Towards Scheduling Virtual Machines Based On Direct User Input

Bin Lin

Peter A. Dinda

Department of EECS
Northwestern University

http://presciencelab.org

Take-away points

- Discovered high variation in user expectations of performance
- Developed interface that captures user variation for CPU scheduling in VM desktop replacement scenario
- Evaluated interface in extensive user study; finding it to be effective
- Currently extending direct user feedback model for other systems problems, including power management

Outline

- **Background**
  - **Virtuoso [http://virtuoso.cs.northwestern.edu]**
  - **User diversity**
    - **User comfort with resource borrowing [Gupta & Lin, HPDC'04]**
  - **Scheduling VM in Virtuoso**
    - **VSched [Lin, SC'05]**
- **Direct user input in VM scheduling**
- **User study**
- **Conclusions**

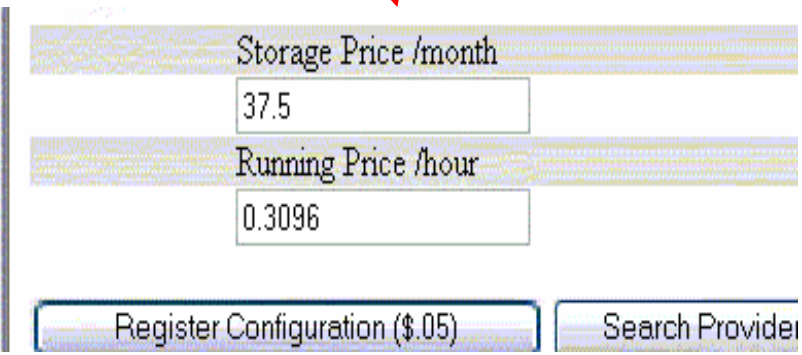
Outline

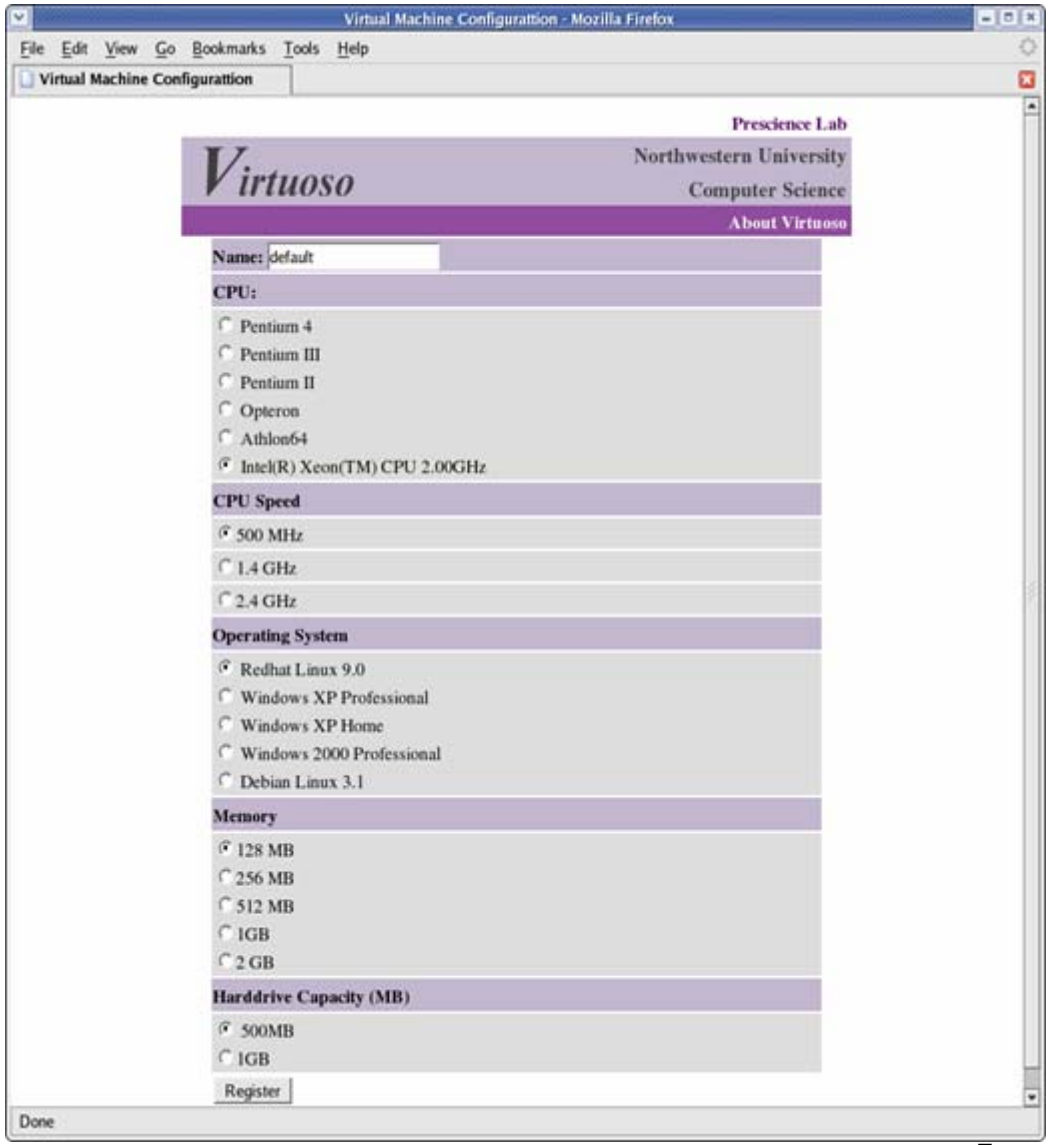
- **Background**
  - **Virtuoso**
  - **User diversity**
    - **User comfort with resource borrowing**
  - **Scheduling VM in Virtuoso**
    - **VSched**
- **Direct user input in VM scheduling**
- **User study**
- **Conclusions**

Virtuoso: VM-based Distributed Computing

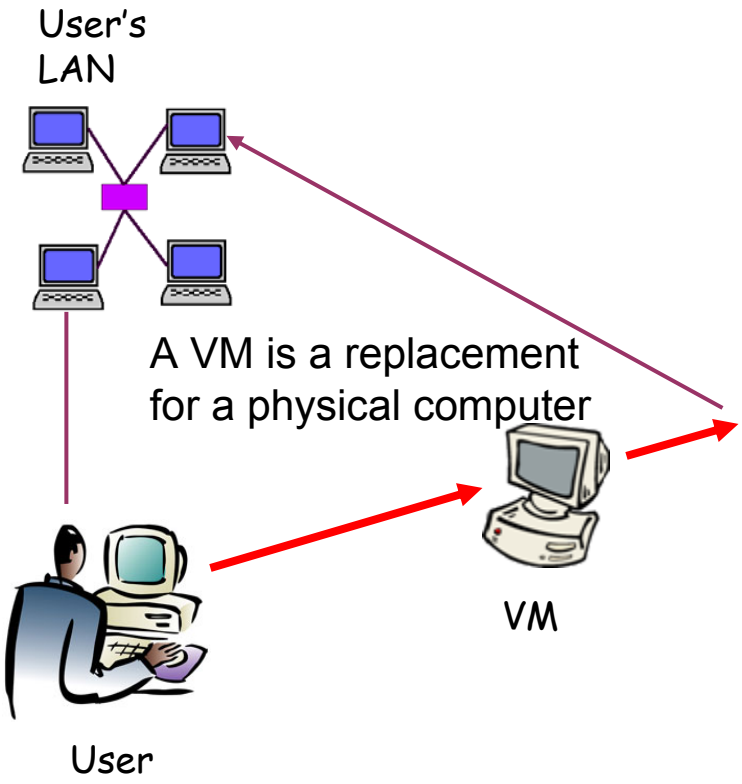
User

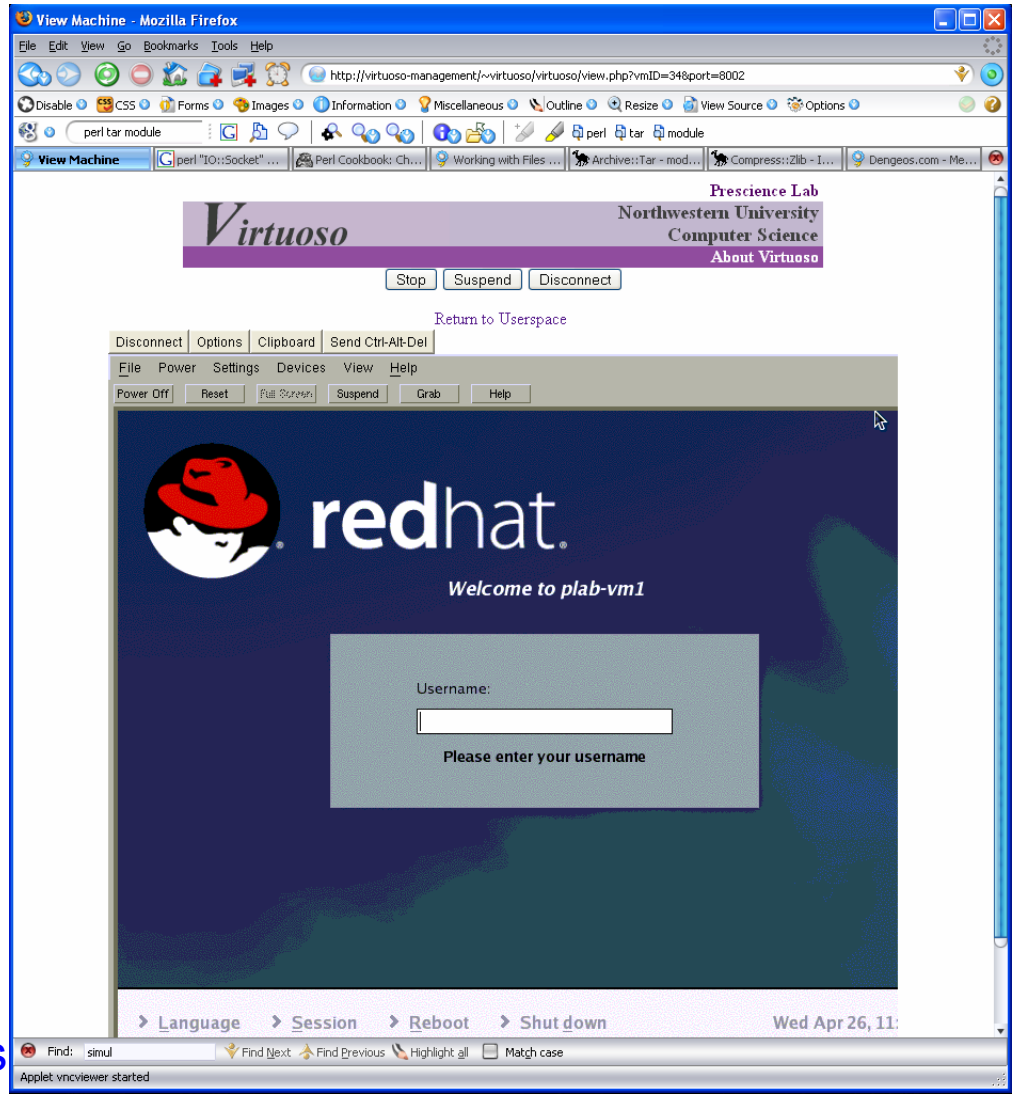
Orders a raw machine





User's View in Virtuoso Model





Multiple VMs may run simultaneously on the same host

Use of existing, unmodified applications & operating systems

[ICDCS'03; NWU-Tech-Report, HPDC'05; IPDPS'06]

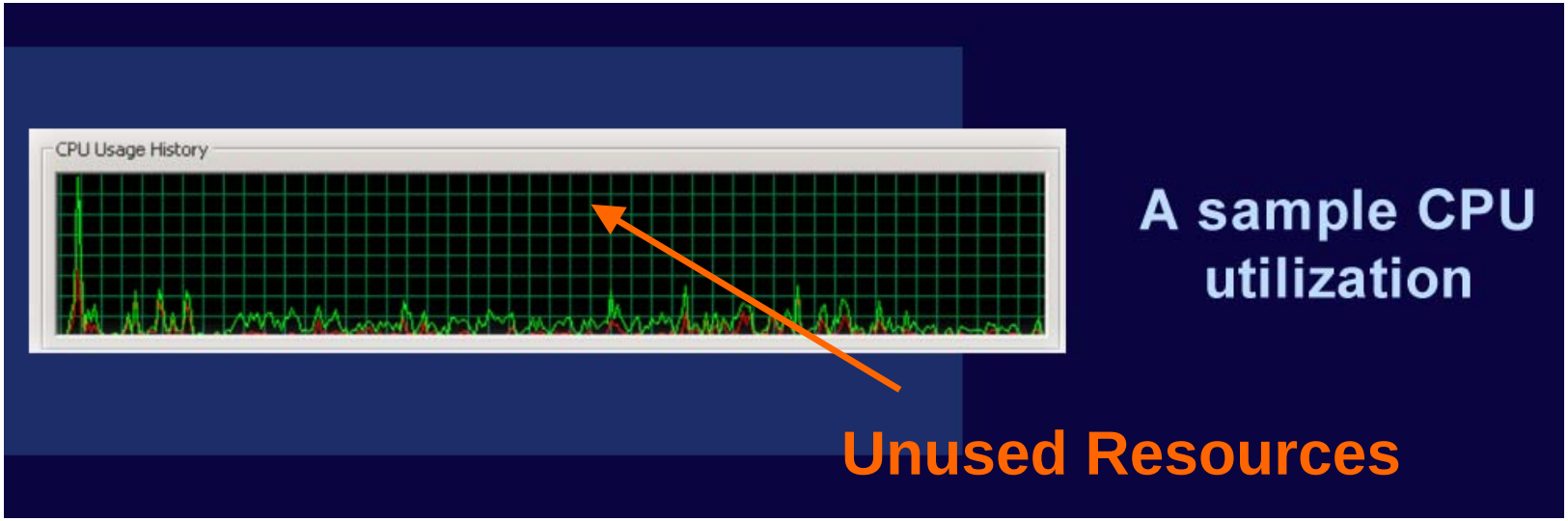
Outline

- **Background**
  - Virtuoso
  - **User diversity**
    - **User comfort with resource borrowing [Gupta & Lin, HPDC'04]**
  - Scheduling VM in Virtuoso
    - VSched
- Direct user input in VM scheduling
- User study
- Conclusions

Measuring and understanding user comfort with resource borrowing

- *HPDC'04*
- Goal: discover how aggressive resource borrowing systems like *SETI@home* can be
  - Or necessary resource share of desktop replacement virtual machine
- Extensive user study

Observation and ideas

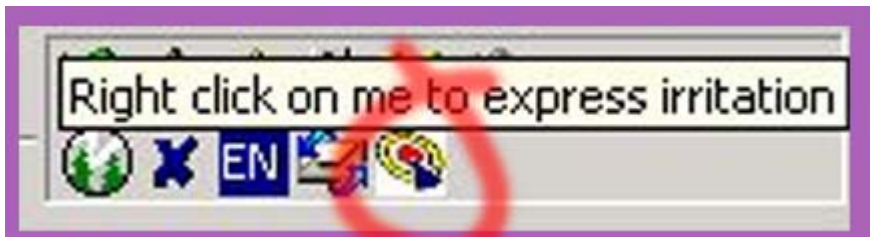


Idea: Why not borrow the unused resources ?

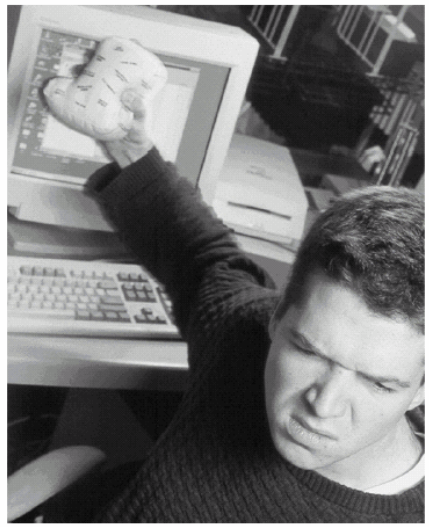
Problem: Performance Slowdown

Understanding User Comfort with Resource Borrowing

- What level of resource borrowing leads to user discomfort for significant fraction of users ?
- A system which emulates resource borrowing *(CPU, MEM, DISK)* and captures user feedback



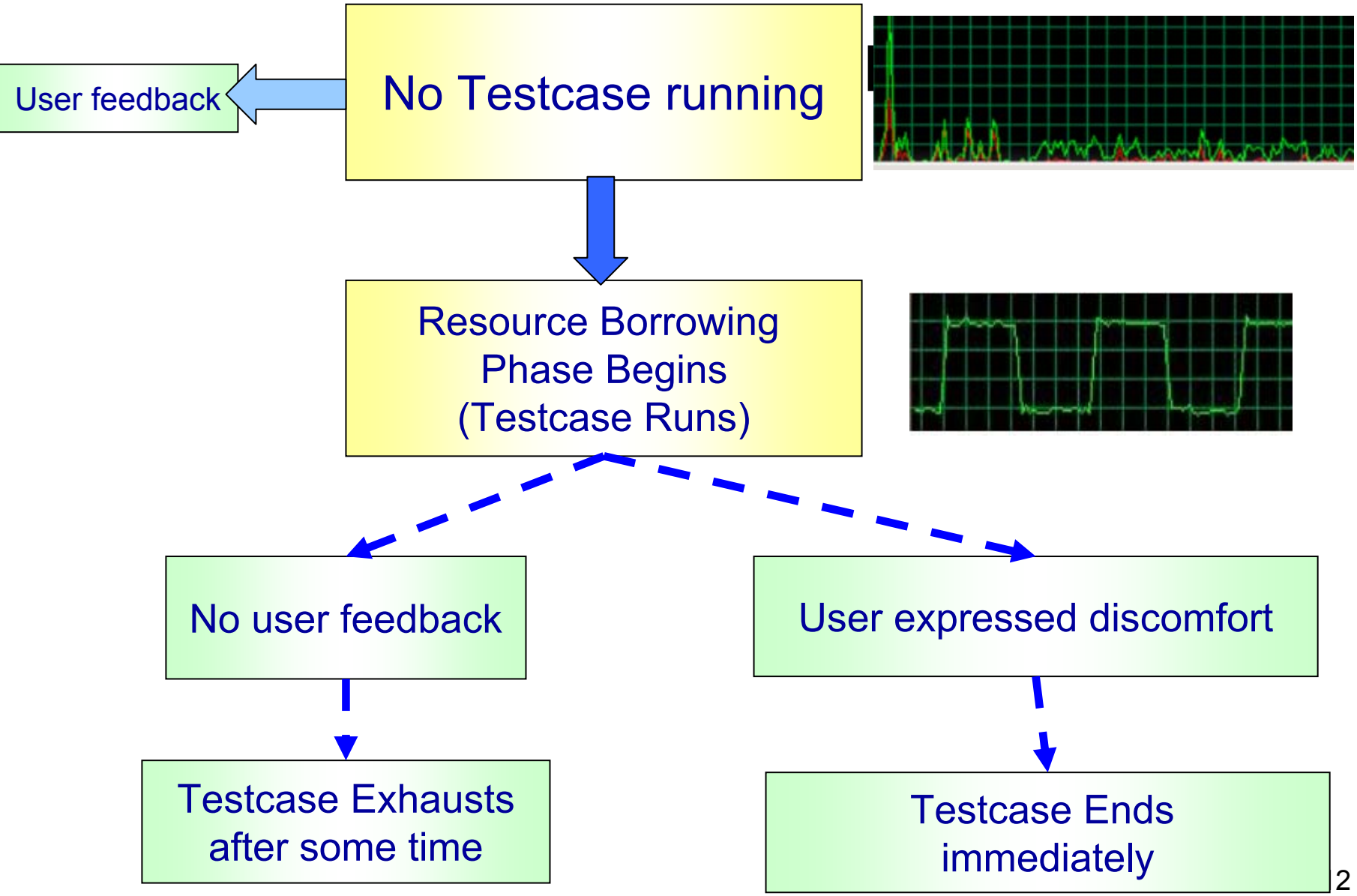
F11



The controlled study description

- 35 undergrad and grad students
- 1.5 hrs each
- Each user was assigned 4 interactive tasks to do
  - MS Word
  - MS Powerpoint
  - MS Explorer → searching and saving information
  - Quake III

Flowchart of Testcase



Resource Exercisers

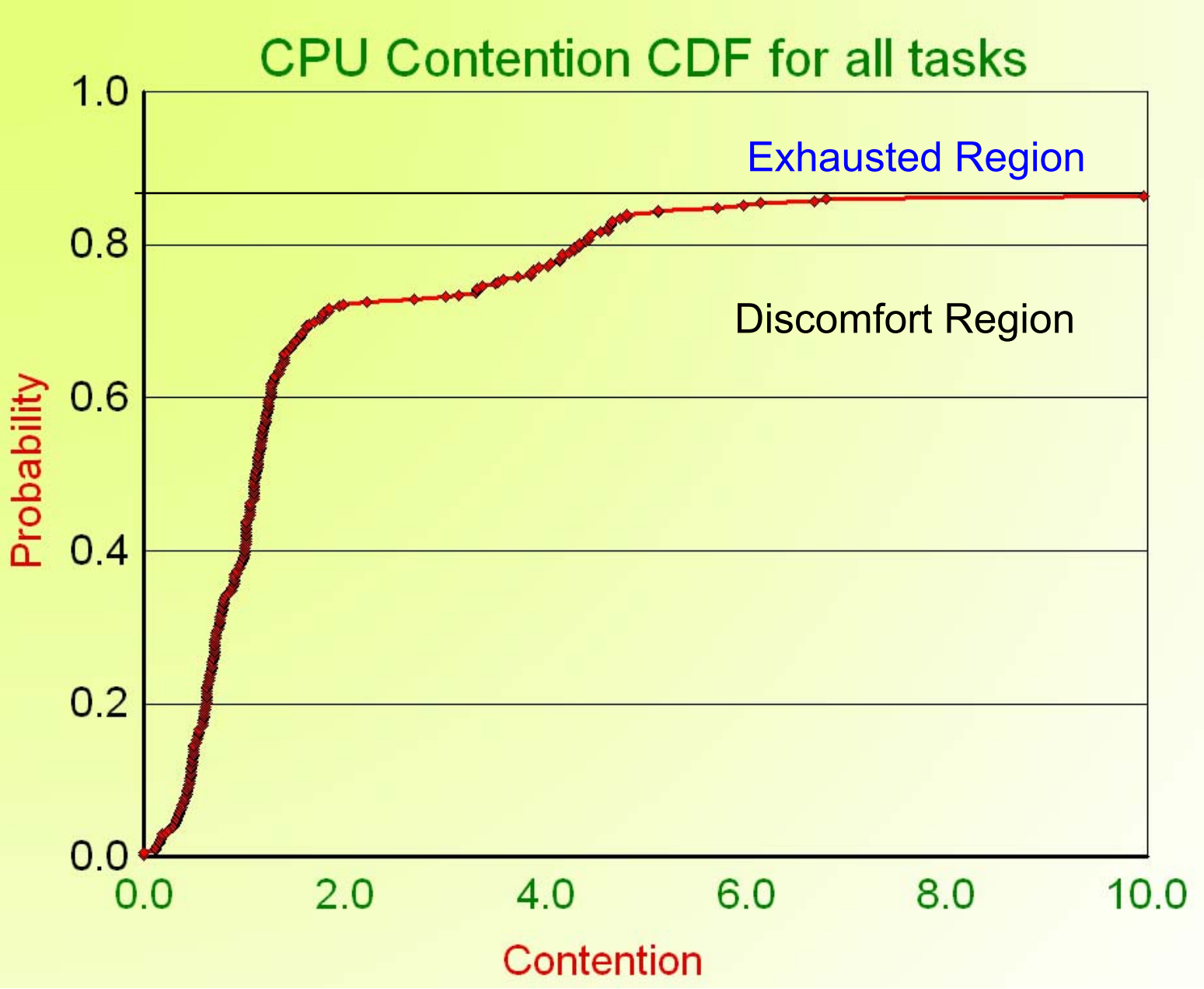
- CPU Exerciser
  - Contention describes the expected extra number of threads in ready queue
  - Fractional resource borrowing using stochastic scheduling methods
  - Validated to contention level of 10
- Disk Exerciser
  - Random seek and read/write in a large file (twice the memory)
  - Validated to contention level of 7
- Memory Exerciser
  - Borrows a fraction of physical memory: from 0 to 1

Resource borrowing vs User Discomfort

- CDFs for discomfort contention level
- Not all contentions cause discomfort: exhausted region

- CDFs allow us to read %age of people discomforted for a given contention
  - **Metric $c_{0.05}$** : At what contention do we discomfort only 5% of the people ?

High variation

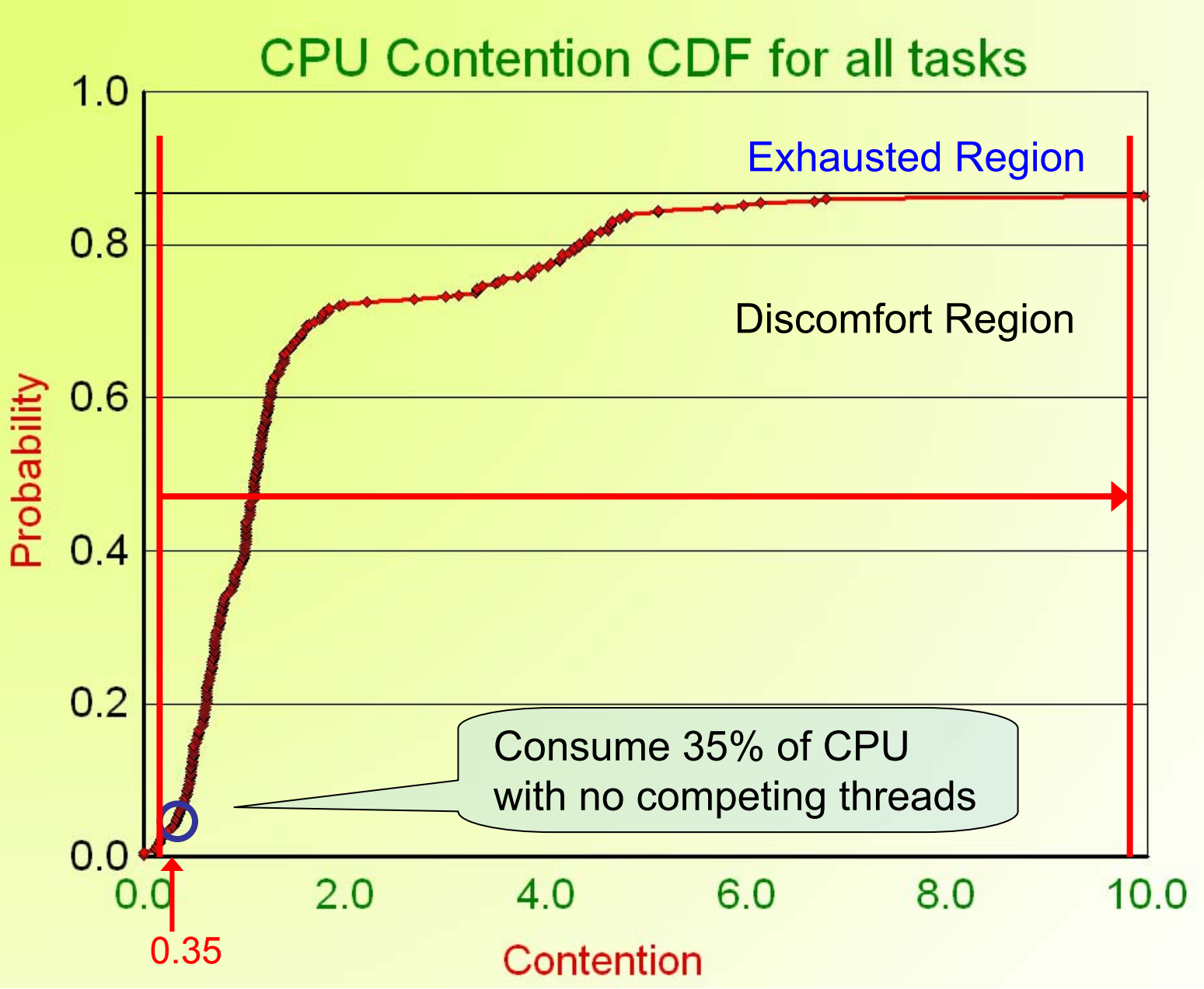


Resource borrowing vs User Discomfort

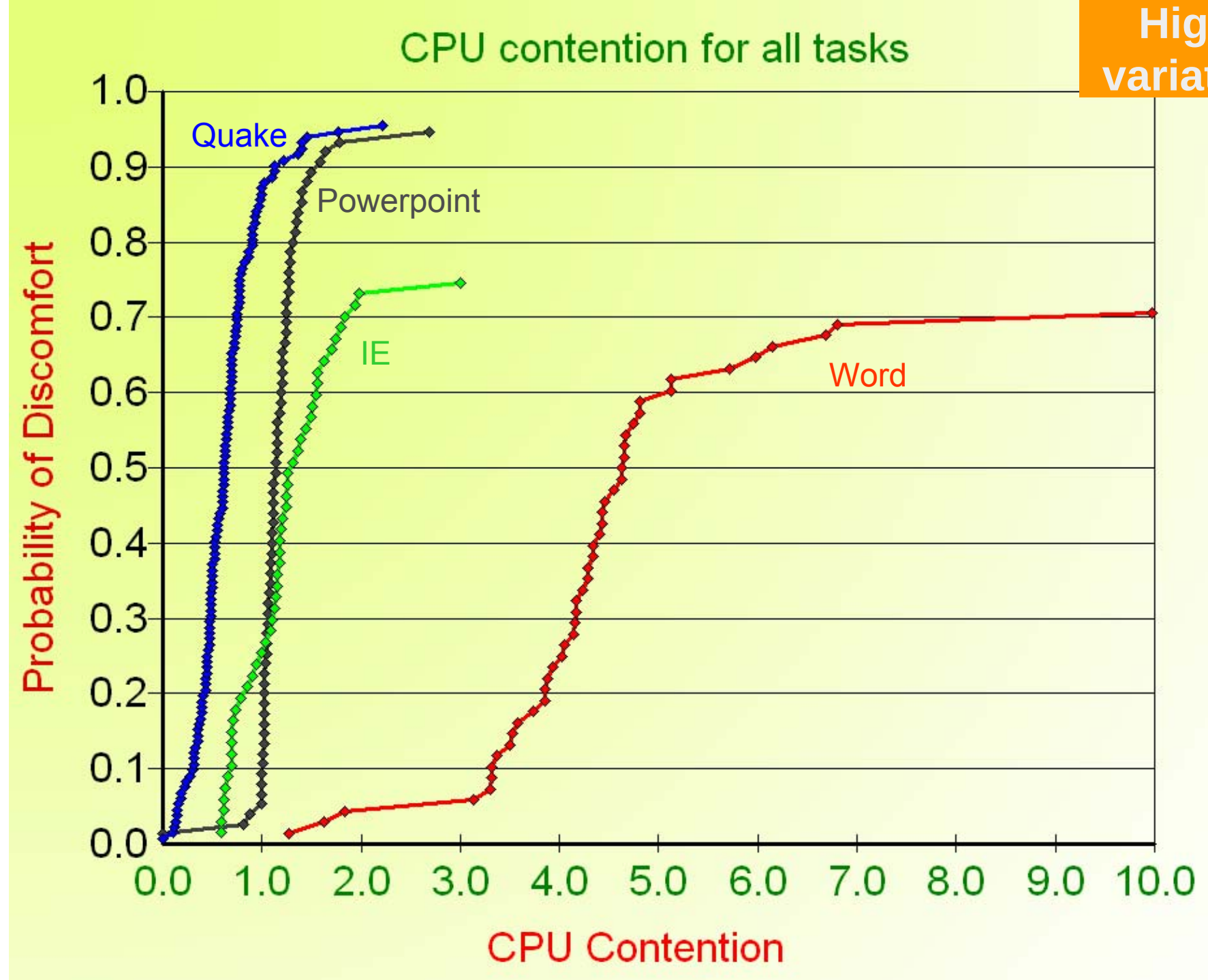
- CDFs for discomfort contention level
- Not all contentions cause discomfort: exhausted region

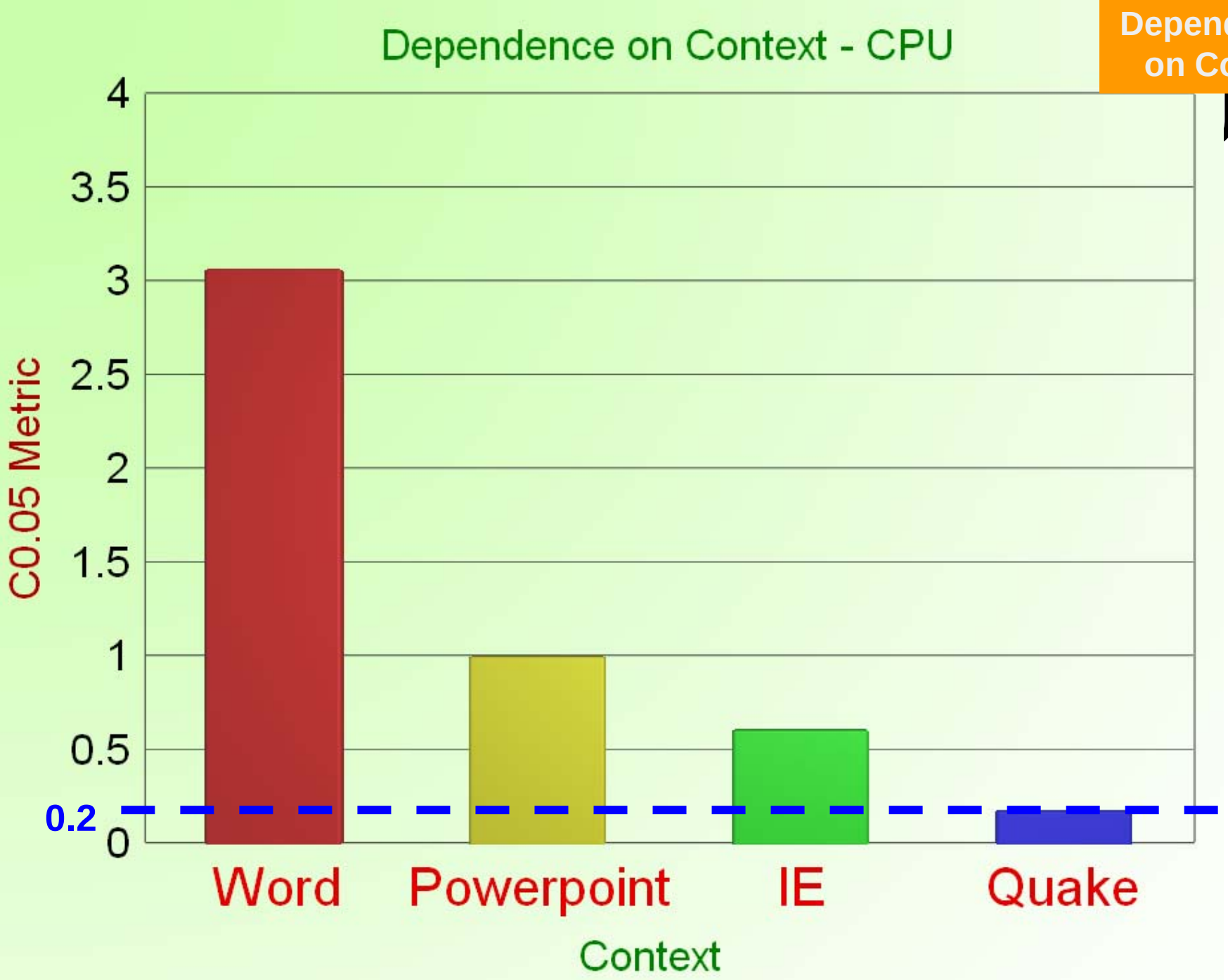
- CDFs allow us to read %age of people discomforted for a given contention
  - **Metric $c_{0.05}$** : At what contention do we discomfort only 5% of the people ?

High variation



- $C_{0.05}$ = 0.35 (aggregated over all applications)


CPU contention for all tasks
High variation
Quake
Powerpoint
IE
Word
Probability of Discomfort
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
10.0
CPU Contention


Dependence on Context - CPU
C0.05 Metric
4
3.5
3
2.5
2
1.5
1
0.5
0.2
0
Word
Powerpoint
IE
Quake
Context

Conclusion

- Resources needed to keep a user happy are highly dependent on the application and on the user
- Direct user feedback may be useful

  (per-user tailoring of resource usage)

Outline

- **Background**
  - Virtuoso
  - User diversity
    - User comfort with resource borrowing [Gupta & Lin, HPDC'04]
  - **Scheduling VM in Virtuoso**
    - **VSched [Lin, SC'05]**
- Direct user input in VM scheduling
- User study
- Conclusions

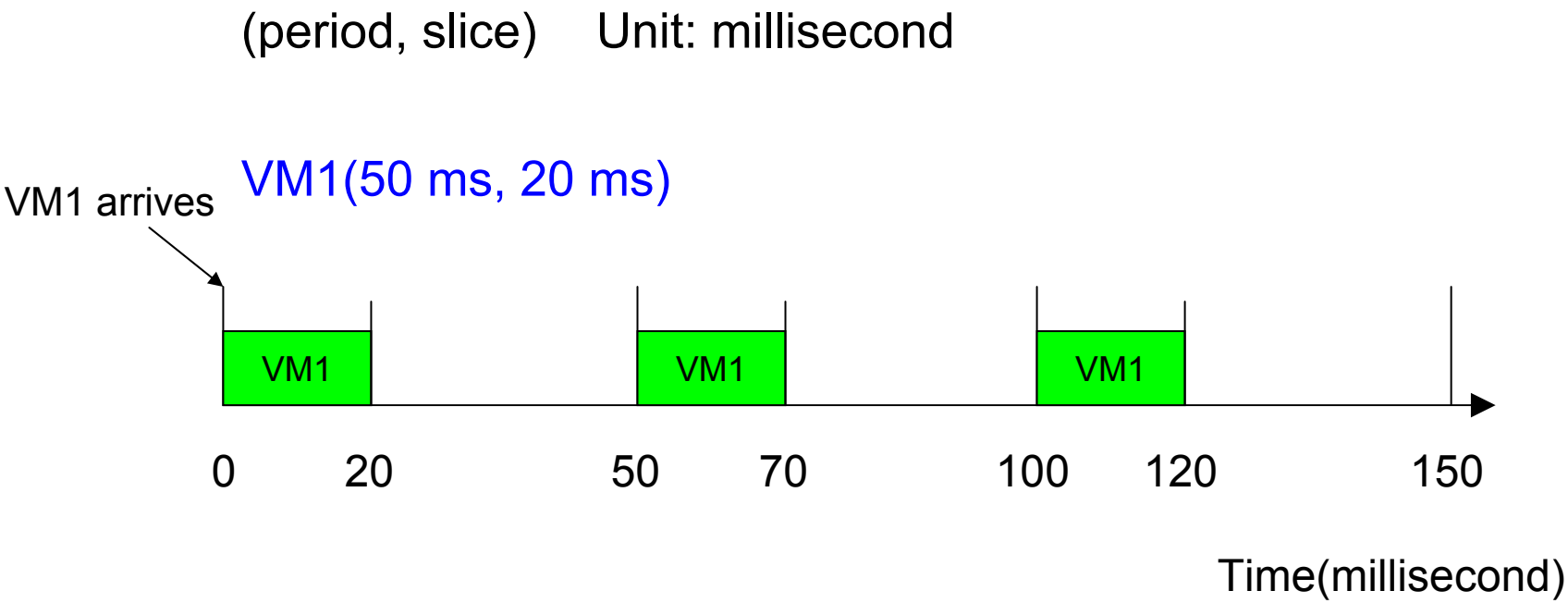
Challenges For CPU Reservations

- Resource providers price VM execution according to interactivity and compute rate constraints
  - How to express, validate, and enforce?
- A workload-diverse set of VMs
  - How to schedule them on a single physical machine?

Periodic Real-time Scheduling Model

- Task runs for slice seconds every period seconds

[JACM 1973]



Periodic Real-time Scheduling Model

- Task runs for slice seconds every period seconds
  - "1 hour every 10 hours", "1 ms every 10 ms"
    - Does NOT imply "1 hour chunk" (but does not preclude it)
  - Compute rate: *slice / period*
    - 10 % for both examples, but radically different interactivity!
  - Completion time: *size / rate*
    - 24 hour job completes after 240 hours
- Unifying abstraction for diverse workloads
  - We schedule a VM as a single task
  - VM's (slice, period) enforced

Implementation - VSched

- Provides soft real-time (limited by Linux)
- Runs at user-level (no kernel changes)
- Schedules any set of processes
  - We use it to schedule VMs (Type II VMM)
- Supports very fast changes in constraints
  - We know immediately whether performance improvement is possible or if VM needs to migrate

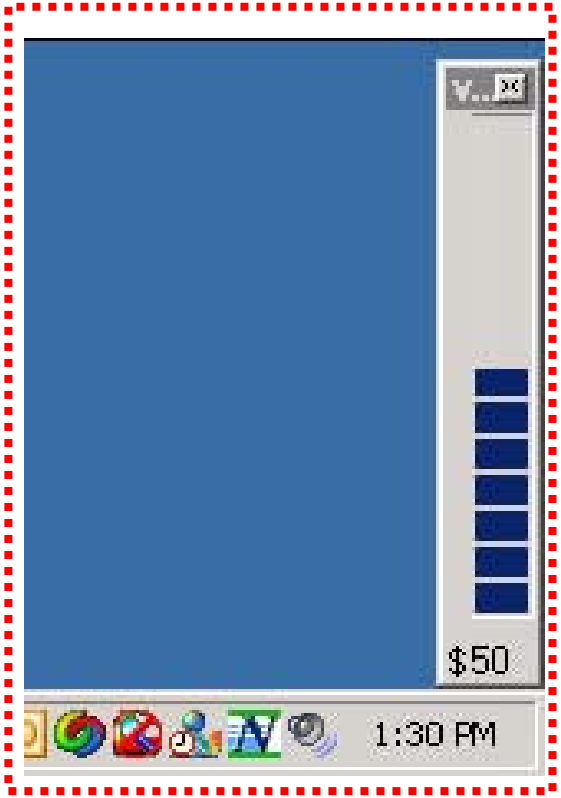
Outline

- Background
  - Virtuoso
  - User diversity
    - User comfort with resource borrowing [Gupta & Lin, HPDC'04]
  - Scheduling VM in Virtuoso
    - VSched [Lin, SC'05]
- **Direct user input in VM scheduling**
- **User study**
- Conclusions

How to choose the right (*period*, *slice*)

- Possible non-intrusive interface
  - Unused until the user is unhappy with performance
  - Instantly manipulated to change the schedule
  - GUI (showing cost)
  - Non-centering joystick

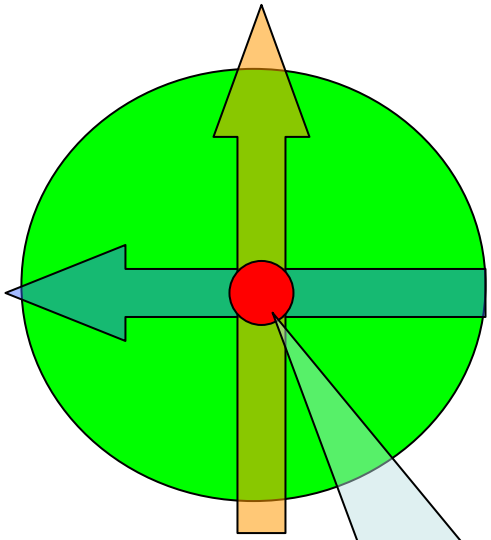
Interfaces



$10
Non-centering joystick

$250

Two-dimension mapping



Starting point,
500 ms period, 50%
utilization

$10

Specific cost function used

$$cost = 100 \times \left(\frac{slice}{period} + \beta \times \frac{overhead}{slice} \right)$$

- Overhead: time to execute scheduling core.
- as *slice* declines, more time spent in VSched & kernel on behalf of the process.

User study

- 18 users
- User used Windows VM for Word processing, presentation creation, web browsing, and game playing

- Can end-users use our interface to find schedules for their interactive VMs that were comfortable?
- Can users trade off between cost and comfort using the interface?

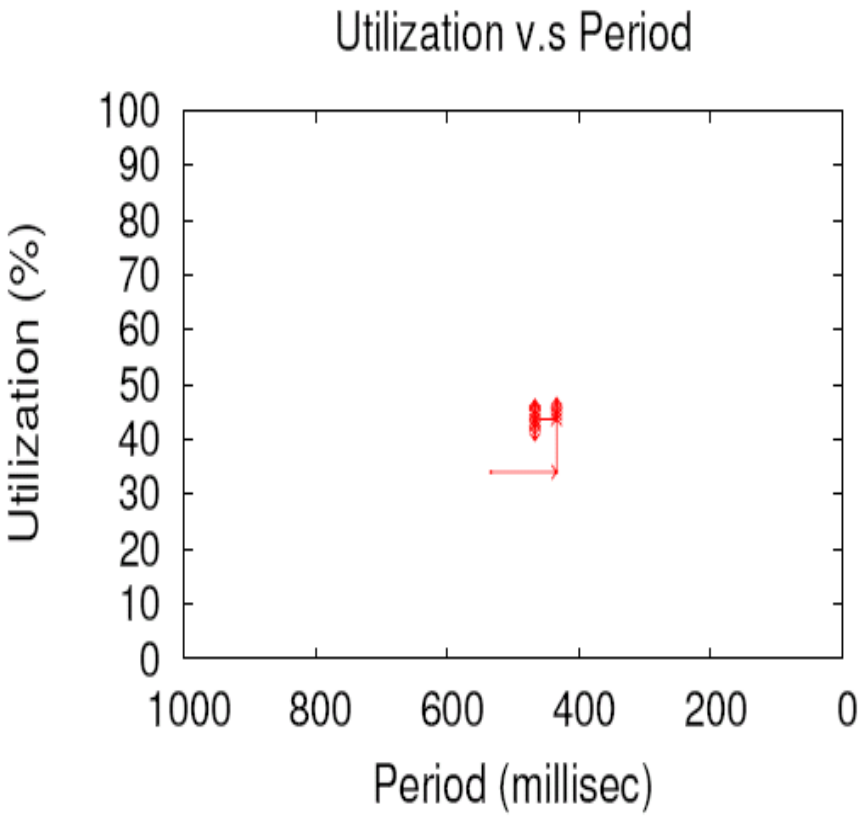
Testbed

- Dell Optiplex GX270 (2 GHz P4, 512 MB, 80 GB, 100 mbit Ethernet)
- VMware GSX Server 3.1
- VSched server running as daemon
- VM running WinXP Pro
- $10 joystick

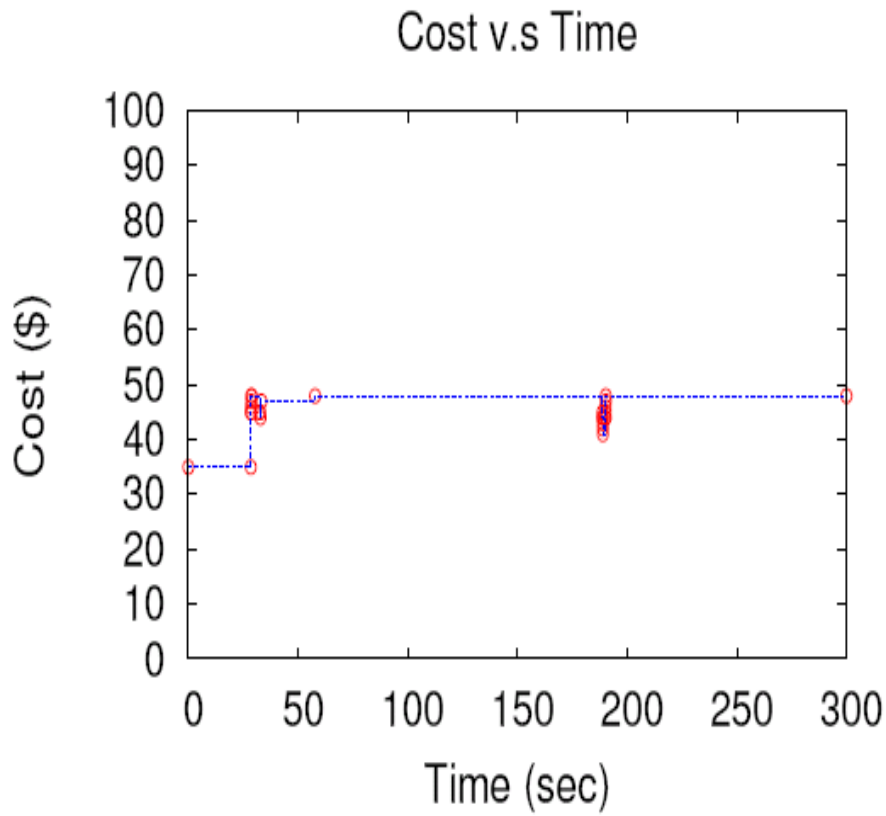
Process

- Adaptation Phase I (8 mins): VM
- Adaptation Phase II (5 mins): Control
- 4 tasks (Word, Powerpoint, IE, Quake II)
- 3 subtasks (5 mins) per task
  - Comfort
  - Comfort and cost
  - Comfort and cost with perceived external observation
- Video-taping (mild deception, a common technique in psychological research)
- Questionnaire per subtask

User A: Tracks, cost versus time (Word)

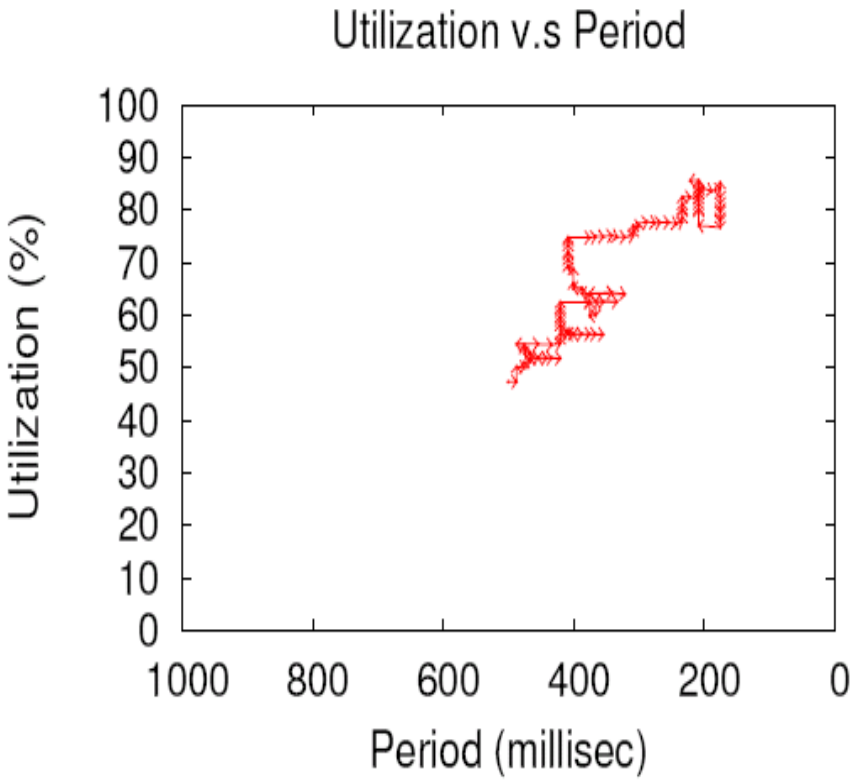


(a) Word (track)

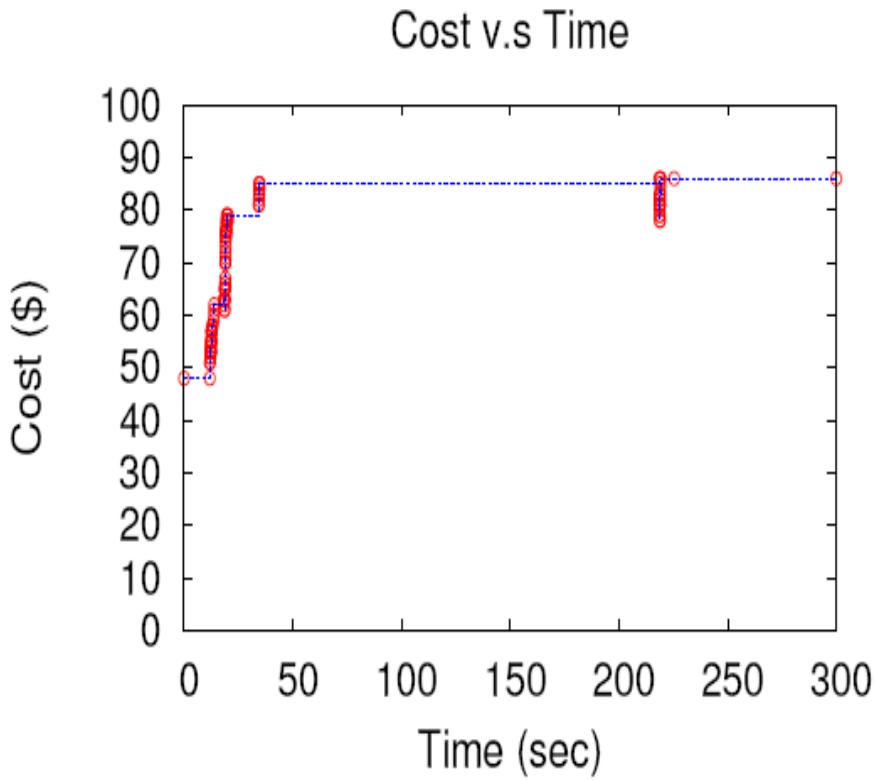


(b) Word (cost v. time)

User A: Tracks, cost versus time. (Game)

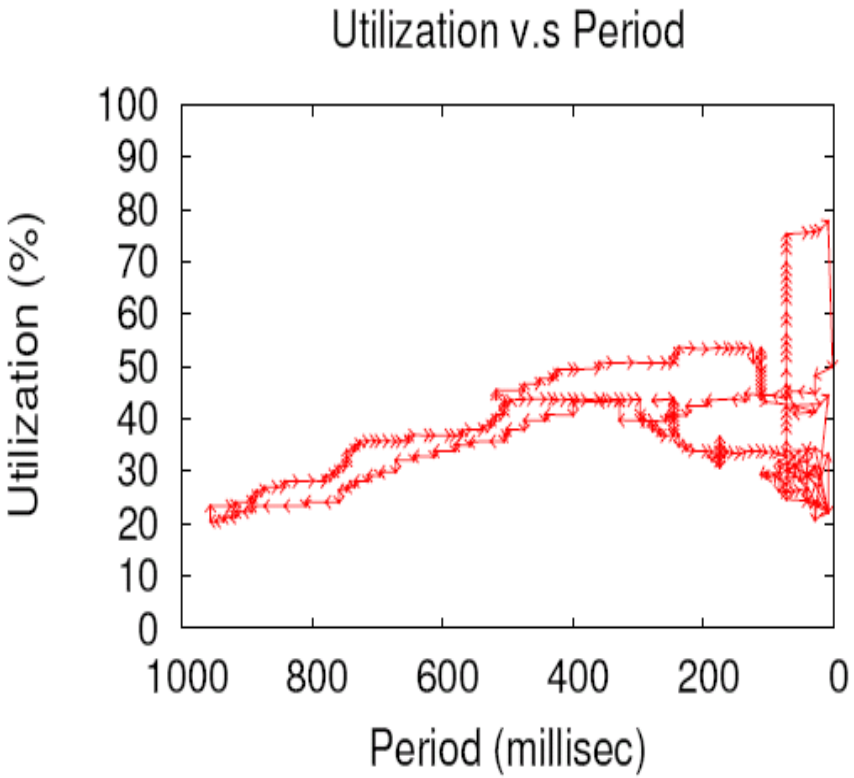


(g) Game (track)

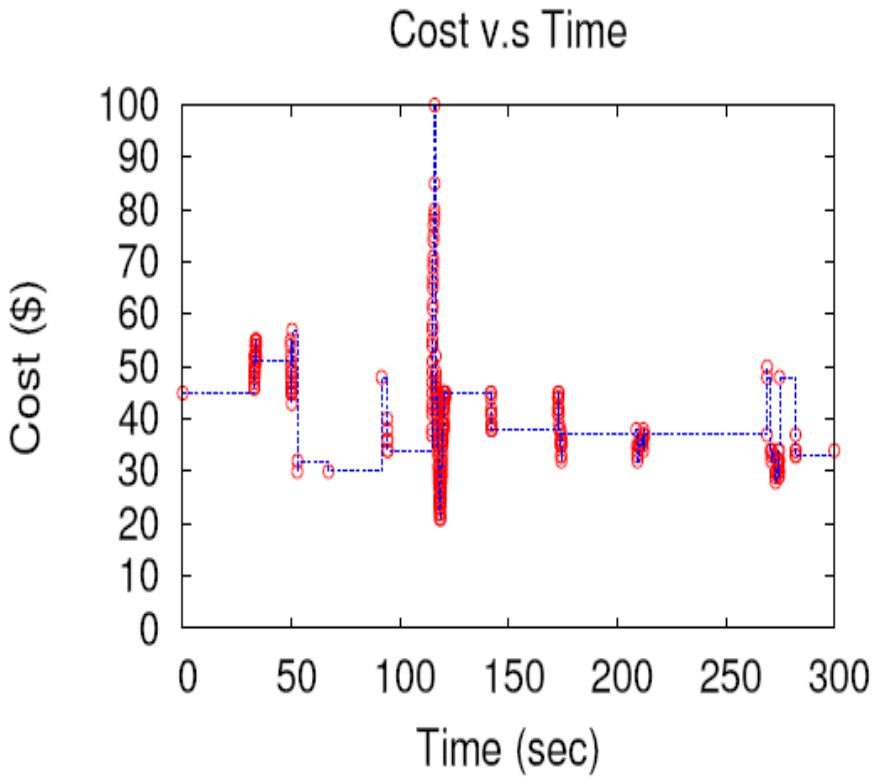


(h) Game (cost v. time)

User B: Tracks, cost versus time (Word)

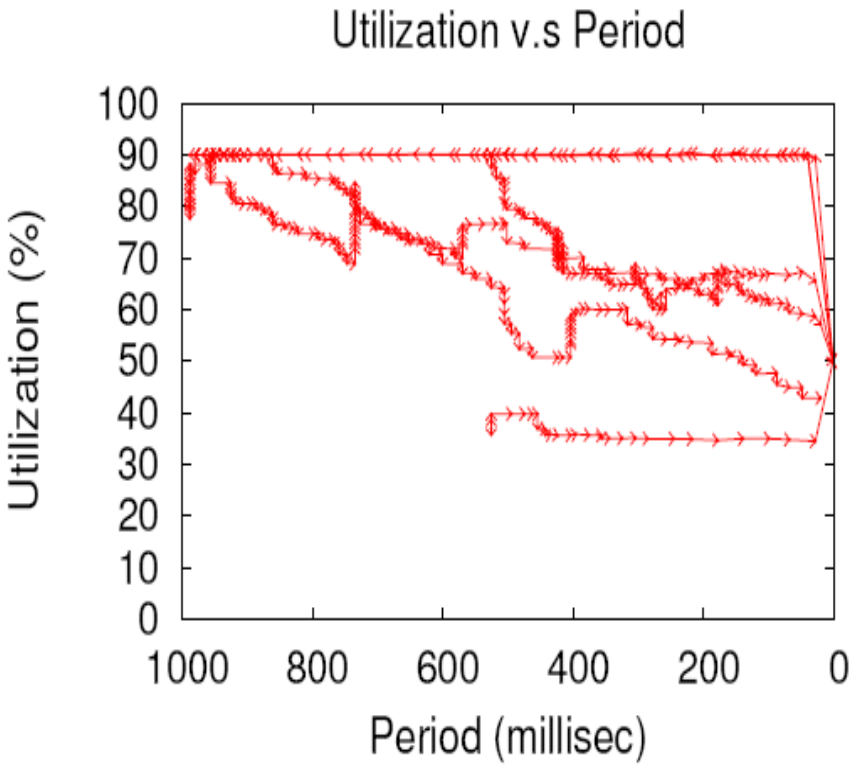


(a) Word (track)

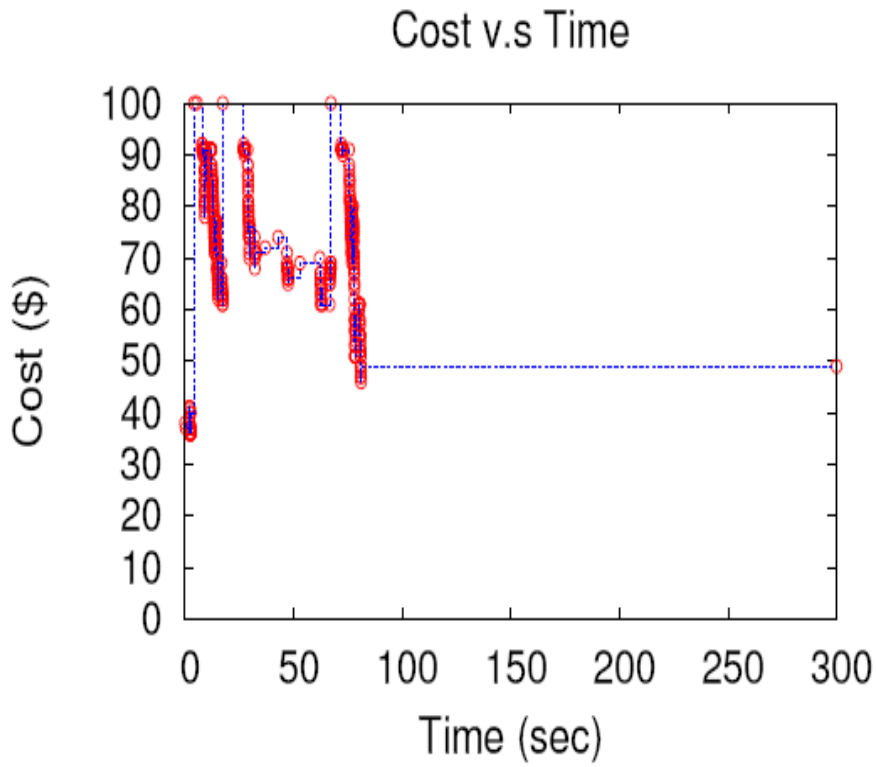


(b) Word (cost v. time)

User B : Tracks, cost versus time. (Game)



(g) Game (track)



(h) Game (cost v. time)

Example questions

- Did you find that the joystick control was understandable in this application? (Y/N)
- Were you able to find a setting that was comfortable? (Y/N)
- If yes, what's the cost?

Task	Sub-task	Avg	Std	Min	Max
Word	II Comfort+Cost	46.0	**20.4**	19	86
	III Comfort+Cost+Ext	48.4	**20.7**	19	84
Powerpoint	II Comfort+Cost	52.4	**19.5**	20	91
	III Comfort+Cost+Ext	52.3	**19.2**	18	87
Web	II Comfort+Cost	49.6	**22.7**	15	90
	III Comfort+Cost+Ext	50.2	**23.3**	16	87
Game	II Comfort+Cost	78.8	**14.1**	50	93
	III Comfort+Cost+Ext	76.5	**14.9**	49	91

LOWEST COST

Results of the user study

- ✓ ≥ 89% of users understood our control mechanism
- ✓ ≥ 72% of users could use it to find a comfortable position
- ✓ ≥ 78% of users could use it to find a comfortable position that they believed was of lowest cost

(Providing 95% confidence intervals)

Results of the user study

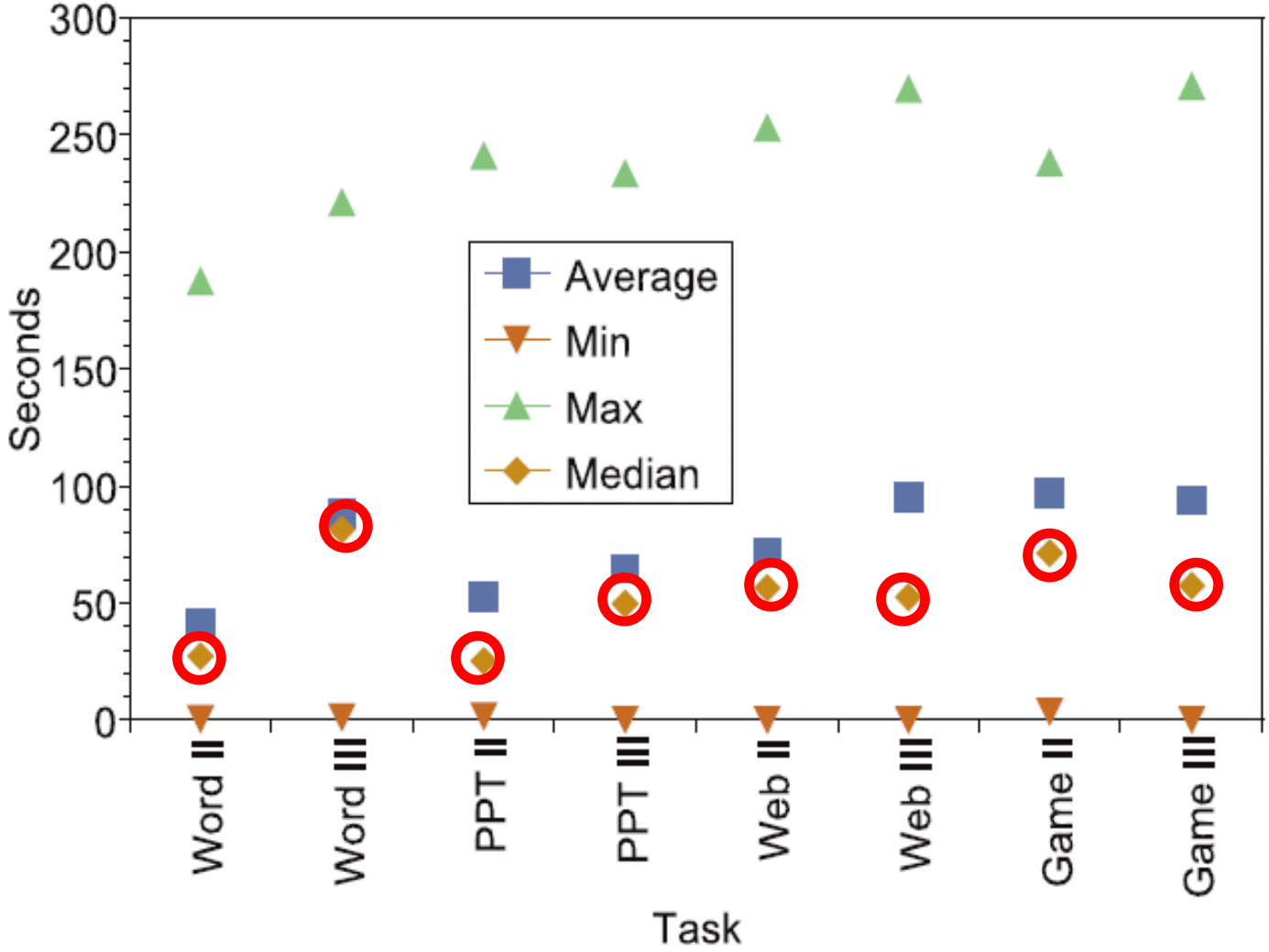
- ✓ ≥ 89% of users understood our control mechanism
- ✓ ≥ 72% of users could use it to find a comfortable position (other 28%)
- ✓ ≥ 78% of users could use it to find a comfortable position that they believed was of lowest cost (other 22%)
- In both cases, numbers result from one user answering the question unintelligibly.

(Providing 95% confidence intervals)

Results of the user study *(cond.)*

- Costs on average increase for applications with increasingly finer grain interactivity.
- Tremendous variation in acceptable cost among the users.
- Almost all users were able to find a setting that gave them comfortable performance.

Duration to first encounter of lowest cost



Results of the user study *(cond.)*

- Median time for the user to find the setting of lowest cost that is comfortable for him is in the range from 25-80 seconds (it includes use of the application).
- Time between further interactions decline as the user is more familiar with the app/system combination.

Conclusion of this work

- Using VSched's joystick control, even a *naive* user can *quickly* guide the system to a schedule that simultaneously optimizes both for his comfort in using an application and for low cost.
- System can run more interactive users simultaneously, or allocate more time for long-running batch VMs.

Power Control in Modern Processors

- In-submission work by Lin, Mallik, Dinda, Memik, Dick
  - Tech report available from us
- User-driven Frequency Scaling (UDFS)
  - User presses button when annoyed with speed of computer
  - Button-press feedback drives model that drives frequency setting
  - Reduces measured system power up to **22.1%**, averaged across users and applications, compared to Windows XP DVFS (*Dynamic Voltage and Frequency Scaling*)
- Process-driven Voltage Scaling (PDVS)
  - Customize frequency to voltage mapping to individual processor at every temperature

Related work: direct user input

- Buttons as on-screen objects; encapsulated code to enable tailoring of applications **[Mclean, CHI'90; Dourish, ECSCW'99]**
- Weighted fair queuing allows users to explicitly weight each of their processes
- Microsoft Windows; user specify scheduling class of a process
- Unix systems provide the "nice" mechanism
- …
- Require user understand scheduler to get good results
- Easy for a user to live-lock the system

Take-away points

- Discovered high variation in user expectations of performance
- Developed interface that captures user variation for CPU scheduling in VM desktop replacement scenario
- Evaluated interface in extensive user study, finding it to be effective
- Currently extending direct user feedback model for other systems problems, including power management

Thank you!

- **Bin Lin**'s homepage: http://www.cs.northwestern.edu/~blin
- Bin Lin, Peter Dinda, *Putting the User in Direct Control of CPU Scheduling*, Tech Report NWU-EECS-06-07, EECS, Northwestern University
- Group project webpage:

  http://virtuoso.cs.northwestern.edu
- Presciencelab webpage:

  http://presciencelab.org