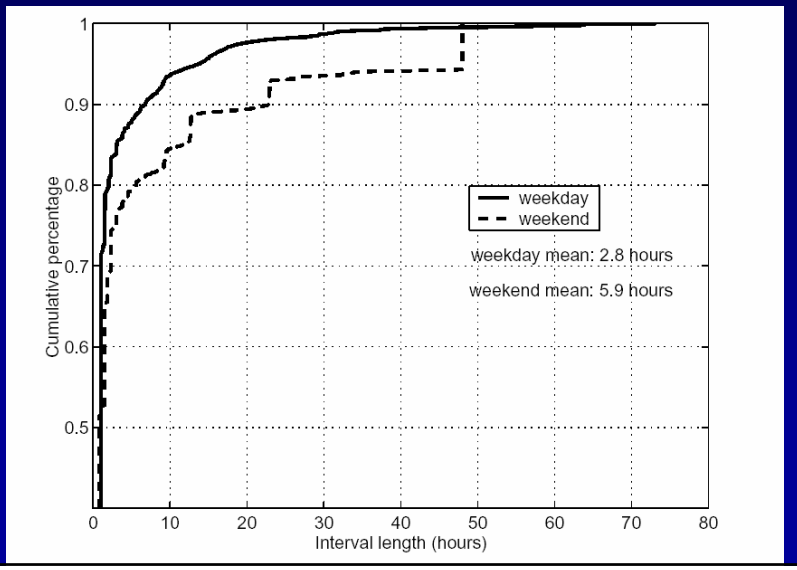
CPU Availability



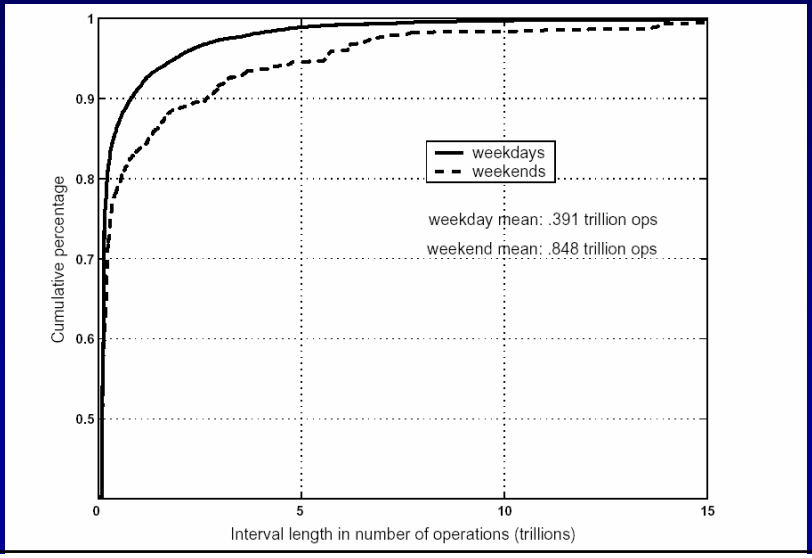
How to explain

- The data point (x, y) means that y% of the time CPU availability is over x%.

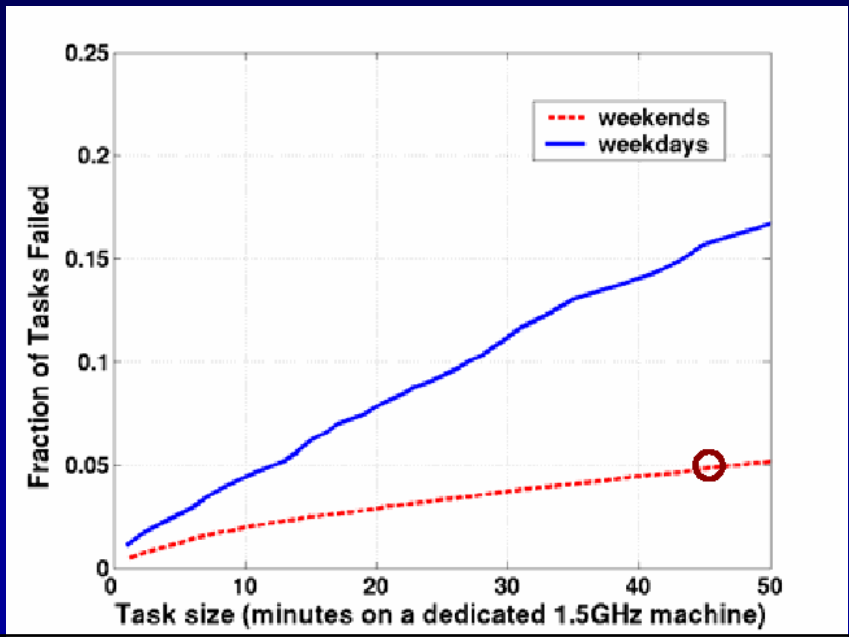
Cumulative percentage vs. interval



Cumulative percentage vs. interval



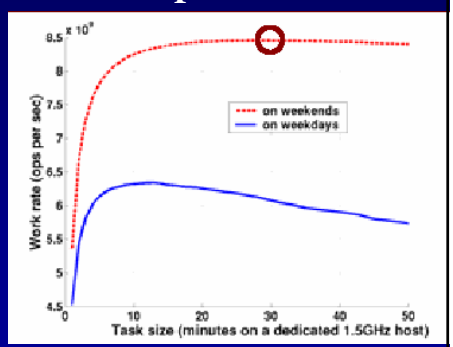
Task Failure Rate



Performance Model

- **N**: number of hosts
- **s**: operations per task
- **F (s)**: failure rate
- **r**: average ops per sec for a host
- **g**: average system overhead per task
- **W (s):** aggregate ops per sec

Optimal task size



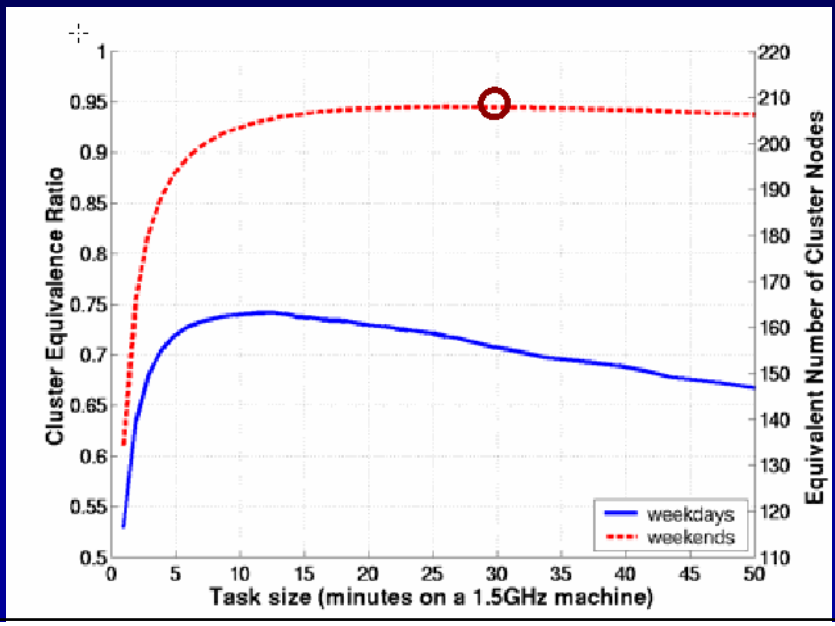
$$W(s) = \frac{N * r (1-f(s))}{1+(r/s)*g}$$

Cluster Equivalence

- Compare utility of desktop grid with that of a dedicated cluster
  - High throughput, task parallel applications
- Determine M/N cluster equivalence ratio

– Given N-host desktop grid, what is equivalent M-node dedicated cluster

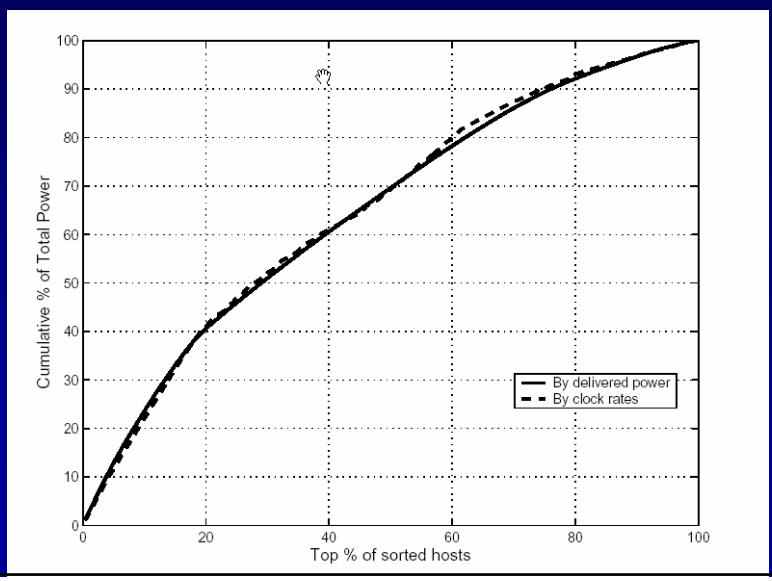
Cluster Equivalence



Contributions

- Measurement data
  - –Captures temporal structure of resource availability
- Model of desktop grid work rate
- Quantify desktop grid utility for high throughput, task parallel applications using cluster equivalence metric

Cumulative percentage of total power



Current and Future Work

- Traces of other desktop grids
  - Xtremweb, BOINC
- More detailed characterization
  - E.g. at host level
- Resource selection for rapid application turnaround

Questions?