Operating Systems

TEAM 6
Leonardo Nahra
Levindo Neto
Li-Wei Chen

Assignment 2

University of Stuttgart
20.11.2017

Assignment 2 - Answers

1.a)
The limitation of the Five State Process Model is that available resource usage relies on the numbers of deployed processes. For instance, myriad processes in memory would degrade the system's performance. Since there are processes which would be blocked for waiting for other processes to be done, probably there is not gonna be any process in the ready state. Hence, a low hit rate of accessing memory would cause slow performance.

1.b)
No, the "Ready/Suspend" state cannot directly go to the "running" state. A process in "ready/suspend" state is on the hard disk, and it has to be transferred to the main memory and be put in the "ready" state first. Then, it could be transferred to the "running" state.

1.c)
User-Level Thread

- Blocking problem: threads may be blocked if a system call is executed.
- Do not need to switch to kernel mode.

Kernel-Level Thread

- No blocking problem since kernel can schedule another thread.
- Need switch to the kernel mode.

1.d)
If we rank the execution time from short to long, we will get something as follow:
1 = 2 single-threaded = 2 multi-threaded > 3 single-threaded > 3 multi-threaded.
On the uni-processor with a single thread, a thread is only executed only once at a time. Since there is only one thread being executed at a time on the user-level thread, there is no benefit from multi-threaded. Also, the Task A is a single process, it does not benefit from multi-processor. Because more than one thread can be executed in the kernel-level thread, so it is faster than process on user-level thread. We assume multi-threaded process is faster, therefore multi-threaded process on multi-processor system with kernel-level thread is the fastest one.

2.

1 job:

1				1			

T 2T

TAT: NT
Throughput: 1/N
Processor utilization: 25%

5 job:

1	2	3	4	5	1	2	3

T 2T

TAT: T+5/4*T*(N-1) = T*(5*N/4-1/4)
Throughput: 5/(5*N/4+3/4)
Processor utilization: 100%

8 job:

1	2	3	4	5	6	7	8

T 2T

TAT: T*(2*N-1)
Throughput: 8/(2*N+3/4)
Processor utilization: 100%