Energy Management of Virtual Memory on Diskless Devices

Jerry Hom Ulrich Kremer

Department of Computer Science

Rutgers University

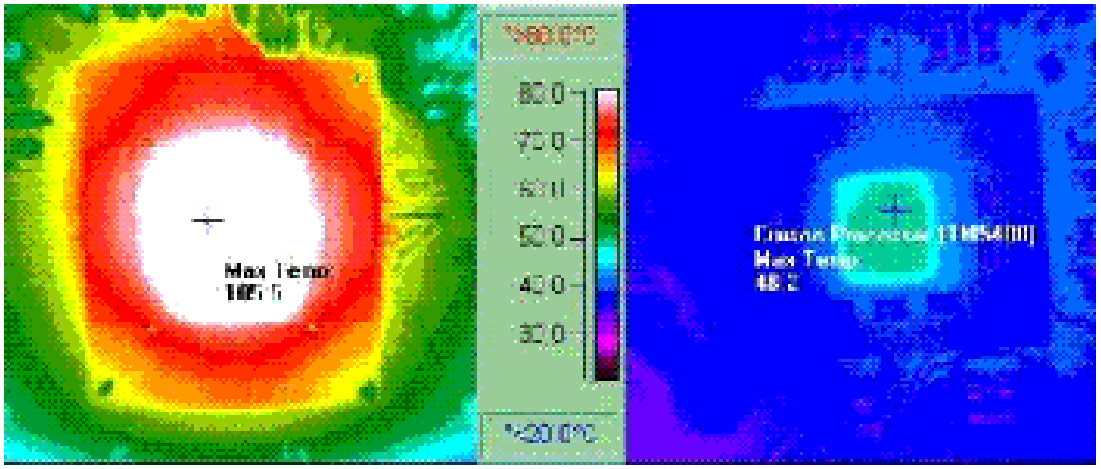
Energy **E**fficiency and **L**ow-Power Lab

This research is partially supported by NSF CAREER award CCR-9985050

Why Power/Energy Management?

- Prolong battery life
- Reduce heat dissipation
  - packaging costs and cooling
  - reliability

Pentium III processor **Transmeta's Crusoe TM5400**



Without cooling: 105C (221F) vs. 42C (118F)
running DVD application (Source: Transmeta's Crusoe processor white paper)

Why Compiler and not OS/Hardware?

Compiler:

- may know about "future" program behavior through aggressive, whole program analysis.
- can reshape program behavior through code transformations and thereby enable optimizations.

Scenarios:

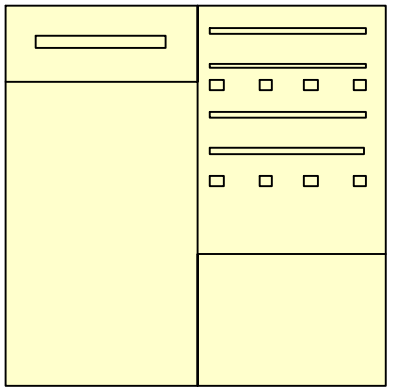
- single process environment: compiler-directed power and energy management is directly "executed" by underlying OS/HW.
- multiple process environment: power/energy application profile is used by OS to make scheduling decisions.

Platform/Resource(s)?

Handheld PC

- "Workstation" class machine
- Depends on battery power
- Wireless communication
- No disk (limited resources)

Pervasive Computing Environment



Handheld PC

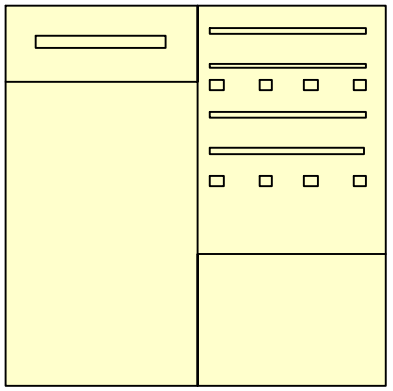
- "Workstation" class machine
- Depends on battery power
- Wireless communication
- No disk (limited resources)

Desktop / Server

- One order of magnitude more resources than handheld

Pervasive Computing Environment

Which Resource(s) to Manage?



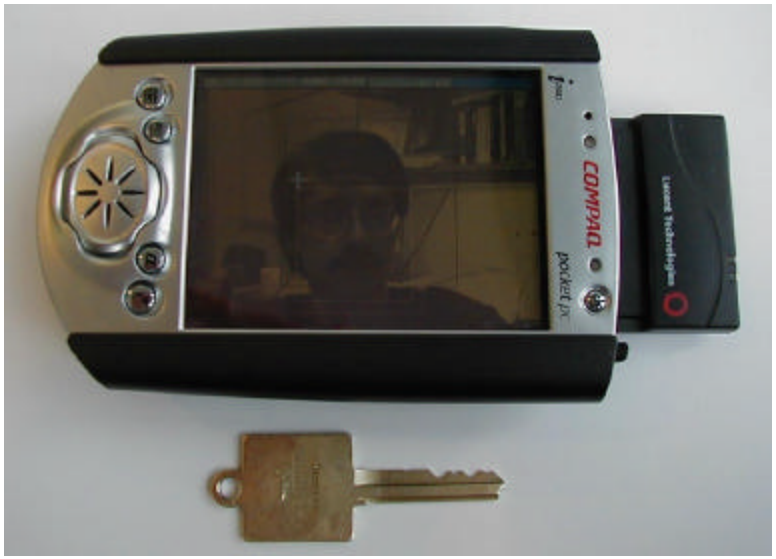
Handheld PC

- "Workstation" class machine
- Depends on battery power
- Wireless communication
- No disk (limited resources)

Desktop / Server

- One order of magnitude more resources than handheld

Handheld PC: iPAQ H3650



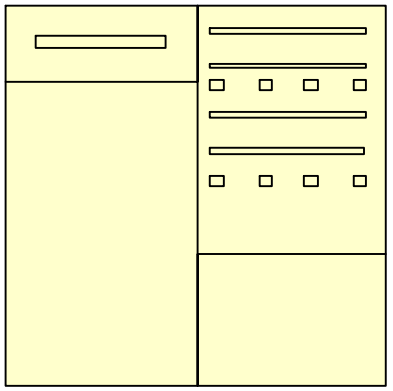
- 206MHz SA1110, dynamic frequency scaling 59MHz - 206MHz 16KB I-cache, 8KB D-cache (L1)
- 66MHz 32/64MB SDRAM, 16MB Flash
- USB and serial port through cradles; PCMCIA and Compact flash through sleeves; light sensor, microphone, thin film transistor (TFT) color display
- 940mAh lithium polymer battery
- mobile “Linux” box
- **1.25W** + **0.95 W** for wireless

Significant portion of overall energy budget is due to wireless communication (WaveLAN > 40%)

Pervasive Computing Environment

What Applications?

- Dynamic
- Static (regular)



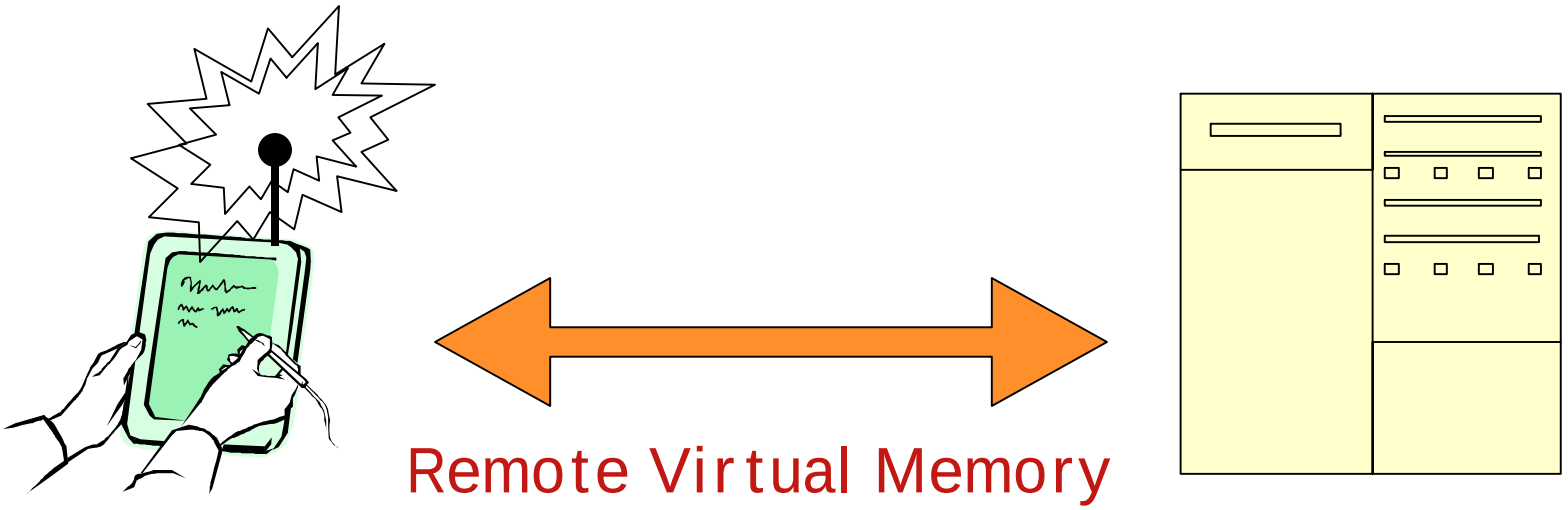
Handheld PC

- "Workstation" class machine
- Depends on battery power
- Wireless communication
- No disk (limited resources)

Desktop / Server

- One order of magnitude more resources than handheld

Pervasive Computing Environment



Handheld PC

- "Workstation" class machine
- Depends on battery power
- Wireless communication
- No disk (limited resources)

Desktop / Server

- One order of magnitude more resources than handheld

Benefit Analysis

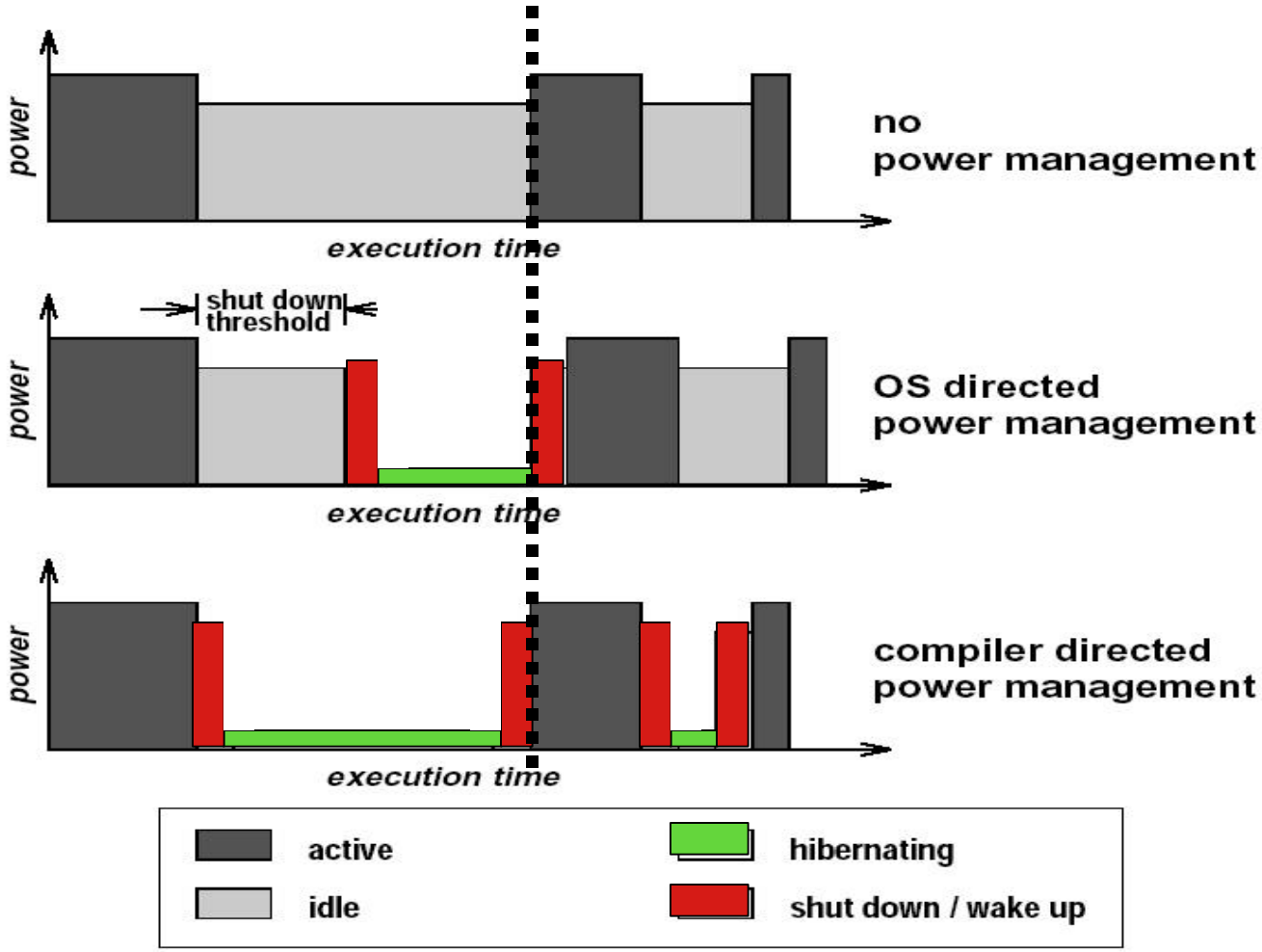
Compiler support for
communication card hibernation

- Regular applications, array based; will be part of program mix on handheld PCs
- Choice of problem and memory sizes to illustrate potential benefits
- Three programs with different page fault profiles:
  - adi: infrequent
  - shal: frequent
  - tomcatv: intermediate frequency

Talk Outline

- Virtual Memory through Wireless Communication
  - Hibernation (deactivation + activation)
  - Basic compilation strategy
  - EEL_{RM} prototype compiler
  - Experimental results
- Related Work
- Summary and Future Work

Threshold based OS vs. Compiler Directed Hibernation



Basic Compilation Strategy

Phase 1: Region analysis
- granularity of program regions for card hibernation
- for each region: card "ready" or "free to hibernate"

Phase 2: Reshape analysis
- page fault clustering $\Rightarrow$ move page faults to region entry
- loop index set splitting $\Rightarrow$ adjustment of granularity

Phase 3: Hibernate/activate instruction generation
- use performance prediction to
  (a) avoid hibernation if closely followed by activation
  (b) activate just-in-time to avoid performance penalty

EEL_{RM} Prototype Compiler

Phase 1: Region analysis

- Regions: inner loop nests (phases) or system calls (printf)
- For each region compute REF sets (entire data objects; future: DAD representation)
- Determine for each regions what data objects / code will be in memory; simulate LRU page replacement policy by "walking" over RCFG.
- Mark regions as "ready" or "free to hibernate"

EEL_{RM} Prototype Compiler

Phase 2: Reshape analysis

Page fault clustering (hand simulated)

Phase 3: Hibernate/activate instruction generation

No performance prediction ⇒

- no just-in-time card activation, i.e., either on demand or when reaching "ready" region
- card hibernation forced if region is marked as "free to hibernate"

EEL_{RM} : Experimental Results

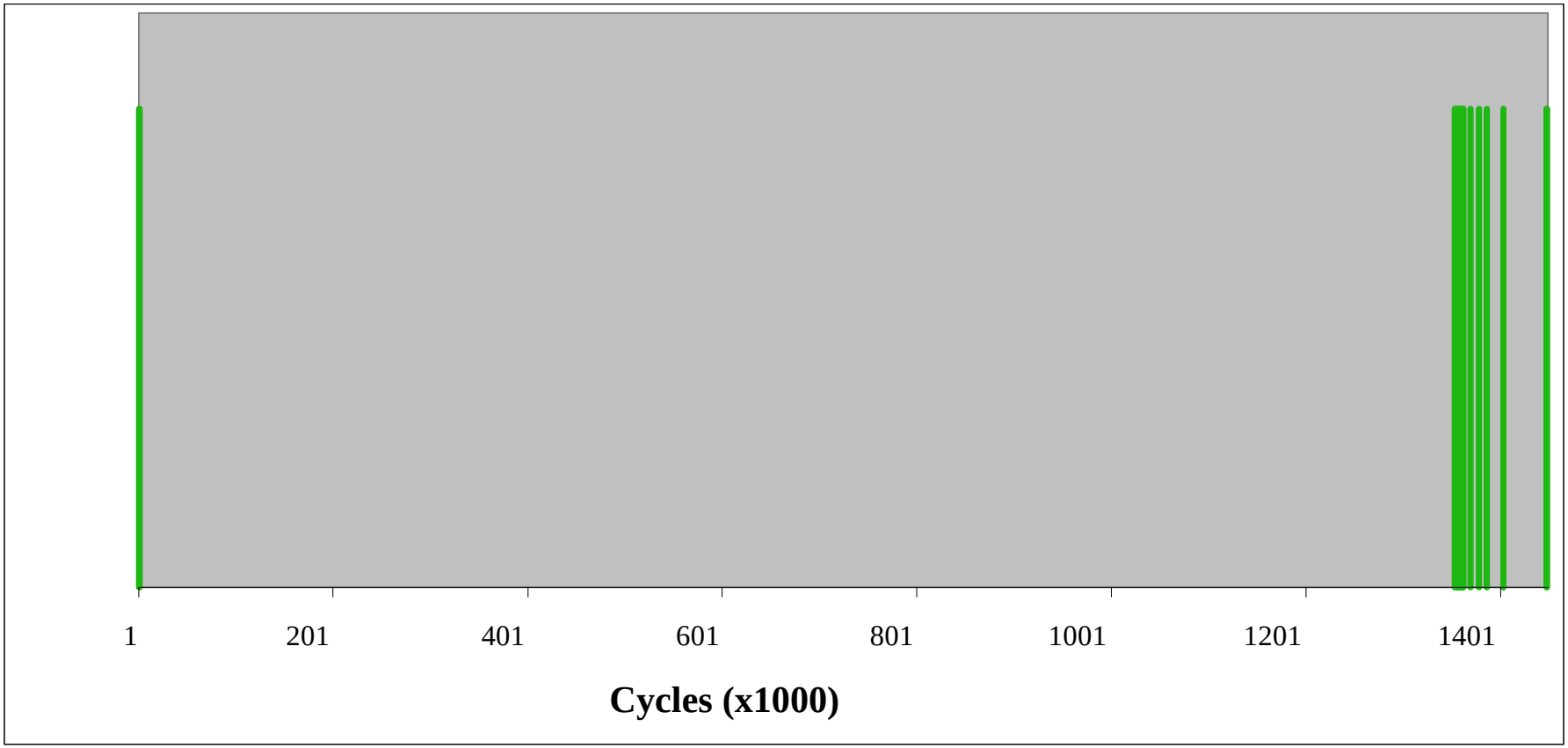
***N*: array dimension length (float)**
***M*: # of 4KB memory pages**

Parameters	*shal*	*adi*	*tomcatv*
N	32	16	32
M	32	16	16

- Prediction accuracy of page faults at region granularity; based on modified SimpleScalar simulator
- Comparison with threshold based OS techniques
  - relative energy savings
  - performance penalties (SimpleScalar)

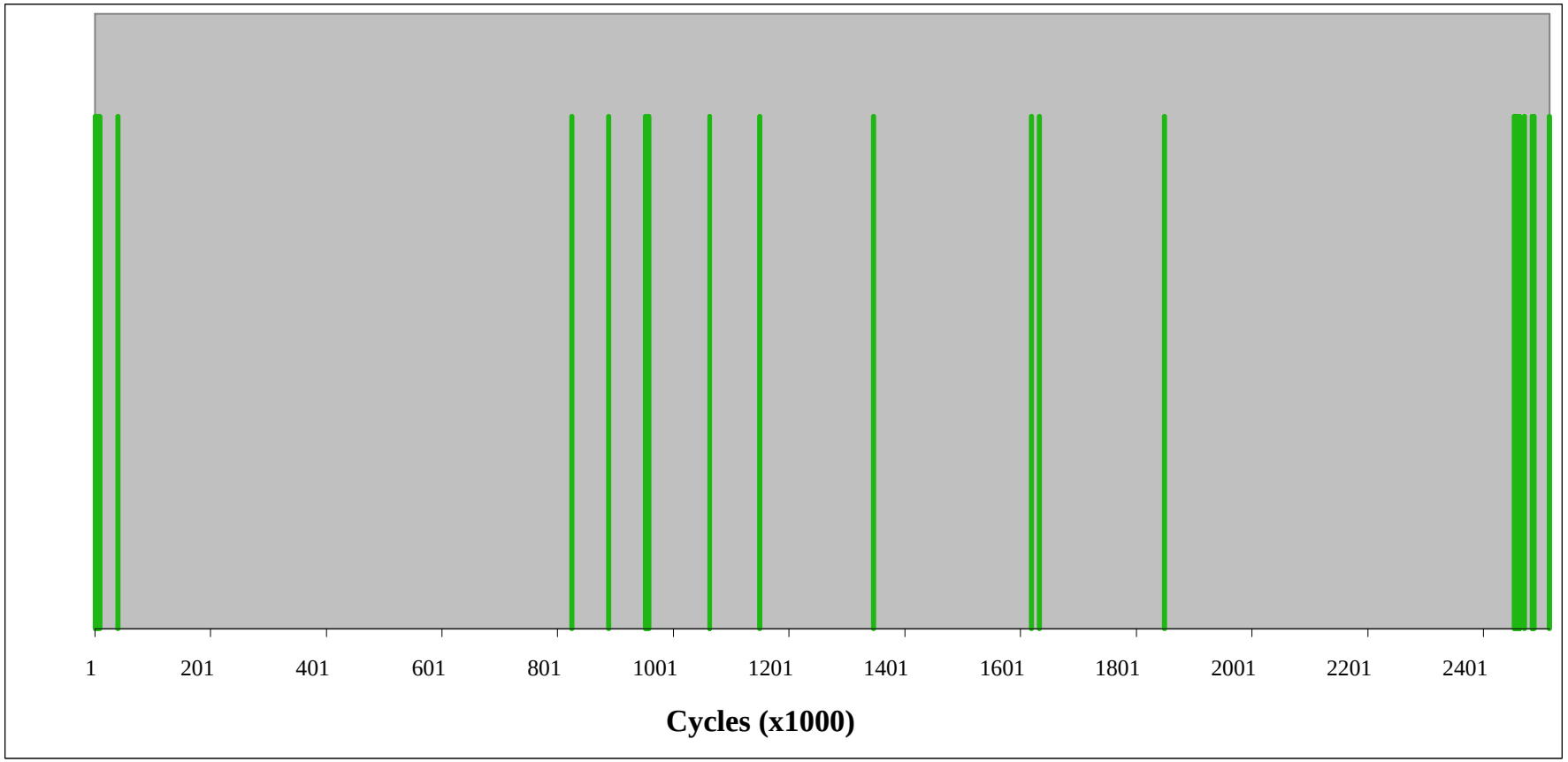
EEL_{RM} : Experimental Results

dynamic page faults for adi



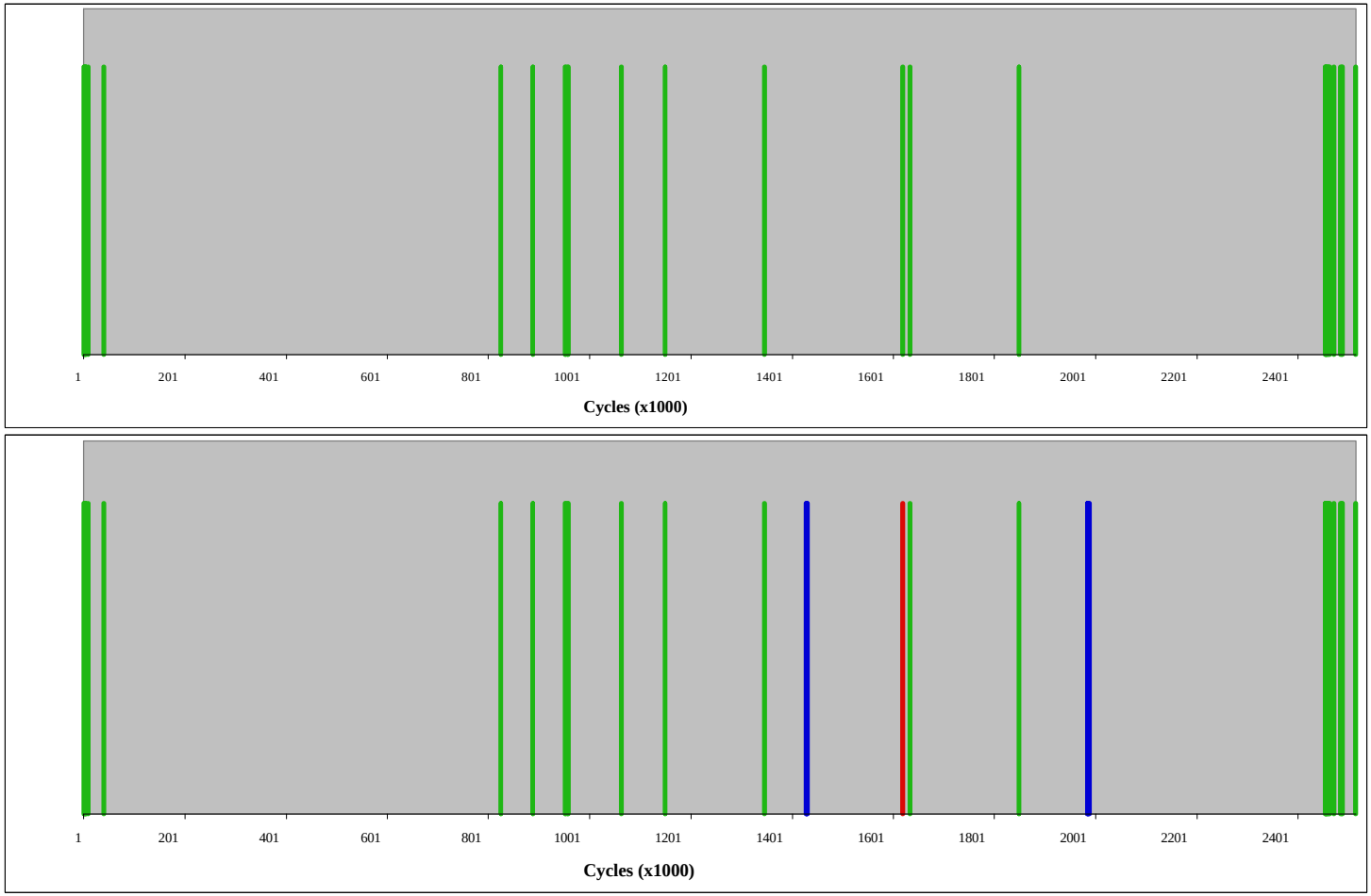
EEL_{RM} : Experimental Results

dynamic page faults for shal



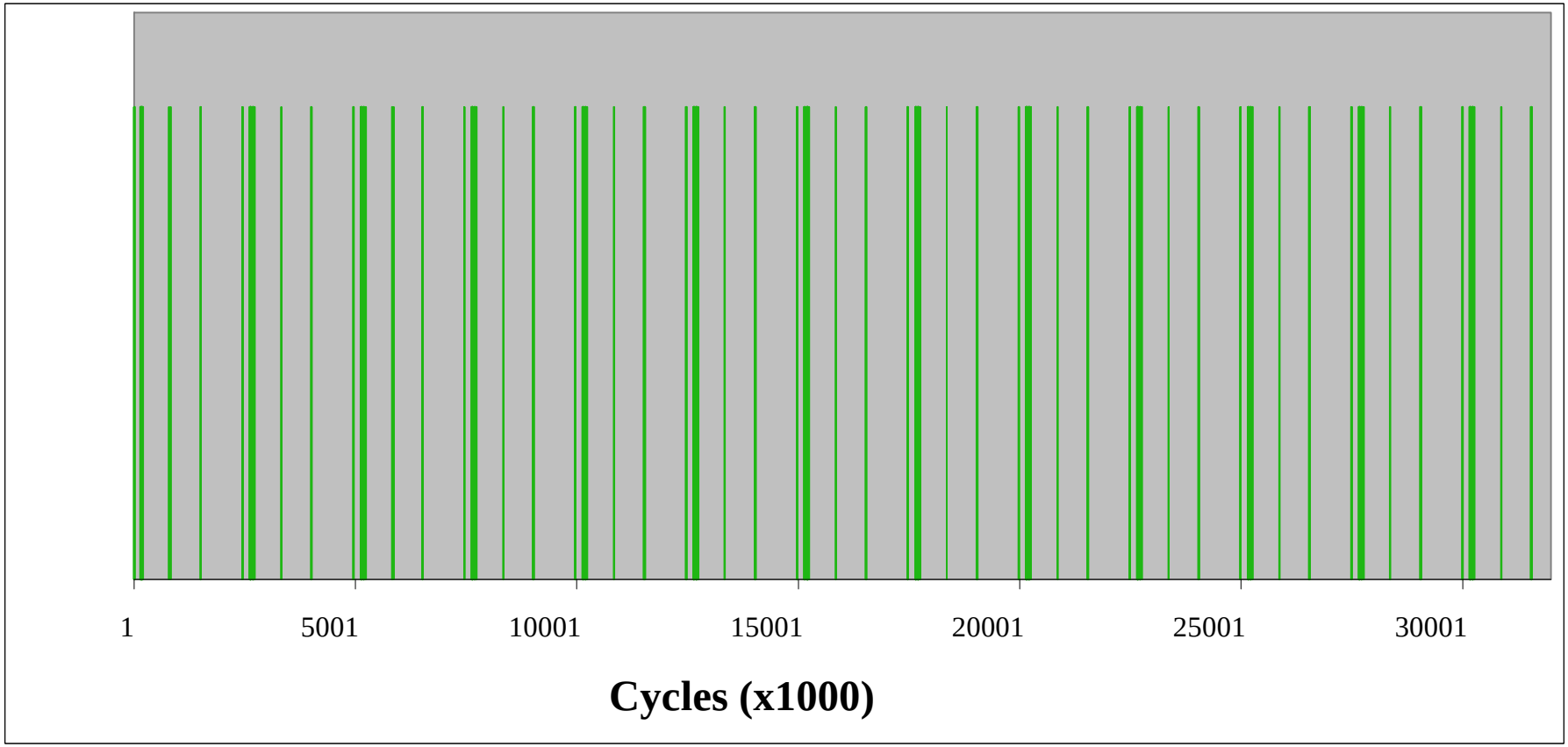
EEL_{RM} : Experimental Results

dynamic page faults for shal



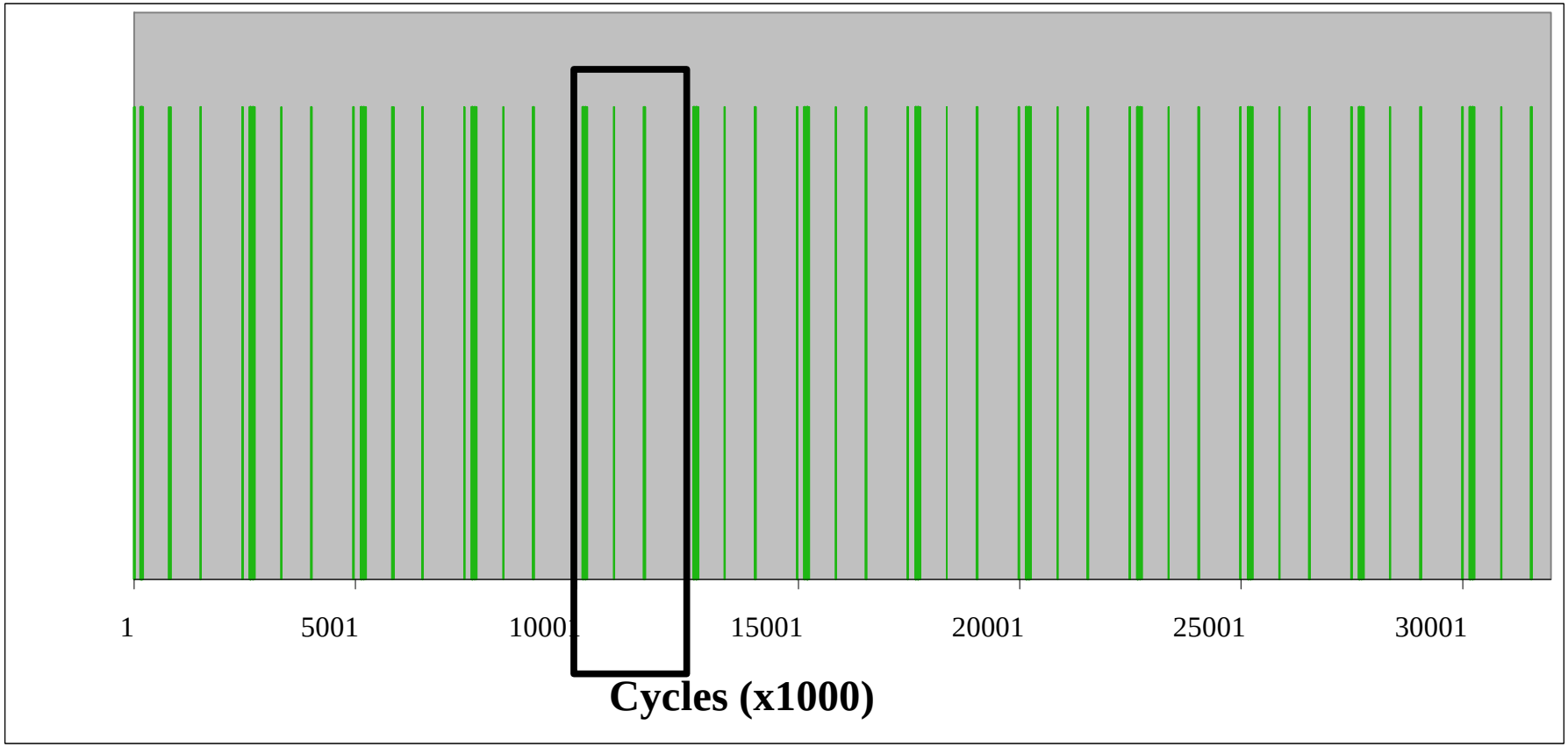
EEL_{RM} : Experimental Results

dynamic page faults for tomcatv



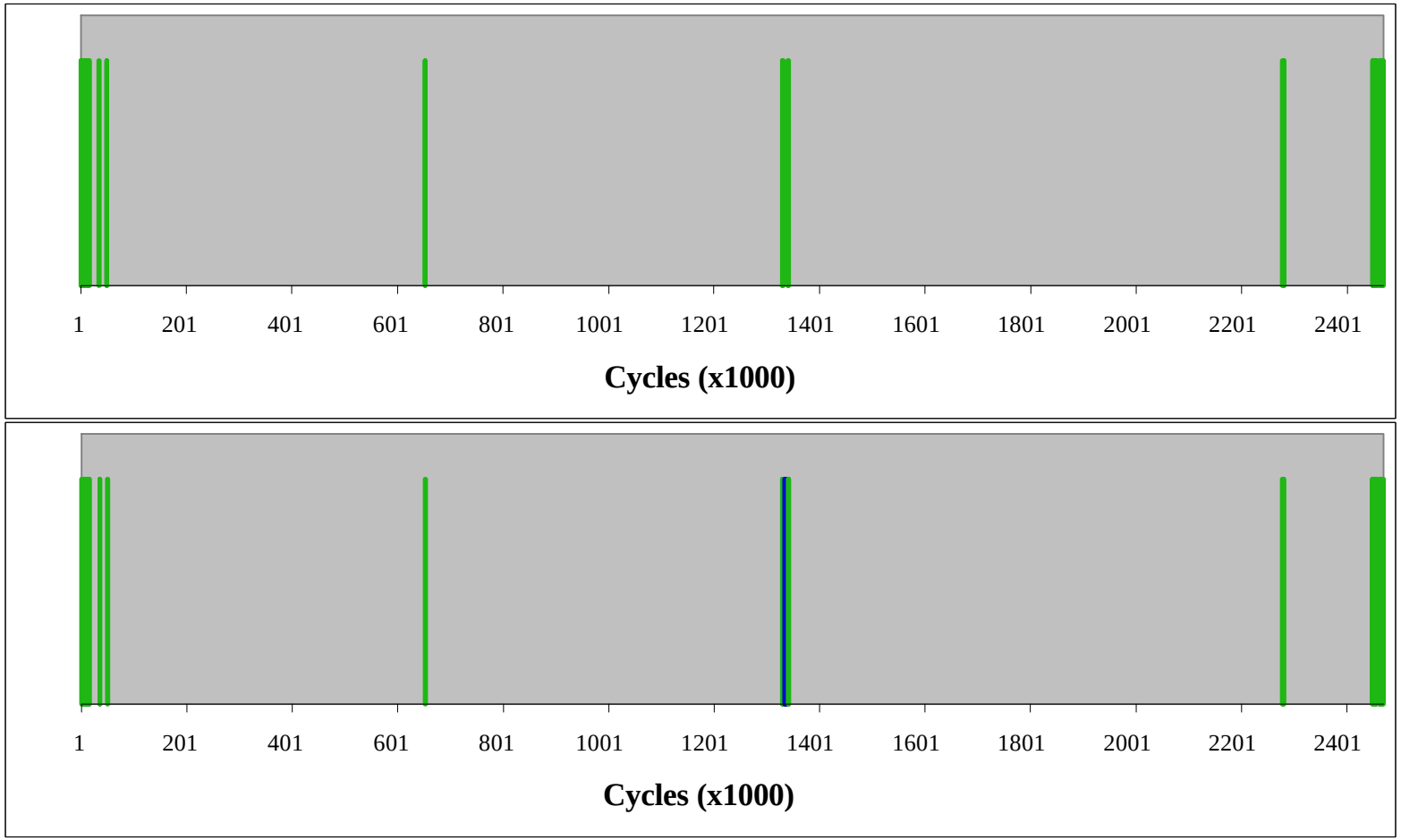
EEL_{RM} : Experimental Results

dynamic page faults for tomcatv



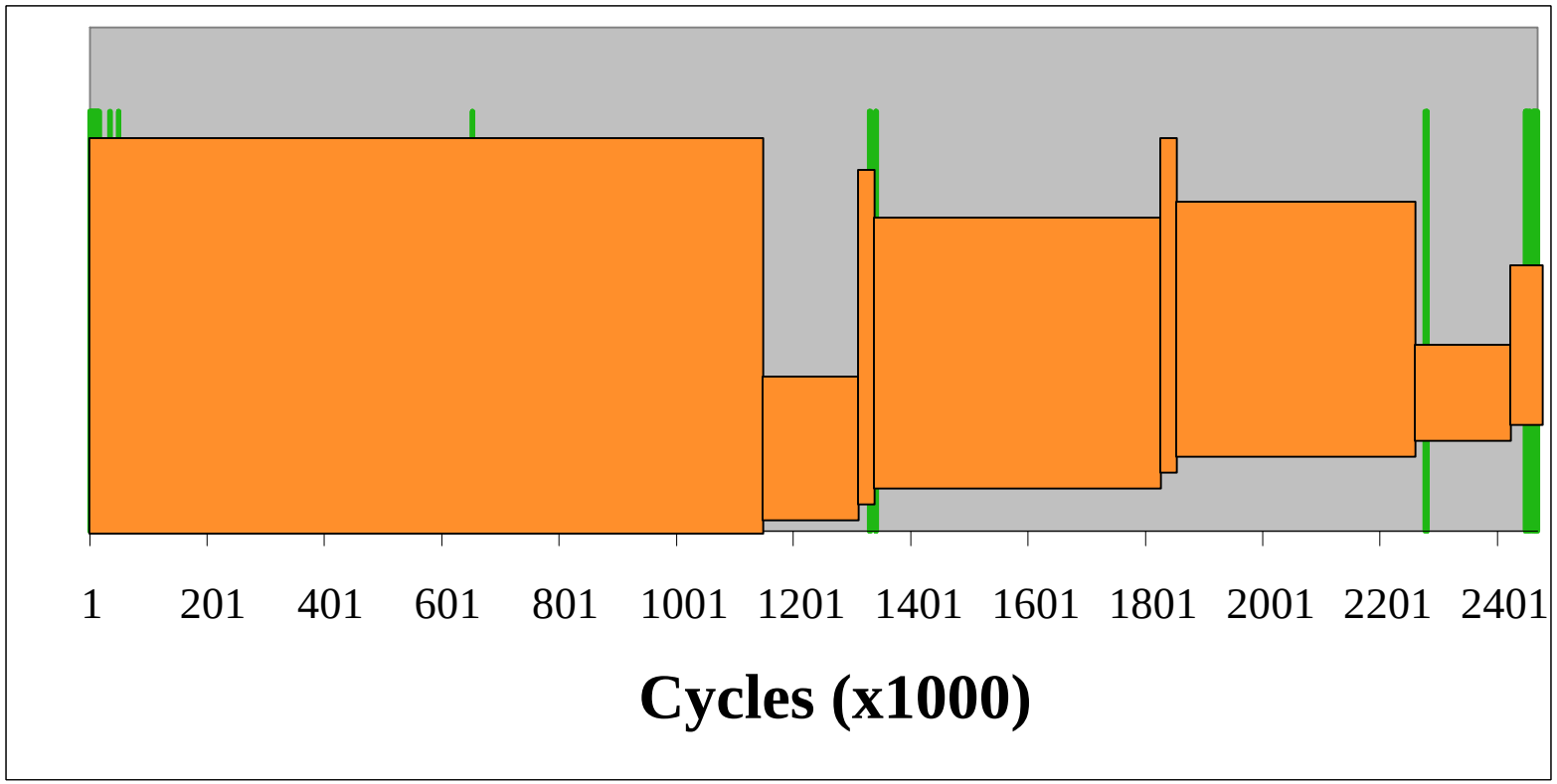
EEL_{RM} : Experimental Results

dynamic page faults for tomcatv



EEL_{RM} : Experimental Results

dynamic page faults for tomcatv



Relative Energy Savings vs. threshold based OS techniques

OS threshold	EEL_{RM} Energy Results			
	shal	*adi*	*tomcatv*	*tomcatv (PFC)*
1×	101.0	99.3	126.5	95.3
10×	100.1	92.6	116.3	87.6
20×	99.7	86.2	104.2	78.5
24×	**99.4**	—	—	—
30×	99.7	80.6	98.6	74.3
35×	99.7	78.1	**96.7**	**72.9**
54×	99.7	**69.1**	96.7	72.8
∞	99.7	71.3	96.7	72.8

Relative Performance Penalties vs. Card Always On

OS threshold	OS and EEL_{RM} Performance Results *shal*	*adi*	*tomcatv*
1×	101.3	101.7	105.4
10×	100.3	100.2	103.0
20×	100.2	100.2	102.9
24×	**100.3**	100.2	—
30×	100.0	100.2	101.0
35×	100.0	100.2	**101.5**
54×	100.0	**103.2**	100.0
∞	100.0	100.0	100.0
EEL_{RM}	100.2	101.7	101.0/103.9 (PFC)

Related Work (partial list)

Remote paging

- Comer and Griffioen: remote memory model
- Schilit and Duchamp: remote paging for thin clients
- InfoPad project at Berkeley: mobile client as I/O device

Dynamic power management:

- Simunic, Benini, Glynn, and De Micheli
- Devadas and Malik
- Lorch and Smith
- Macii, Pedram, and Somenzi

Summary and Future Work

- Our work is the first on compiler support for energy management of a wireless connection.
- Initial prototype systems based on SUIF2 infrastructure.
- For OS static inactivity thresholds between 10x and 20x card suspension overhead, we showed energy savings of up to 21.5%.
- Performance penalty was at most 3.9% relative to program versions without any card hibernation.
- Reshape optimizations are important.

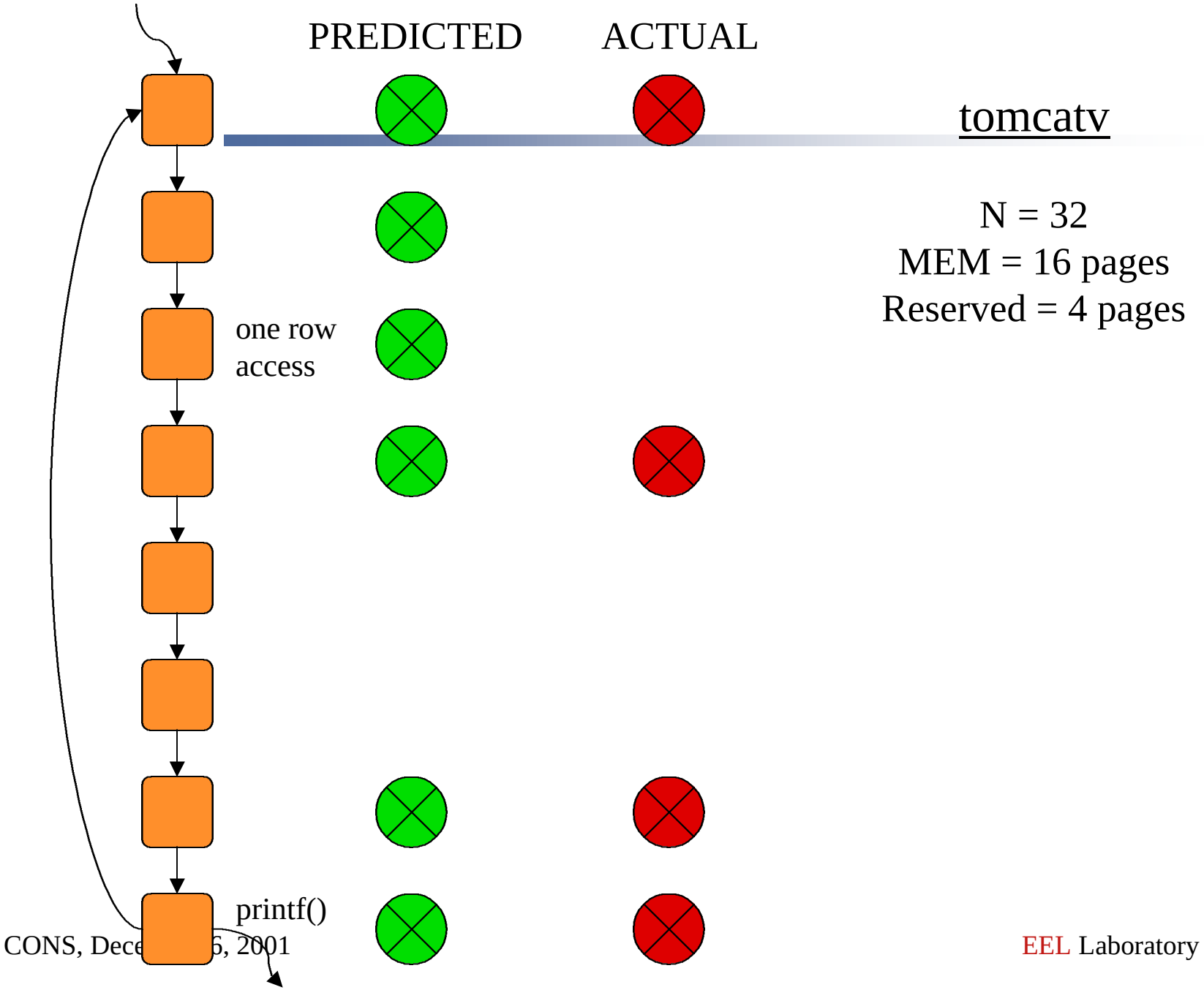
Summary and Future Work

- Extend techniques to irregular and dynamic programs.
- Investigate interactions between different power/energy management strategies.
- Include explicit I/O operations.

More Information

Energy **E**fficiency and **L**ow-Power Lab

http://www.cs.rutgers.edu/~uli/eel


PREDICTED
ACTUAL
tomcatv
N = 32
MEM = 16 pages
Reserved = 4 pages
one row
access
printf()

EEL_{RM} : Experimental Results

***N*: array dimension length (float)**
***M*: # of 4KB memory pages**

Parameters	*shal*	*adi*	*tomcatv*
N	32	16	32
M	32	16	16

	shal	*adi*	*tomcatv*
True Hit	17	62	304
True Miss	9	1	304
False Hit	1	0	2
False Miss	2	0	100

Different Ways to Save

Goal: Reduce the energy needed for executing an application with an execution time deadline constraint.

How to save power/energy?

- Dynamic resource configuration/hibernation
- Dynamic frequency/voltage scaling
- Remote task mapping
- Quality of Result (QoR)

Different Ways to Save

Goal: Reduce the energy needed for executing an application with an execution time deadline constraint.

How to save power/energy?

- Dynamic resource configuration/hibernation
- Dynamic frequency/voltage scaling [PACS'00], [LCPC'01], [ISLPED'01]
- Remote task mapping [COLP'00], [LCPC'01]
- Quality of Result (QoR)

Threshold based OS vs. Compiler Directed Hibernation