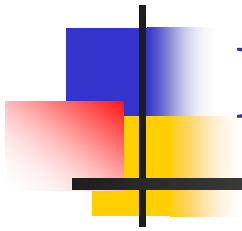
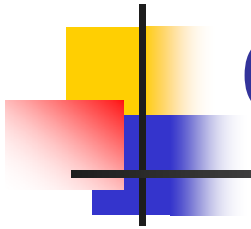


Power-Aware Operating Systems for Interactive Systems

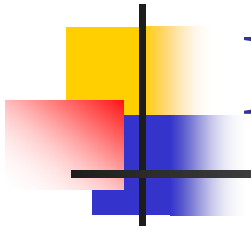
Yung-Hsiang Lu, Luca Benini, *Member*, *IEEE*, and Giovanni De Micheli, *Fellow*, *IEEE*

IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS



Outline

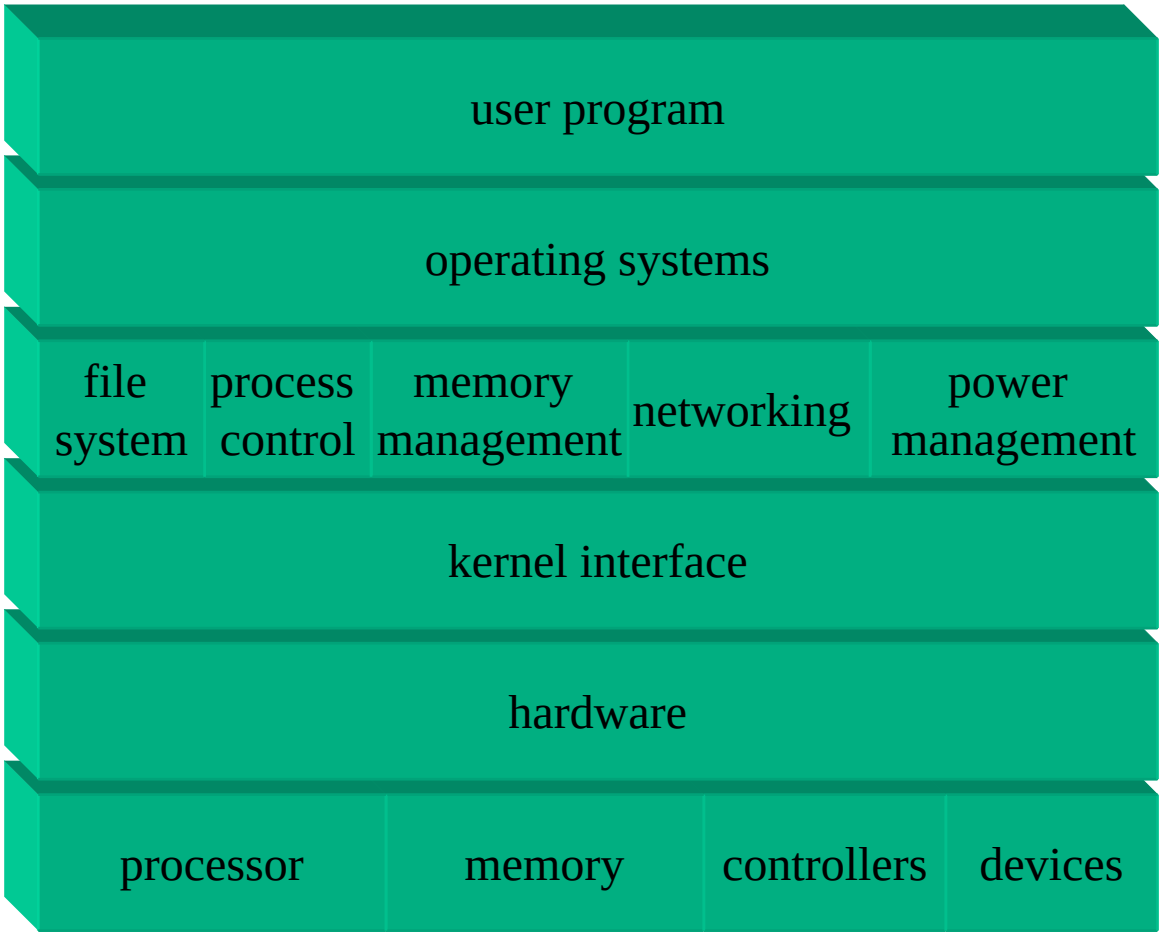
- Introduction
- Background
- Process-based power management
- Scheduling
- Implementation
- Experiment
- Conclusion



Introduction

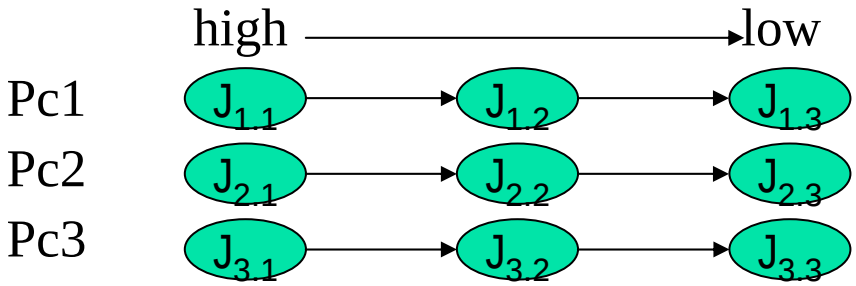
- OS kernel:
  - observe relationship between processes and devices to estimate the utilization of each device.
  - Provide new system call to allow application specify their future hardware requirement.

Introduction con’t



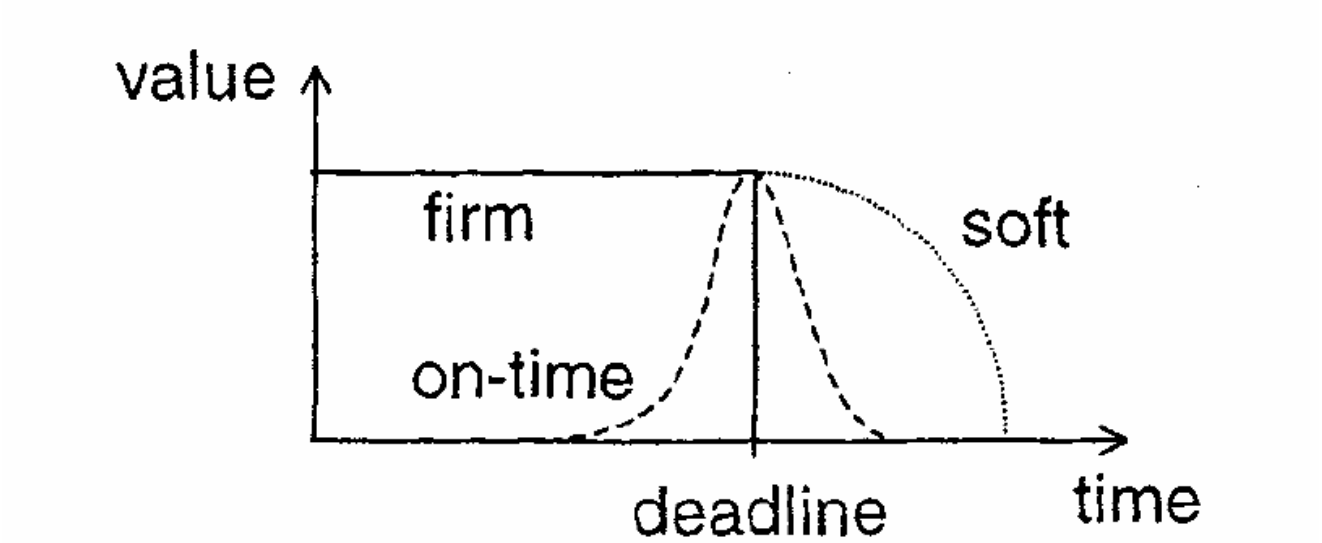
Background 1/3

- Job:
  - A unit to finish a specific task.
  - Can be scheduled to start at a specific time.
- Priority:



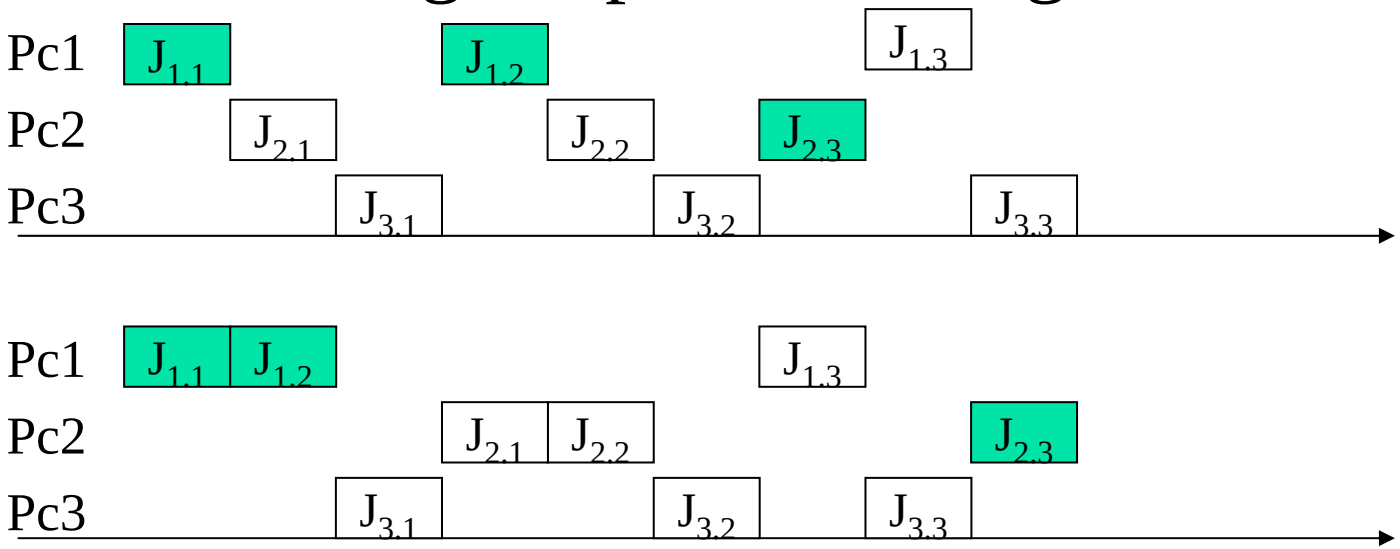
Background 2/3

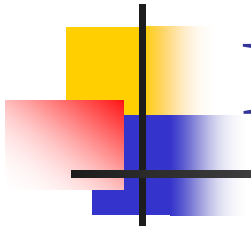
- Timing:
  - Firm.
  - Soft .
  - On-time.



Background 3/3

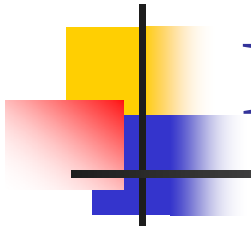
- Scheduling for power management:





Process-based power management

- When a process created, a new requester is born.
- Using a request generation model:
  - Separates request resource by processes.
  - Detects the termination of process.
  - Consider how often process execute.



Process-based power management

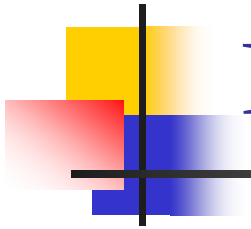
- Power manager consider both device and processor utilization.
- Device utilization :
  - $tbr_n = a*tbr+(1-a)*tbr_{n-1}$
  - $u_{i,j}= 1/tbr_n$

Process-based power management

- When process change from I/O burst to CPU burst, device utilization would overestimate.
- A function is used to adjustment the utilization:
  - $fi,j=e^{-(li,j/tbe.i)}$
  - Device utilization after adjustment is : $w_{i.j} = u_{i,j}*f_{i,j}$

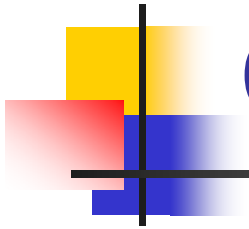
Process-based power management

- Processor utilization:
  - $c_j = \frac{\text{CPUTime}(pc_j)}{\sum_{\text{all process } pc_i} \text{CPUTime}(pc_i)}.$
- The aggregate utilization for device d_i is:
  - $u_i = \sum_{\text{all process } pc_j} w_{i,j} \times c_j.$



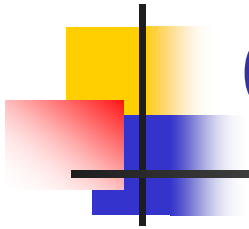
Process-based power management

- Power manager shutdown the device only when:
  - $u_i < k/t_{be.i.}$



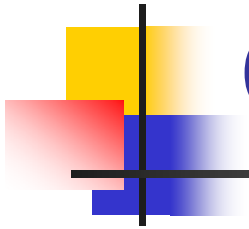
On-line Scheduling

- On-line scheduling can improve power management.
- Some are requests created by “timers” so that they can arrive at specific time in the future.



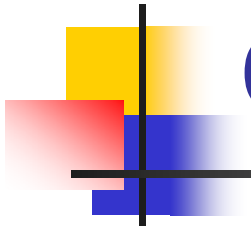
On-line Scheduling

- Scheduler improve power management in two step:
  - It wake up device before scheduling a job that requires this device.
  - It arranges execution orders to facilitate power management.



On-line Scheduling

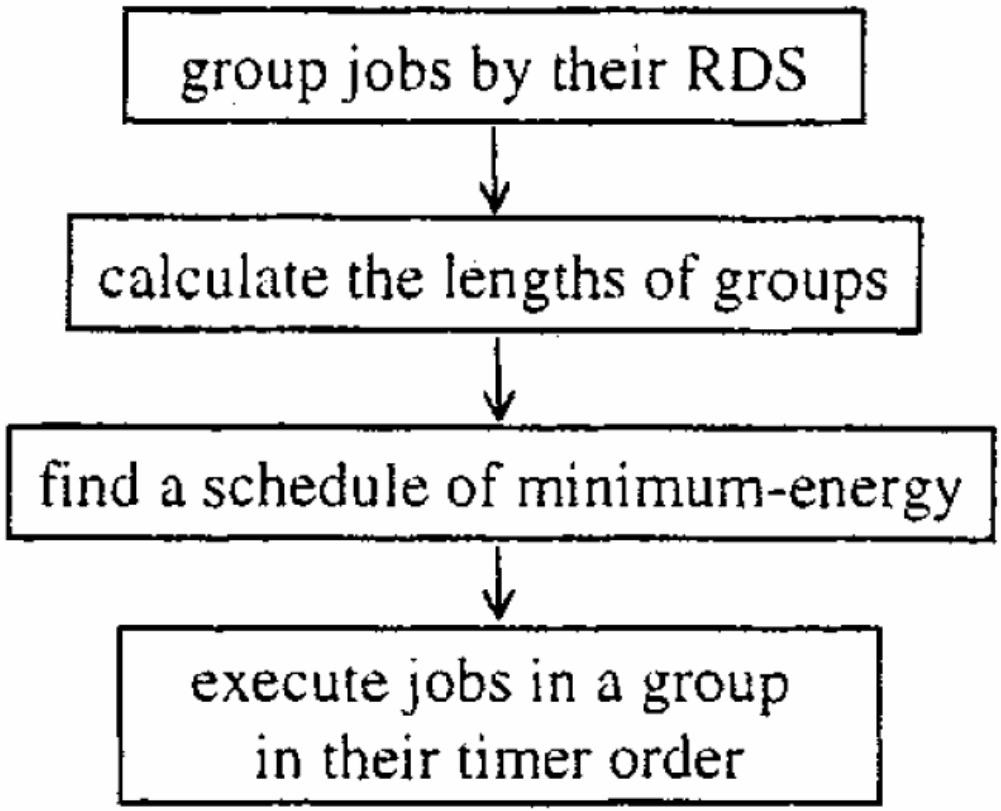
- Wake up device in advance can achieved by providing interface for processes to specify their device requirement.
  - RequireDevice(device, time, callback).
  - RequireDevice(device, tolerance, time, callback).



On-line Scheduling

- Scheduler can scheduling process for power management, if we know process's execution time.
  - RequireDevice(device, execution, tolerance, time, callback).

On-line Scheduling



On-line Scheduling

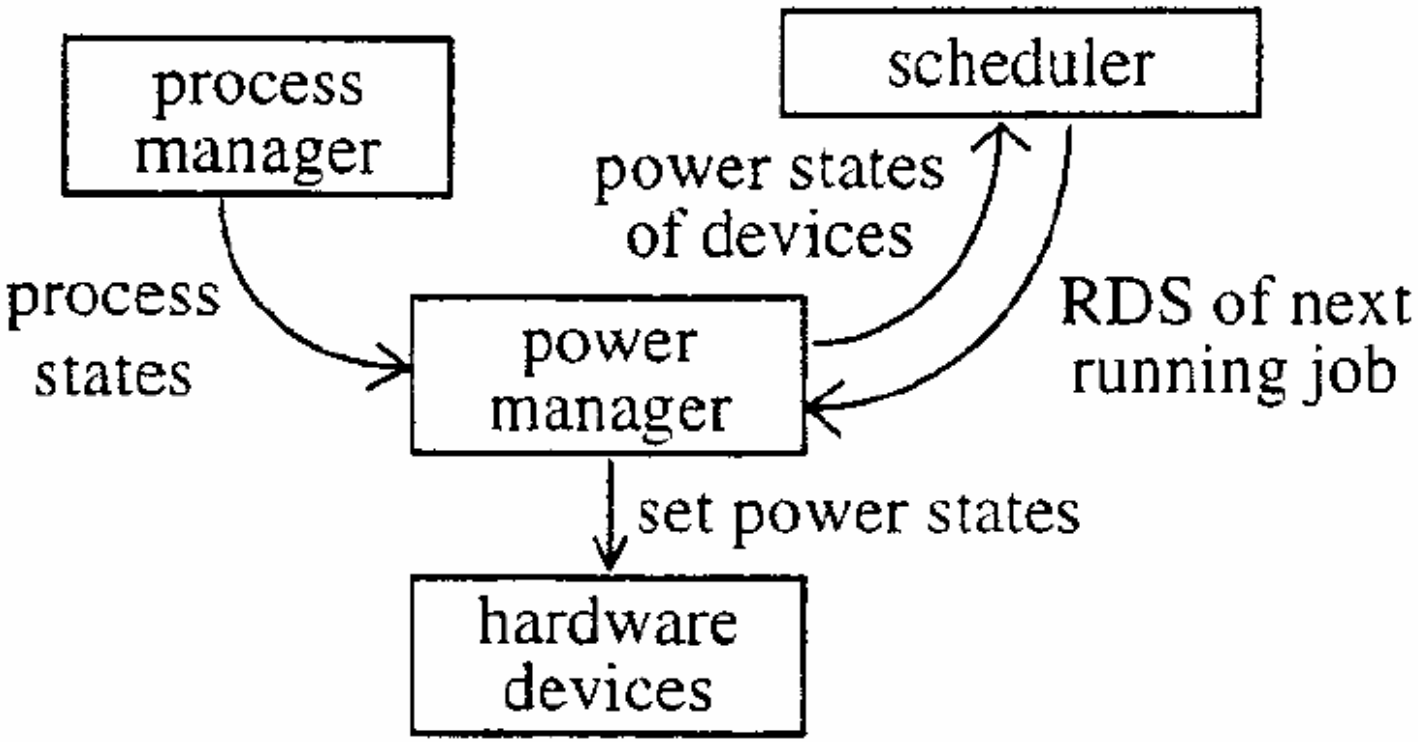
- Required device set: *RDS*
  - *RDS* = $|\mathcal{J}_i| = \sum_{\forall j_x \in \mathcal{J}_i} ex_x.$
- Low-power group schedule:*LPGS*
- *LPGS* = $(J_{s1}, J_{s2}, \ldots, J_{sq})$

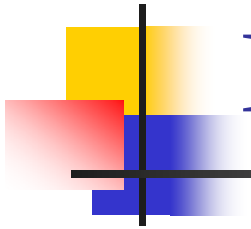
On-line Scheduling

```
/* initialization */
LPGS := empty; /* low-power group schedule */
/* end of initialization */

..... /* the steps in Figure 15; extension starts below */
if (new jobs are created)
      group jobs by their RDS's;
      calculate the length of each group;
      /* find one schedule first */
      S := one schedule of these groups;
      eng := energy of S;
      LPGS := S;
      /* check whether there are better schedule */
      for each schedule of these groups
            ethis := the energy of this schedule;
            if (ethis < eng) /* a better schedule */
                  eng := ethis;
                  LPGS := this schedule;
```

Implementation





Experiment

- Platform: Redhat 6.2 Linux on Sony VAIO notebook.
- Power management devices:
  - Linksys Ethernet card.
  - Hitachi 2.5” hard disk.

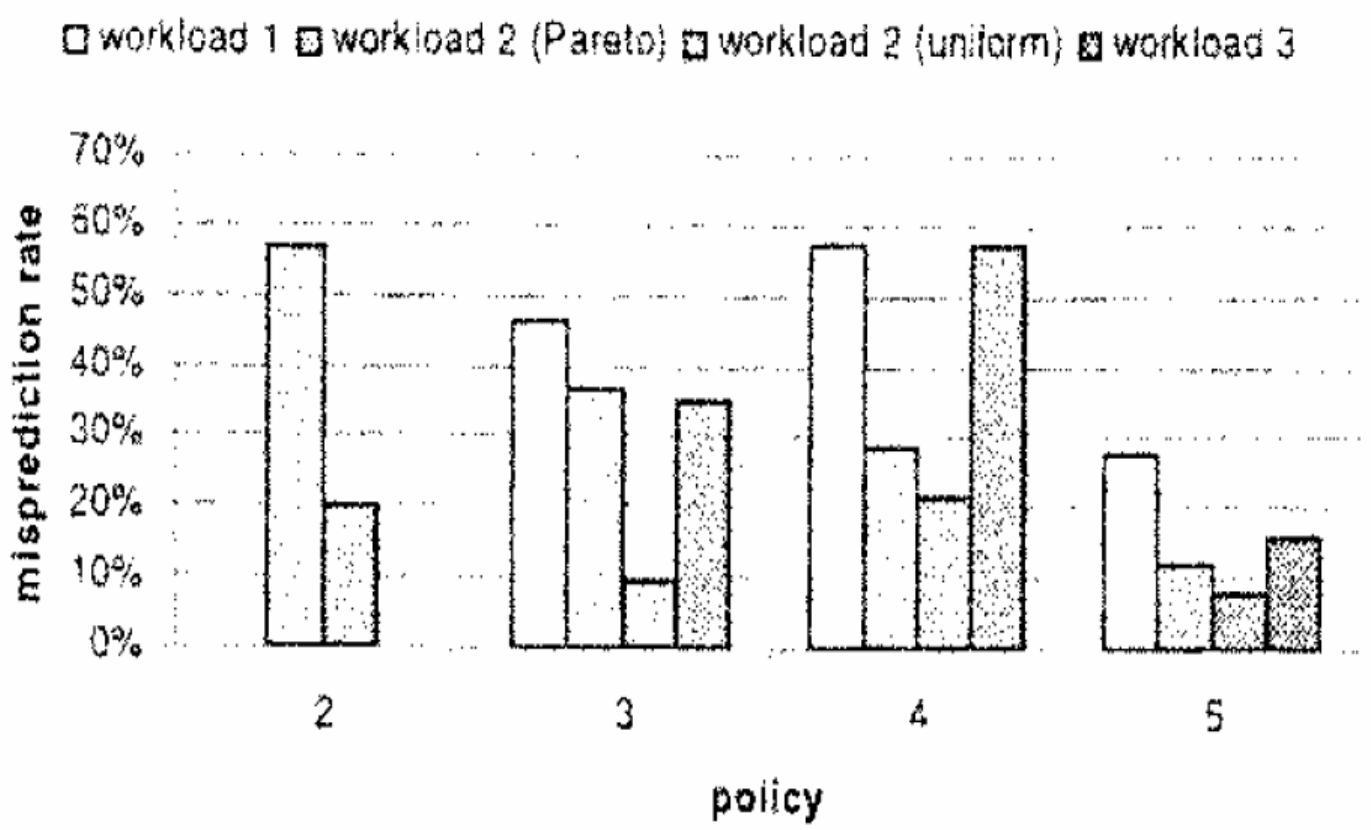
Experiment Result

workload	policy	hard disk					Ethernet card				
		p_a	t_s	t_t	sd	sd_w	p_a	t_s	t_t	sd	sd_w
1 trace	1	0.91	0	-	-	-	0.77	0	-	-	-
	2	0.89	44	53	5	0	0.58	352	14	5	0
	3	0.58	68	435	41	19	0.26	93	149	54	8
	4	0.74	19	774	73	42	0.57	17	262	95	21
	5	0.50	42	647	61	17	0.27	136	94	34	1
2 Pareto	1	0.90	0	-	-	-	0.77	0	-	-	-
	2	0.88	16	74	7	4	0.77	24	19	7	0
	3	0.57	43	721	68	25	0.43	40	283	103	12
	4	0.51	28	1113	105	30	0.42	14	605	220	46
	5	0.45	41	954	90	11	0.41	66	157	57	7
2 uniform	1	0.84	0	-	-	-	0.76	0	-	-	-
	2	0.81	46	106	10	2	0.73	89	11	4	0
	3	0.42	80	551	52	5	0.36	45	206	75	0
	4	0.44	45	837	79	17	0.43	16	497	181	29
	5	0.39	88	530	50	4	0.35	46	190	69	6

Experiment Result

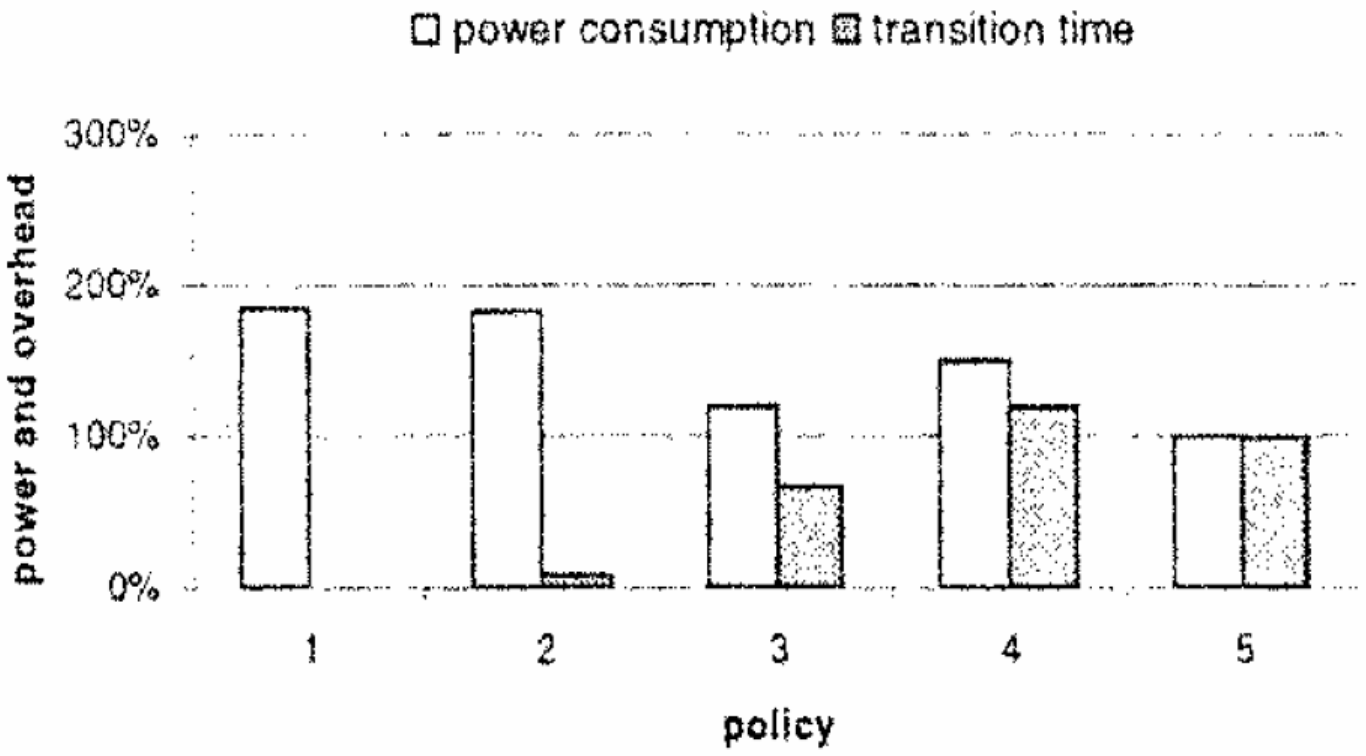
workload	policy	hard disk					Ethernet card				
		p_a	t_s	t_t	sd	sd_w	p_a	t_s	t_t	sd	sd_w
	1	0.88	0	-	-	-	0.76	0	-	-	-
	2	0.87	0	0	0	0	0.76	0	0	0	0
	3	0.45	105	392	37	13	0.31	33	302	110	9
3 timer	4	0.64	14	1198	113	65	0.39	41	225	82	22
	5	0.37	72	583	55	9	0.30	104	96	35	3
	6	0.35	85	530	50	0	0.29	80	140	51	1
	7	0.25	220	265	25	0	0.19	228	73	25	0

Experiment Result

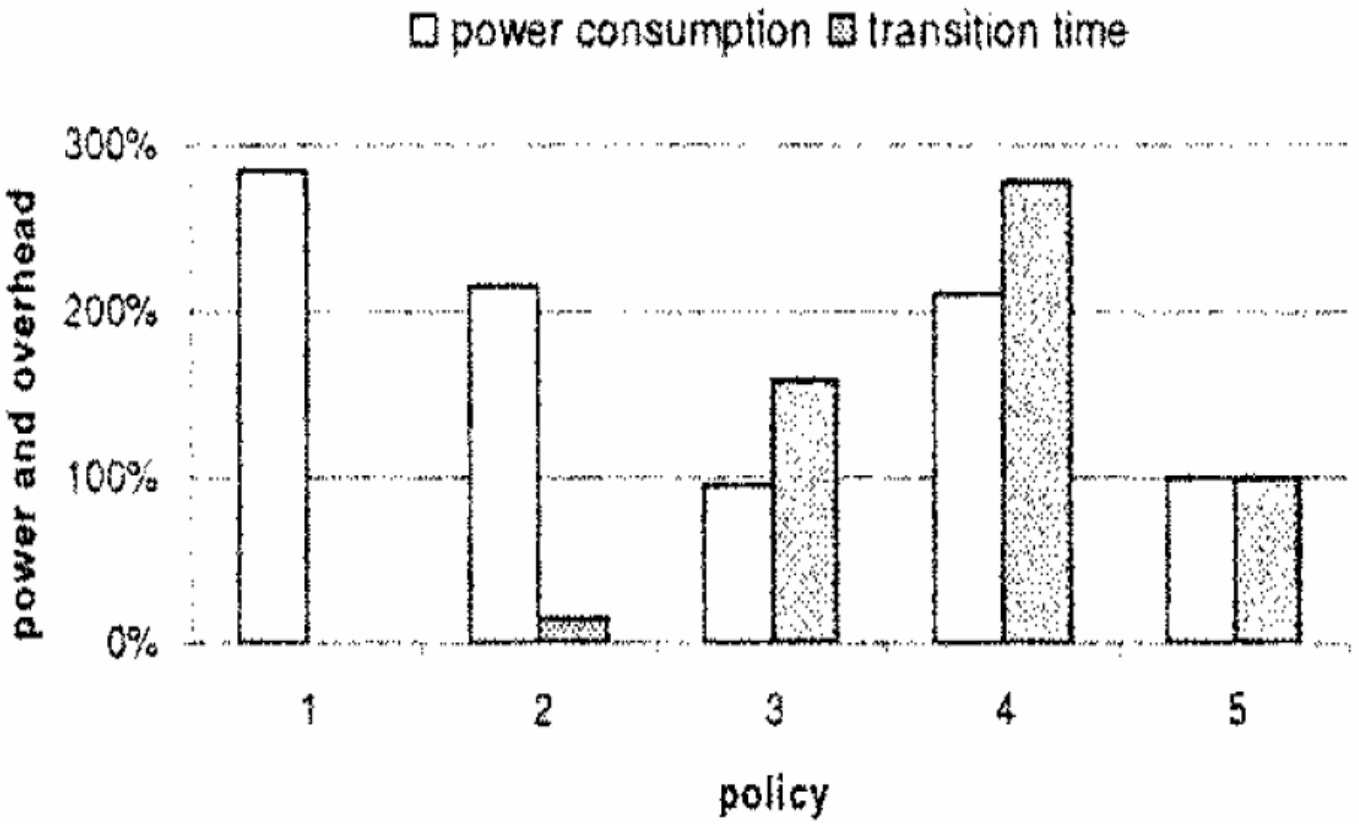


Experiment Result

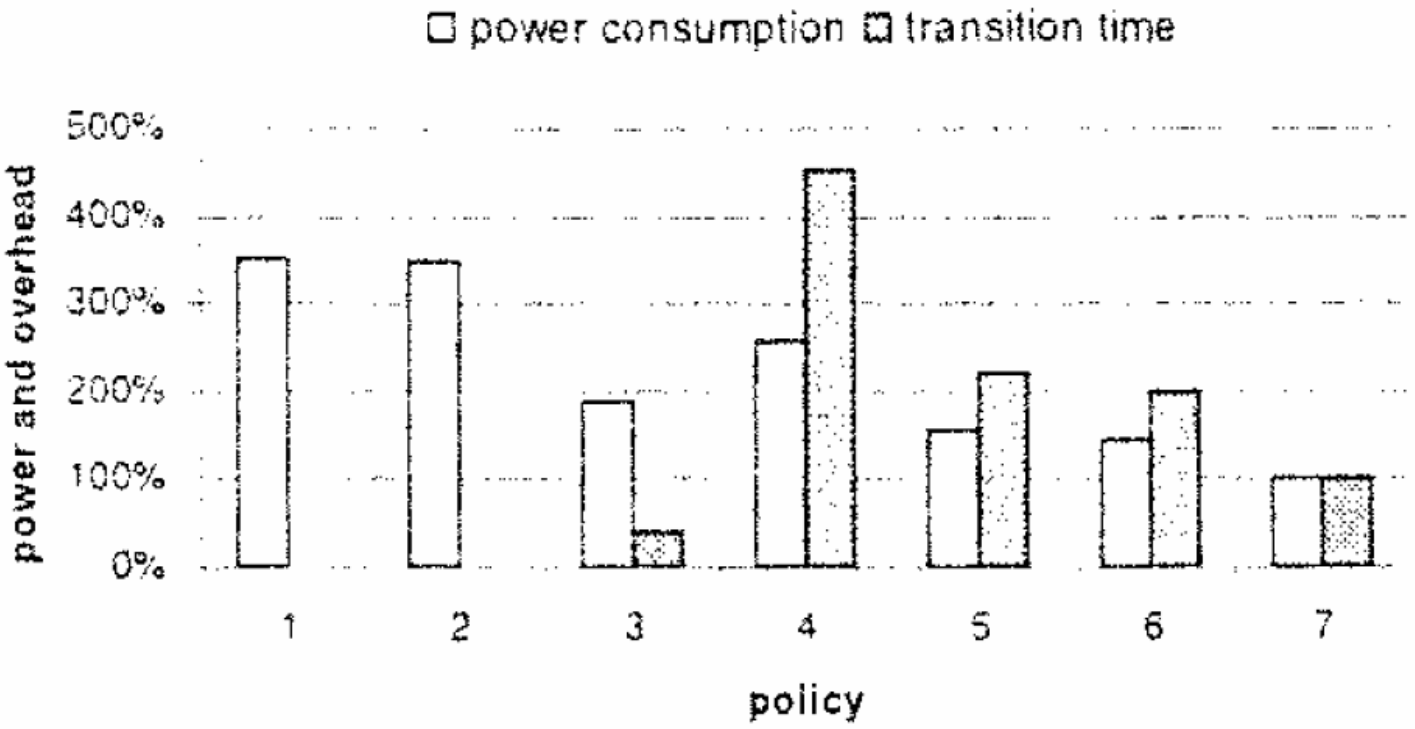
Experiment Result



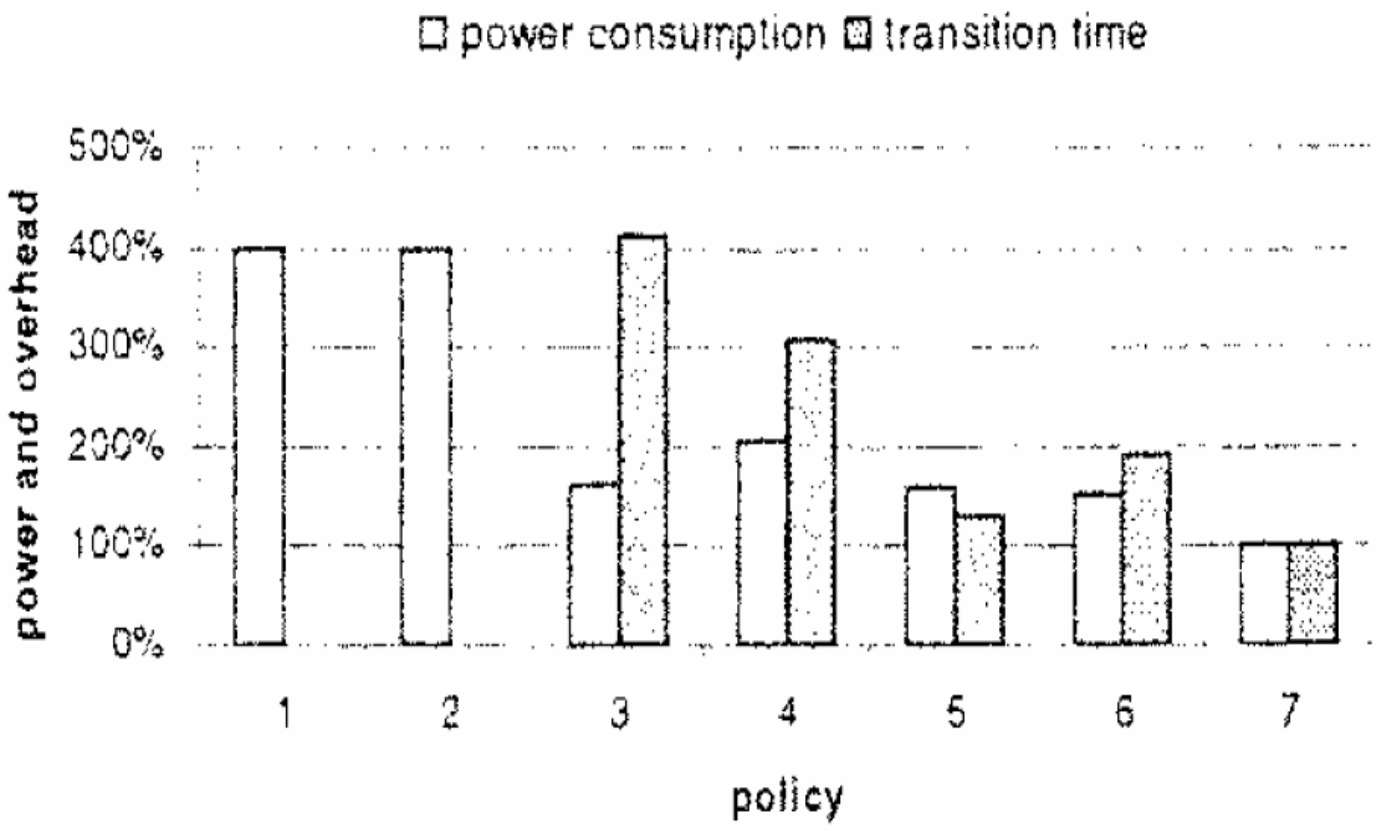
Experiment Result



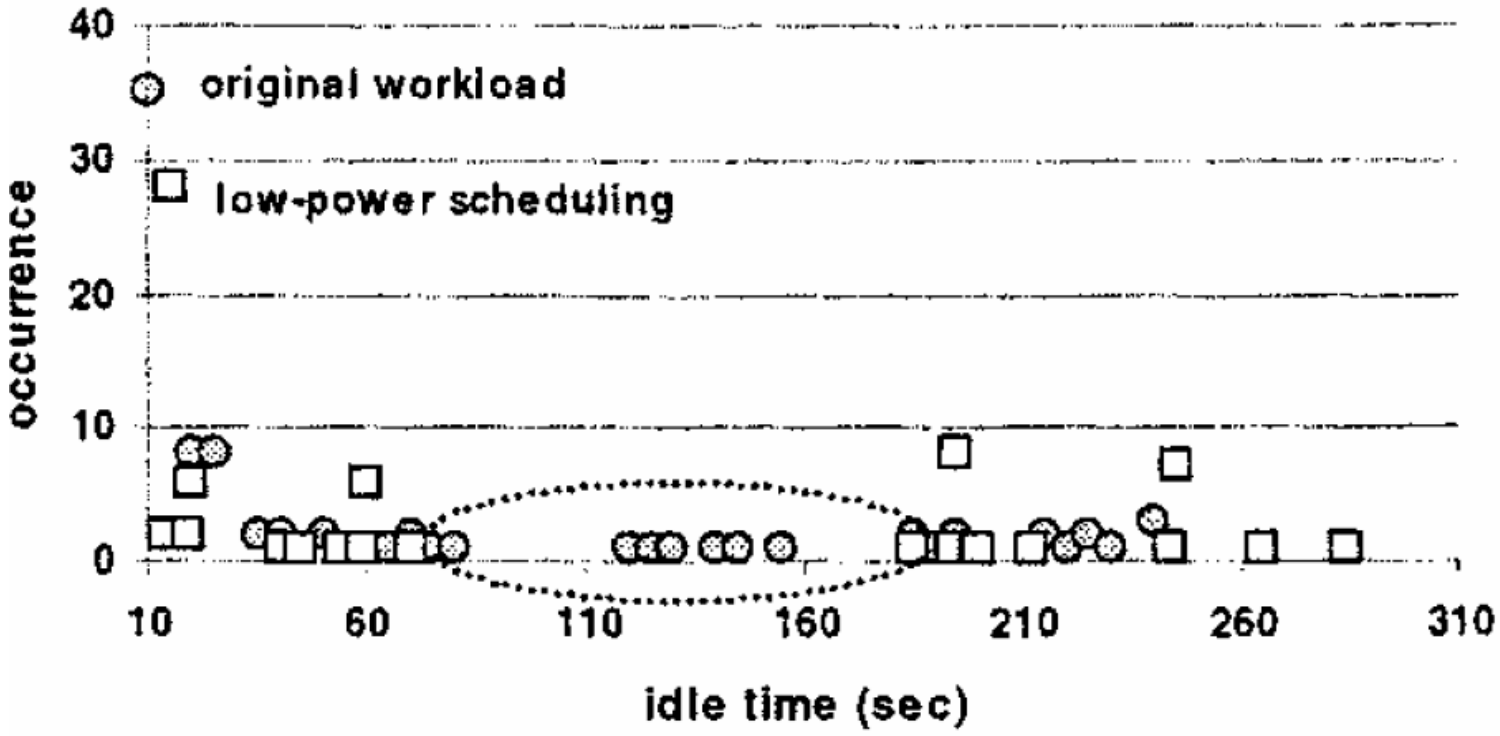
Experiment Result

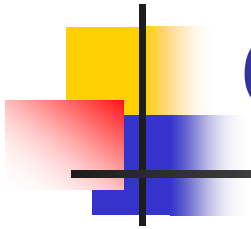


Experiment Result



Experiment Result





Conclusion

- Using OS kernel to distinguish individual processes for estimating the utilization of IO devices.
- Unused devices are shut down to save power.
- Experimental results achieved up to 72% power saving.
-