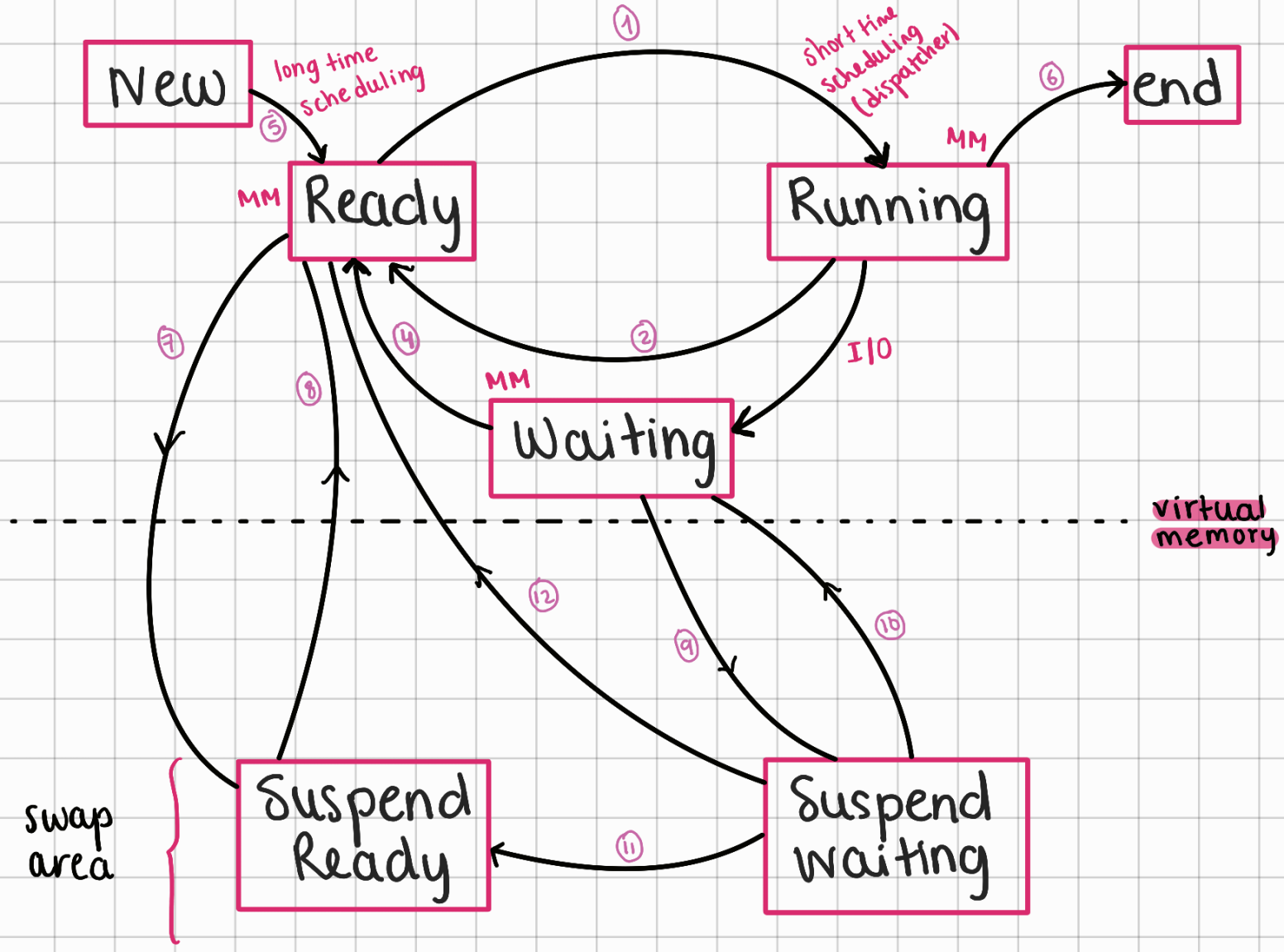
lecture 6

Scheduling

Prog1.c → SM
compile → Prog1.o
./Prog1 ↓
new SM.



① the schedule chooses the process to execute.
② quantum time expires / a process with higher priority takes the place of the current process.
③ the process requests I/O.
④ the I/O request is done.
⑤ new process is created.

⑥ the process ends.
⑦ the ready queue is full so it sends processes to the suspend ready
⑧ there is space in the ready queue for the processes in the suspend ready to return.
⑨ the block queue is full so it sends processes to the suspend wait.
⑩ there is space in the block queue for the processes in the suspend wait to return.
⑪ a process in the suspend wait was served but there is no space in the ready queue so it goes to the suspend ready.
⑫ a process in the suspend wait was served and returns to the ready queue.

Context Switching:

→ save the state of P_1
→ load the state of P_2

Multiprogramming

With preemption

the process needs more time to finish its execution but the dispatcher gives the CPU to another process

Without preemption

the process executes, on CPU with termination on making I/O request.

Multiprogramming with preemption:

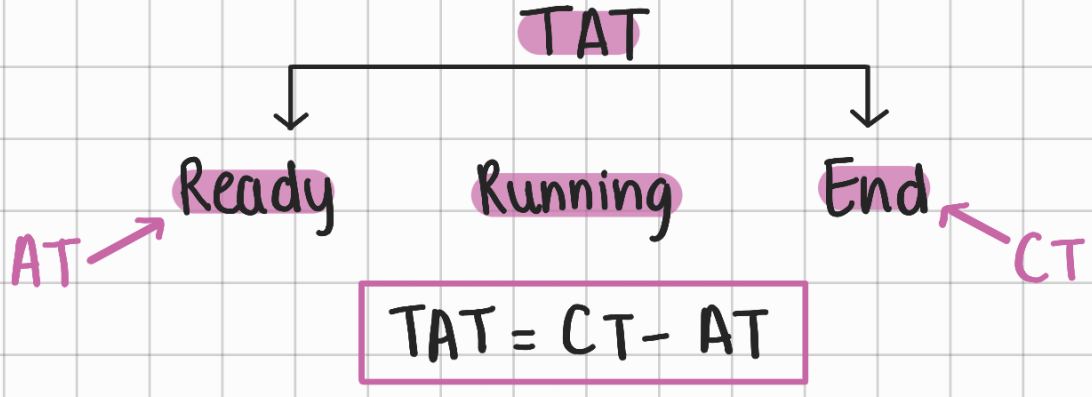
→ multi-tasking
→ time-sharing system

Important parameters for process:

① **Arrival Time (AT)**:
the time the process is inserted in the ready queue.

② **Burst Time (BT)**:
amount of CPU time required by a process to finish executing

③ **Completion Time (CT)**
the time at which the process ends its execution.



$$TAT = CT - AT$$

④ **Turn around Time (TAT)**
the time of the trip from starting to end of the execution.

⑤ **Waiting Time (WT)**
the time spent in the ready queue.

⑥ **I/O Time**:
the time spent into waiting queue

$$TAT = CT - AT$$
$$= BT + WT + IOT$$

⑦ **Response Time:**
the first instant it starts execution in the CPU.

AT → Ready ⟶ Running ← RT

RT = WT (no preemption)

Scheduling Criteria(s)

① **CPU Utilization**
the CPU would be busy
theory: 100%.
reality: 40%.-90%.

② **Throughput**
Number of completed processes during unit of time

T: interval of time
n: processes completed.
efficiency: $\frac{n}{T} \times 100$ (percentage of time where the CPU was busy).

③ **TAT ↘ minimize**

④ **WT ↘ minimize**

⑤ **RT ↘ minimize.**

Scheduling:

who ? → short term scheduler / dispatcher.

where? → ready state → running state.

when? → when a process moves from:

- Run → Termination
- Run → Ready
- Run → Wait
- New → Ready
- Wait → Ready