Operation System Project 1 result report

R08922164

Jeon Junyong

1. Coding Architecture:

- **CPU core :** parent process mapping to core2, child process mapping to core1 → scheduling just happened inside core1.
- **Method:**
  - **FIFO :** considering the order of size of ready time, reorder process priority (small → large), after fork( ), parent process make a child process, and call exec( ). "time.out" hold every part related with time, and this one finish one process.
  - **SJF :** first, get the order considering ready time, and it ready time is same, then compare execution time. With this two criteria, reorder whole process. After getting the whole order, invoke program, and "finish time is R[i] + T [i]". If there is a process which has smaller "finish time", then the next proce ss be invoked is that one. Depends on time order, after the first process co mpleted, the next process will be implemented(finish time = R[P[1]+finish_ti me]. Simply pseudo code is like below.

```
int finish_time = R[0];
for(int i = 0; i < p_num; i++){
    point = i;
    finish_time += T[P[i]];
    for(int j = i ; j < p_num; j++)
    {
        if(R[P[j]]>finish_time){
            break;
        }
        point++;
    }
    for(int j = point - 1; j >= i + 1; j--){
        for(int k = i + 1; k < j; k++){
            if(T[P[k]]>T[P[k+1]]) swap(&P[k], &P[k+1]);
        }
    }
}
```

  - **PSJF :** most of all, to set the suitable priority of something inside ready queue, I choose FIFO. Every one time unit for each process complete, choose which chil d process has the highest priority, then do fork( ) child process.
  - **RR :** Based on FIFO, adding time_count, and from the initial time of process sta

rts, circular rotates with clockwise. When time_count reaches 500, then the next process come up to be progressed.

2. System information :

OS : Windows 10
CPU : Intel® Core™ i7-4770 CPU @ 3.40GHz, 3401Mhz, 4core
Memory : 28GB
Viutual OS : Ubuntu 16.04 LT
Virtual OS Kernel version : 4.14.25
Virtual OS memory : 8GBVideo Link

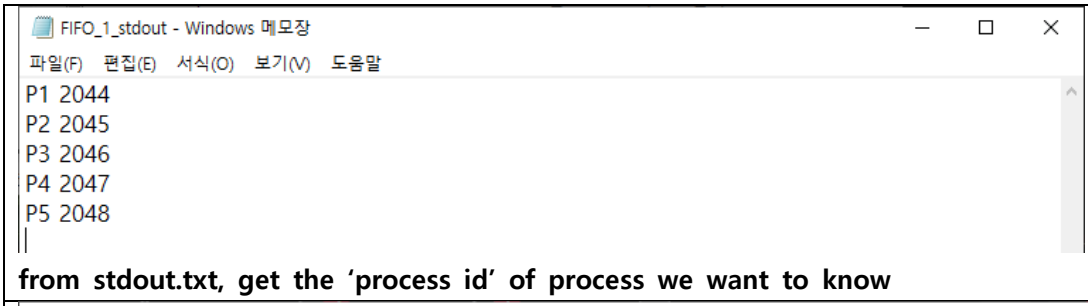
3. Result

3.1. summary

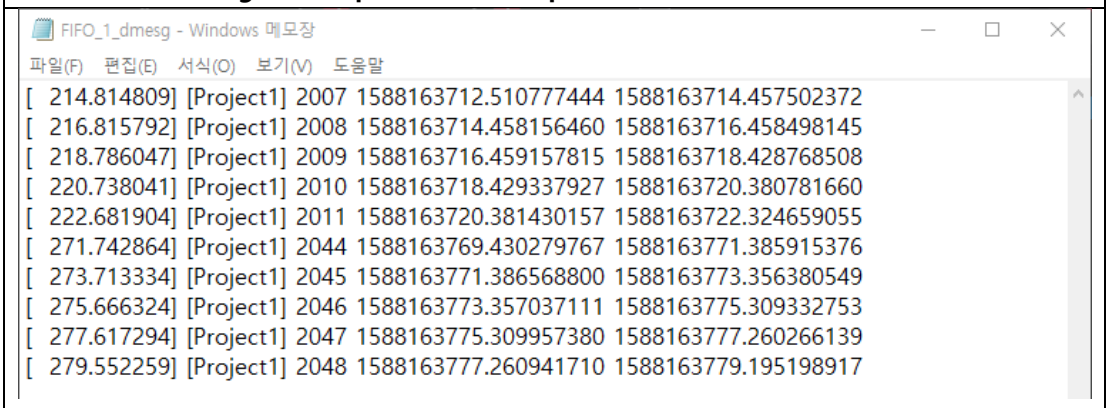
task		Theoretical value(s)			Real Value(s)		Time gap(s)	
		Average	Min	Max	(a)	(b)	(a)-理論	(b)-理論
FIFO	1	10.083	9.997	10.140	9.762	9.765	-0.321	-0.318
	2	350.894	347.897	352.876	347.763	347.765	-3.131	-3.129
	3	92.765	91.973	93.289	92.132	92.144	-0.632	-0.621
	4	12.906	12.796	12.979	12.770	12.772	-0.136	-0.134
	5	92.765	91.973	93.289	90.109	90.113	-2.656	-2.652
PSJF	1	100.832	99.971	101.401	202.890	86.386	102.058	-14.446
	2	44.366	43.987	44.616	59.428	39.201	15.062	-5.164
	3	14.116	13.996	14.196	19.905	11.771	5.788	-2.345
	4	56.466	55.983	56.785	59.476	55.039	3.010	-1.427
	5	61.709	61.182	62.057	61.477	60.687	-0.232	-1.022
RR	1	10.083	9.997	10.140	10.052	10.055	-0.031	-0.029
	2	36.299	35.989	36.504	63.625	35.664	27.325	-0.635
	3	120.998	119.965	121.681	406.973	118.908	285.975	-2.090
	4	92.765	91.973	93.289	330.710	84.654	237.945	-8.111
	5	92.765	91.973	93.289	332.099	85.131	239.334	-7.634
SJF	1	56.466	55.983	56.785	56.408	56.410	-0.058	-0.055
	2	61.709	61.182	62.057	62.083	61.258	0.374	-0.451
	3	129.145	128.042	129.874	128.637	128.642	-0.508	-0.503
	4	44.366	43.987	44.616	44.825	44.828	0.459	0.462
	5	14.116	13.996	14.196	13.999	14.001	-0.117	-0.115

Table 1. Result summary

3.2. measuring method (ex. FIFO_1)



from stdout.txt, get the 'process id' of process we want to know



find suitable time log from dmesg.txt

	A	B	C	D	E	F
1		pid	start time	finish time	time - gap	
2	FIFO_1	2044	1588163769.43027000	1588163771.38591000	1.95564008	
3		2045	1588163771.38656000	1588163773.35638000	1.96982002	
4		2046	1588163773.35703000	1588163775.30933000	1.95230007	
5		2047	1588163775.30995000	1588163777.26026000	1.95030999	
6		2048	1588163777.26094000	1588163779.19519000	1.93424988	=D6-C2
7				summation :	9.76232004	9.76492000

Table2. real value (a), (b) computation

In the table, real value (a) was computed in the green area.

real value (b) was computed in the yello area

3.3 Result analysis

- First of all, the time value simulation is faster than theoretical value, but it is obviously non sense. I guess it may because the execution time that is computed by averaging is not precis e, or it is not impossible to understand this situation.

- the reason why there is a time gap between theoretical value and real value is maybe com e from two reason; 1) Virtual OS, 2)busy waiting

1) Even we set intimate code where there is no time-wasting, but because Virtual OS cannot guarantee full occupation of CPU resources, so the lack of system resources may be the cause of time delay.

2) Scheduling method : In the process of scheduling, because parent process is hold on busy waiting situation to sound out time, and it may cause time delay(busy waiting). And the other suspect we can doubt is our scheduler is on 'user space', and because kernel still carry on 'round robin' methods to do scheduling, so it may cause time dumping.