프로세스 관리

Process Management

양희재 교수 (hjyang@ks.ac.kr) / 경성대학교 컴퓨터공학과

프로세스

- 프로그램 vs 프로세스 (program vs process)
  - process, task, job …
  - program in execution: text + data + stack, pc, sp, registers, …
  - 무덤 속 *프로그램*, 살아 움직이는 *프로세스*
- 프로세스 상태
  - new, ready, running, waiting, terminated (그림)
  - 프로세스 상태 천이도 (process state transition diagram)
  - 상태 천이는 언제 발생?

PCB

- Process Control Block (PCB)
  - Task Control Block (TCB)
  - 프로세스에 대한 모든 정보
  - process state (running, ready, waiting, …), PC, registers, MMU info (base, limit), CPU time, process id, list of open files, …
  - 사람과 비유?

Queues

- Job Queue
  - Job scheduler
  - Long-term scheduler
- Ready Queue
  - CPU scheduler
  - Short-term scheduler
- Device Queue
  - Device scheduler

Multiprogramming

- Degree of multiprogramming
- i/o-bound vs CPU-bound process
- Medium-term scheduler
  - Swapping
- Context switching (문맥전환)
  - Scheduler
  - Dispatcher
  - Context switching overhead

CPU 스케줄링

CPU Scheduling

- Preemptive vs Non-preemptive
  - 선점 (先占) : 비선점(非先占)
- Scheduling criteria
  - CPU Utilization (CPU 이용률)
  - Throughput (처리율)
  - Turnaround time (반환시간)
  - Waiting time (대기시간)
  - Response time (응답시간)
  - …

CPU Scheduling Algorithms

- First-Come, First-Served (FCFS)
- Shortest-Job-First (SJF)
  - Shortest-Remaining-Time-First
- Priority
- Round-Robin (RR)
- Multilevel Queue
- Multilevel Feedback Queue

First-Come, First-Served

- Simple & Fair
- Example: Find *Average Waiting Time*
  - AWT = (0+24+27)/3 = 17 msec *cf. 3 msec!*
- Gantt Chart

Process	Burst Time (msec)
P_1	24
P_2	3
P_3	3

- Convoy Effect (호위효과)
- Nonpreemptive scheduling

Shortest-Job-First (1)

- Example: AWT = (3+16+9+0)/4 = 7 msec
  - *cf.* 10.25 msec (FCFS)
- Provably *optimal*
- *Not realistic*; prediction may be needed

Process	Burst Time (msec)
P_1	6
P_2	8
P_3	7
P_4	3

Shortest-Job-First (2)

- Preemptive or Nonpreemtive
  - *cf.* Shortest-Remaining-Time-First (최소잔여시간 우선)
- Example
  - Preemptive: AWT = (9+0+15+2)/4 = 26/4 = 6.5 msec
  - Nonpreemptive: 7.75 msec

Process	Arrival Time	Burst Time (msec)
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

Priority Scheduling (1)

- Priority (우선순위): typically an integer number
  - Low number represents high priority in general (Unix/Linux)
- Example
  - AWT = 8.2 msec

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

Priority Scheduling (2)

- Priority
  - Internal: time limit, memory requirement, i/o to CPU burst, …
  - External: amount of funds being paid, political factors, …
- Preemptive or Nonpreemptive
- Problem
  - Indefinite blocking: *starvation* (기아)
  - Solution: againg

Round-Robin (1)

- Time-sharing system (시분할/시공유 시스템)
- Time *quantum* 시간양자 = time *slice* (10 ~ 100msec)
- Preemptive scheduling
- Example
  - Time Quantum = 4msec
  - AWT = 17/3 = 5.66 msec

Process	Burst Time (msec)
P_1	24
P_2	3
P_3	3

Round-Robin (2)

- Performance depends on the size of the time quantum
  - $\Delta \rightarrow \infty$ FCFS
  - $\Delta \rightarrow 0$ Processor sharing (* context switching overhead)
- Example: *Average turnaround time (ATT)*
  - ATT = 11.0 msec ($\Delta = 1$), 12.25 msec ($\Delta = 5$)

Process	Burst Time (msec)
P_1	0
P_2	3
P_3	1
P_4	7

Multilevel Queue Scheduling

- Process groups
  - System processes
  - Interactive processes
  - Interactive editing processes
  - Batch processes
  - Student processes
- Single ready queue → Several separate queues
  - 각각의 Queue 에 절대적 우선순위 존재
  - 또는 CPU time 을 각 Queue 에 차등배분
  - 각 Queue 는 독립된 scheduling 정책

Multilevel *Feedback* Queue Scheduling

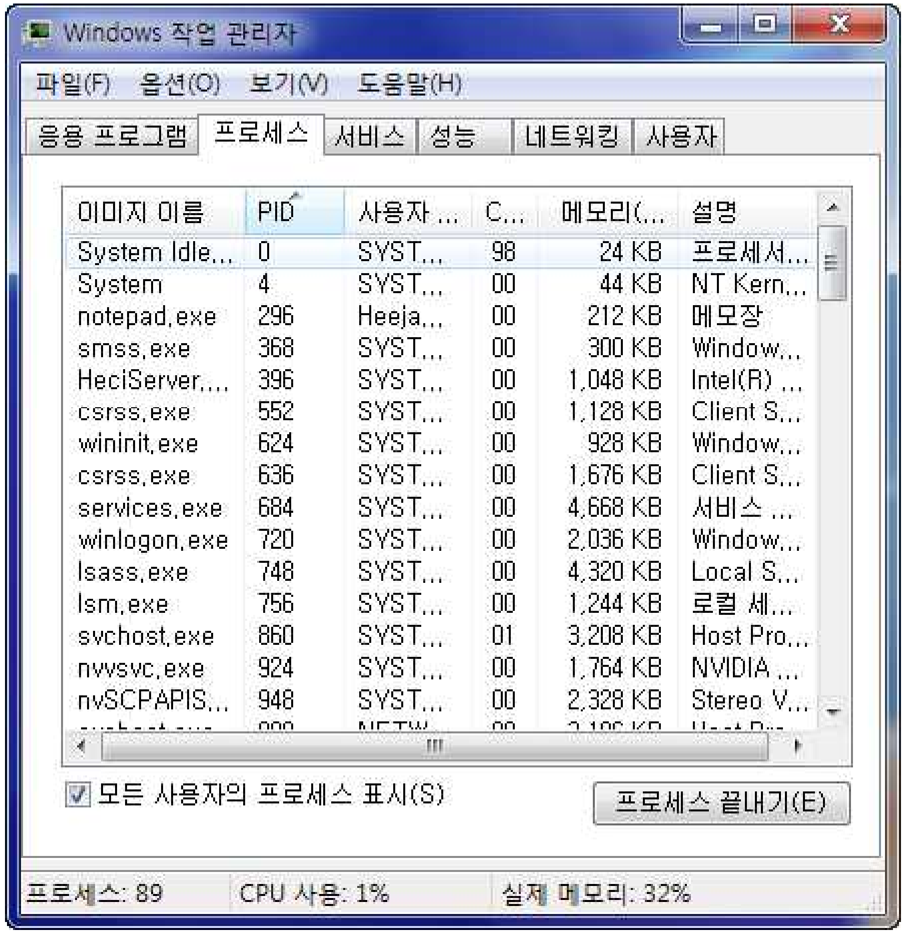
- 복수 개의 Queue
- 다른 Queue 로의 점진적 이동
  - 모든 프로세스는 하나의 입구로 진입
  - 너무 많은 CPU time 사용 시 다른 Queue 로
  - 기아 상태 우려 시 우선순위 높은 Queue 로

프로세스 생성과 종료

Process Creation

- 프로세스는 프로세스에 의해 만들어진다!
  - 부모 프로세스 (Parent process)
  - 자식 프로세스 (Child process)
    - *cf.* Sibling processes
  - 프로세스 트리 (process tree)
- Process Identifier (PID)
  - Typically an integer number
  - *cf.* PPID
- 프로세스 생성
  - *fork()* system call – 부모 프로세스 복사
  - exec() – 실행파일을 메모리로 가져오기

예제: Windows 7 프로세스 나열

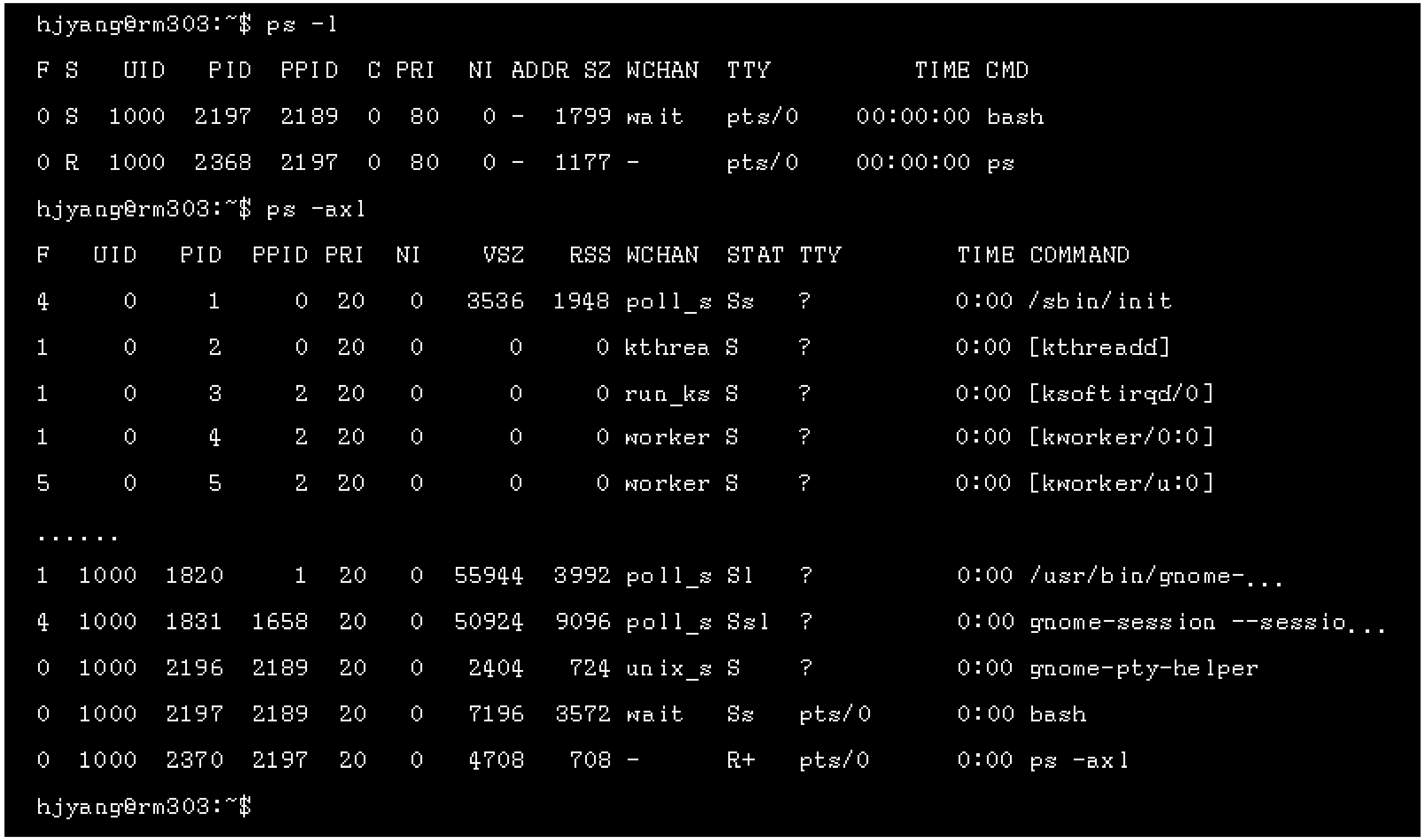


이미지 이름	PID	사용자 ...	C...	메모리(...	설명
System Idle...	0	SYST...	98	24 KB	프로세서...
System	4	SYST...	00	44 KB	NT Kern...
notepad.exe	296	Heeja...	00	212 KB	메모장
smss.exe	368	SYST...	00	300 KB	Window...
HeciServer....	396	SYST...	00	1,048 KB	Intel(R) ...
csrss.exe	552	SYST...	00	1,128 KB	Client S...
wininit.exe	624	SYST...	00	928 KB	Window...
csrss.exe	636	SYST...	00	1,676 KB	Client S...
services.exe	684	SYST...	00	4,668 KB	서비스 ...
winlogon.exe	720	SYST...	00	2,036 KB	Window...
lsass.exe	748	SYST...	00	4,320 KB	Local S...
lsm.exe	756	SYST...	00	1,244 KB	로컬 세...
svchost.exe	860	SYST...	01	3,208 KB	Host Pro...
nvvsvc.exe	924	SYST...	00	1,764 KB	NVIDIA ...
nvSCPAPIS...	948	SYST...	00	2,328 KB	Stereo V...

☑ 모든 사용자의 프로세스 표시(S)

프로세스: 89 CPU 사용: 1% 실제 메모리: 32%

예제: Ubuntu Linux 프로세스 나열



```
hjyang@rm303:~$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  2197  2189  0  80   0 -  1799 wait   pts/0    00:00:00 bash
0 R  1000  2368  2197  0  80   0 -  1177 -      pts/0    00:00:00 ps
hjyang@rm303:~$ ps -axl
F   UID   PID  PPID PRI  NI    VSZ   RSS WCHAN  STAT TTY        TIME COMMAND
4     0     1     0  20   0   3536  1948 poll_s Ss   ?          0:00 /sbin/init
1     0     2     0  20   0      0     0 kthrea S    ?          0:00 [kthreadd]
1     0     3     2  20   0      0     0 run_ks S    ?          0:00 [ksoftirqd/0]
1     0     4     2  20   0      0     0 worker S    ?          0:00 [kworker/0:0]
5     0     5     2  20   0      0     0 worker S    ?          0:00 [kworker/u:0]
......
1  1000  1820     1  20   0  55944  3992 poll_s Sl   ?          0:00 /usr/bin/gnome-...
4  1000  1831  1658  20   0  50924  9096 poll_s Ssl  ?          0:00 gnome-session --sessio...
0  1000  2196  2189  20   0   2404   724 unix_s S    ?          0:00 gnome-pty-helper
0  1000  2197  2189  20   0   7196  3572 wait   Ss   pts/0      0:00 bash
0  1000  2370  2197  20   0   4708   708 -      R+   pts/0      0:00 ps -axl
hjyang@rm303:~$
```

Process Termination

- 프로세스 종료
  - *exit()* system call
  - 해당 프로세스가 가졌던 모든 자원은 O/S 에게 반환

    (메모리, 파일, 입출력장치 등)