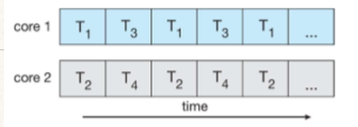
Chapter 4

multithread

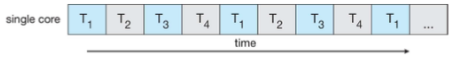
benefits: 1) responsiveness 2) resource sharing

3) economy 4) scalability

parallelism: system can perform more than one task simultaneously


core 1
T_1 T_3 T_1 T_3 T_1 ...
core 2
T_2 T_4 T_2 T_4 T_2 ...
time

concurrency: support more than one task making progress



speed up in latency $(L_2, L_1) = \frac{L_1}{L_2}$.

speedup $\leq \frac{1}{S + \frac{(1-S)}{N}}$ S: serial (concurrency) portion N: core number

multithreading modes:

1) many-to-one: kernel is limiting parallelism

2) one-to-one: kernal may limit or run out of resource.

3) many-to-many: kernel needs to manage threads

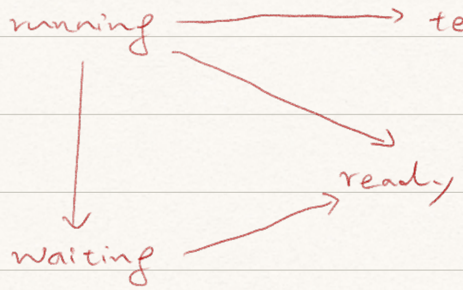
asynchronous cancellation: terminates the target thread immediately

deferred cancellation: check periodically target thread should be terminate

Chapter 5

CPU-I/O Burst Cycle: process execution consist of a cycle of CPU execution and I/O wait

decision take place in:



preemptive vs nonpreemptive: a preemptive scheduling scheme could terminate the process while running, and it might result in race condition

dispatch latency: time taken for stopping one process and start running another

Algorithm	scheme	drawback
FCFS	non	convey effect: short processes stuck at the end of FCFS queue
LCFS	non	risk of starvation
SJF	non	unknown CPU burst time ; risk of starvation

$$T_{n+1} = \underset{\substack{\uparrow \\ \text{usualy} \\ 1/2}}{\alpha}\ \underset{\substack{\uparrow \\ \text{actual} \\ \text{time}}}{\underline{t_n}} + (1-\alpha)\ \underset{\substack{\uparrow \\ \text{estimate} \\ \text{time.}}}{\underline{T_n}}$$

Round-Robin	preemptive	if time quantum q is large, it is almost same as FCFS.
Priority	non	risk of starvation