Scheduling

Chapter 7

When to act?

- Previously, we covered mechanisms for **how** to:
  - Request privileged operations from user process from the OS
  - Perform a context switch to swap out a running user process different process to share the CPU

- We now need to address the policies to address **which** process will be selected when it is time for a context switch

- Our **scheduling policies** (disciplines) are the rules we use to make decisions about which **jobs** (processes) run

Considerations for Scheduler Policies

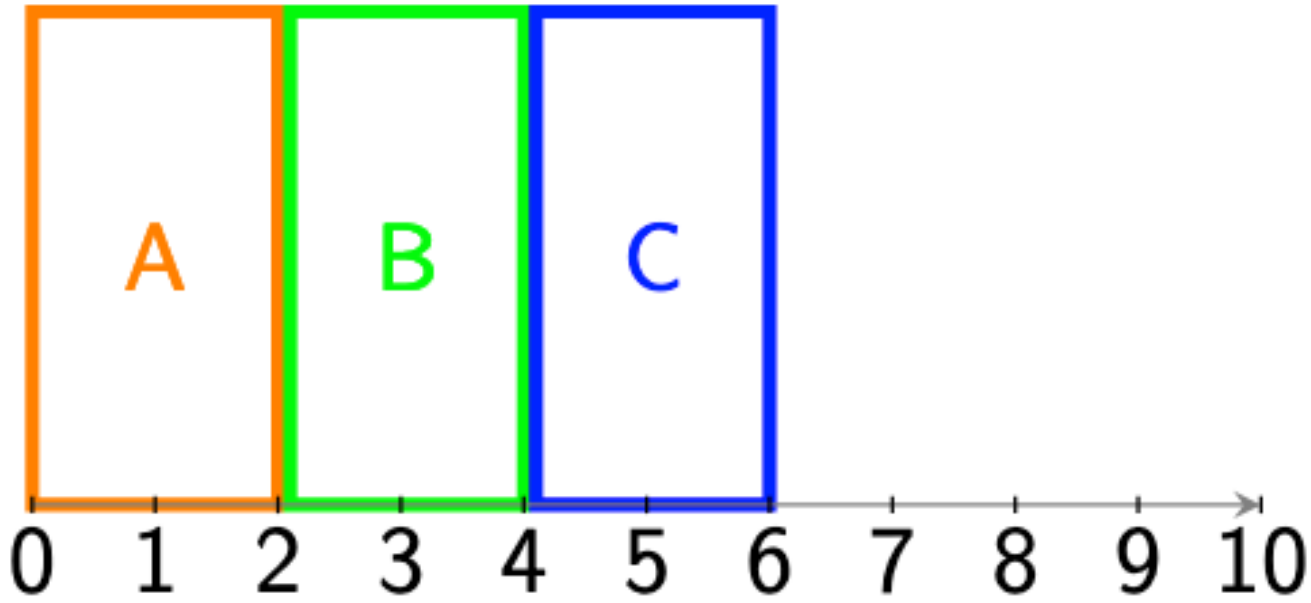
- **Utilization** is the fraction of time the CPU is used (maximize)
- **Turnaround time** is the difference between the time a job completes, and the time it arrived in the system (minimize)
  - $Time_{turnaround} = Time_{completion} – Time_{arrival}$
- **Response time** is the difference between the time a job is first run, and the time it arrived in the system (minimize)
  - $Time_{response} = Time_{first\ run} - Time_{arrival}$
- **Fairness** is an emphasis on processes being treated equally (no starvation)
- The **overhead** costs of context switching (minimize kernel interrupts)

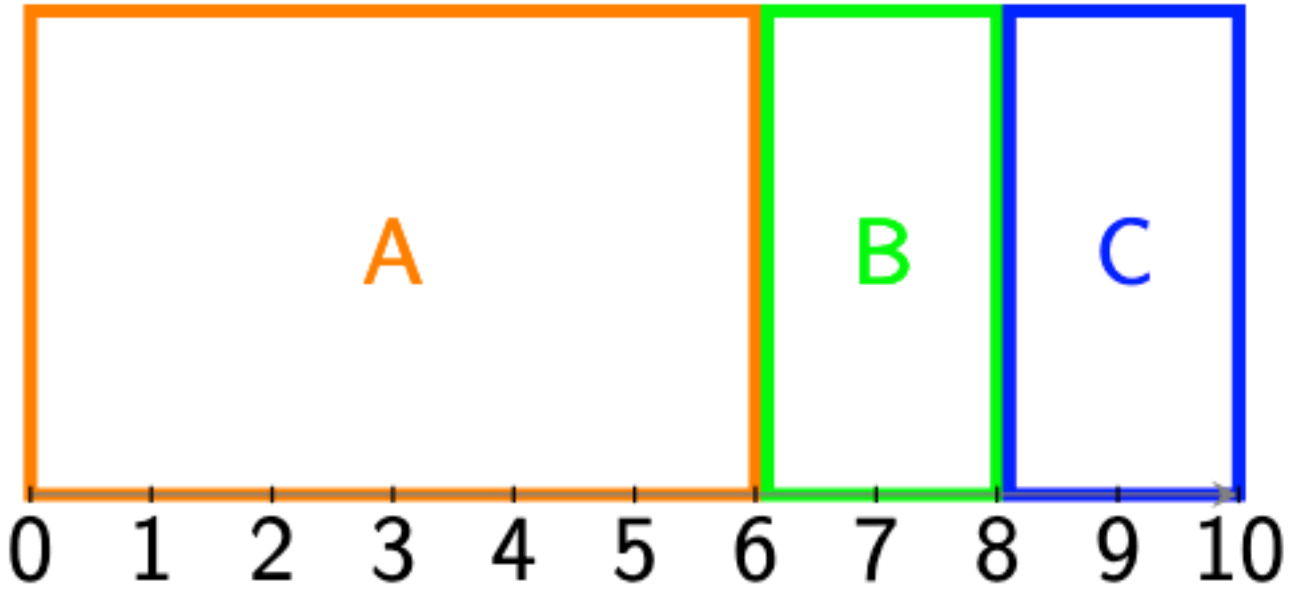
Scheduling Policies

- First In, First Out (FIFO) / First Come, First Served (FCFS)
  - Jobs run to completion in the order they are received
- Shortest Job First (SJF)
  - Jobs are chosen with the least amount of work needed first and run to completion (when the jobs arrive is important)
- Shortest Time-to-Completion First (STCF) / Shortest Remaining Time First (SRTF)
  - The job with the least amount of work needed is always selected and run first even if that means switching jobs
- Round Robin (RR)
  - Every job is given a fix time on the CPU and swapped out repeatedly until complete

FIFO Example

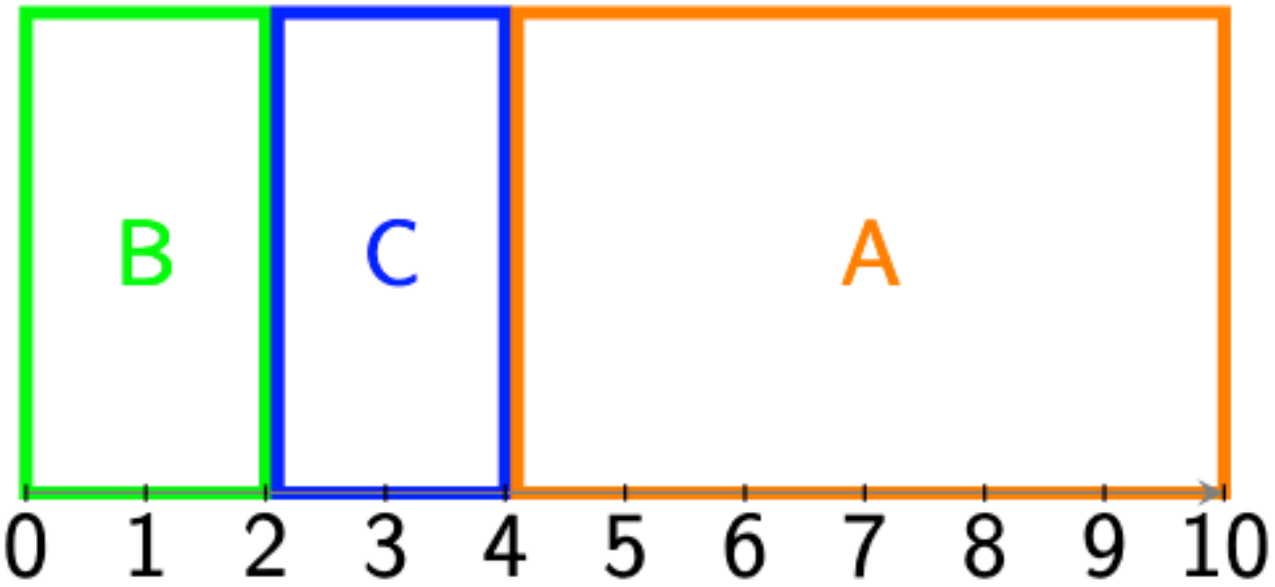
- Jobs A, B, C
  - Arrive at T=0
- Average Turnaround Time
  - (2+4+6) / 3 = 4
- Average Response Time
  - (0 + 2 + 4) / 3 = 2
- What could go wrong?
  - Convoy Effect
  - Turnaround: (6+8+10)/3 = 8
  - Response: (0+6+8)/3 = 4.7

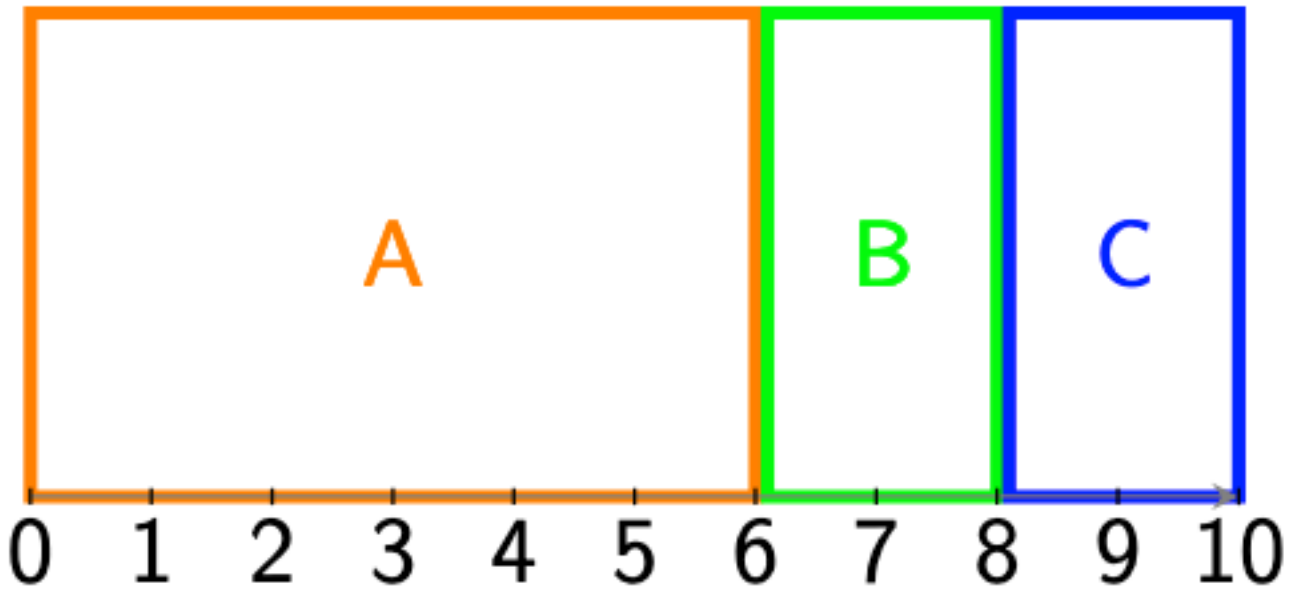




SJF Example

- Jobs A, B, C
  - Arrive at T=0
  - Length B,C = 2
  - Length A = 6
- Average Turnaround Time
  - (2+4+10) / 3 = 5.3
- Average Response Time
  - (0 + 2 + 4) / 3 = 2
- What could go wrong?
  - Job B and C Arrive at T = 1
  - Turnaround: (6+7+9)/3 = 7.3
  - Response: (0+5+7)/3 = 4



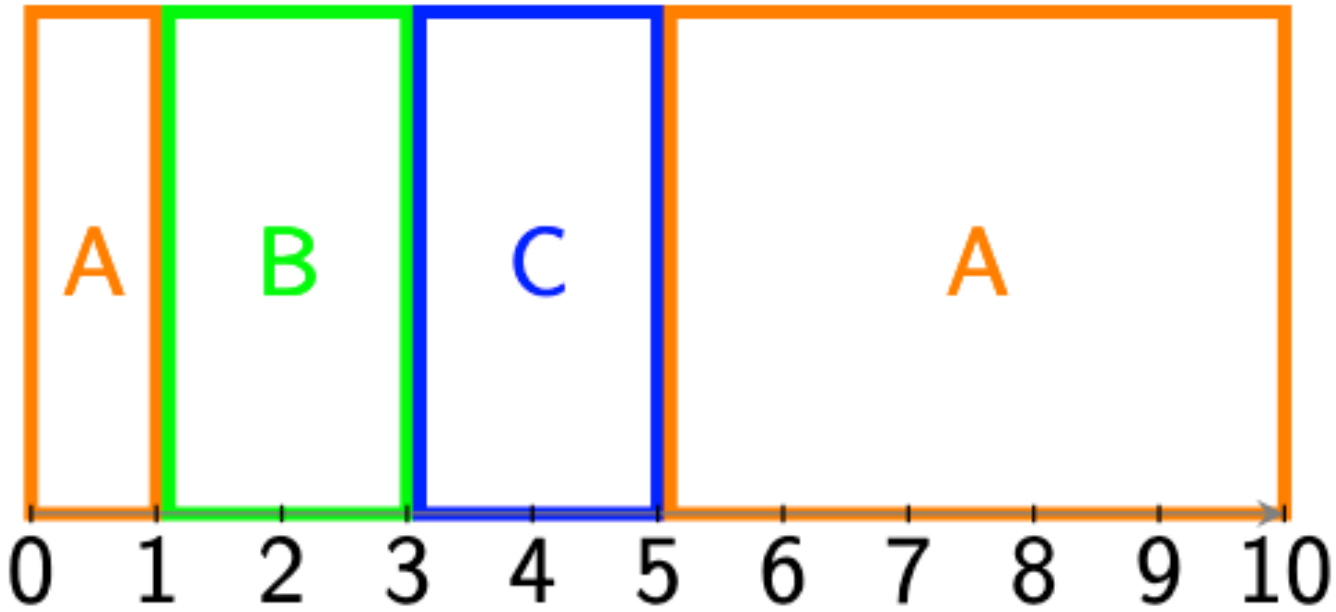


Preemptive Scheduling

- FIFO and SJF are non-preemptive
  - They only switch to another process if the current process gives up the CPU voluntarily (Yield, Done, Error, etc.)
- Preemptive scheduling can take control of the CPU at any time, and switch to other processes according to the scheduling policy
- STCF is preemptive and always runs the job that will complete the fastest

STCF Example

- Jobs A, B, C
  - Job A Arrive at T=0
  - Job B, C Arrive at = 1
- Average Turnaround Time
  - (2+4+10) / 3 = 5.3
- “First” Response
  - (0 + 0 + 2) / 3 = 0.7
  - Task A Preempted
- Reschedule when new jobs arrive



Currently we have optimized for turnaround time…

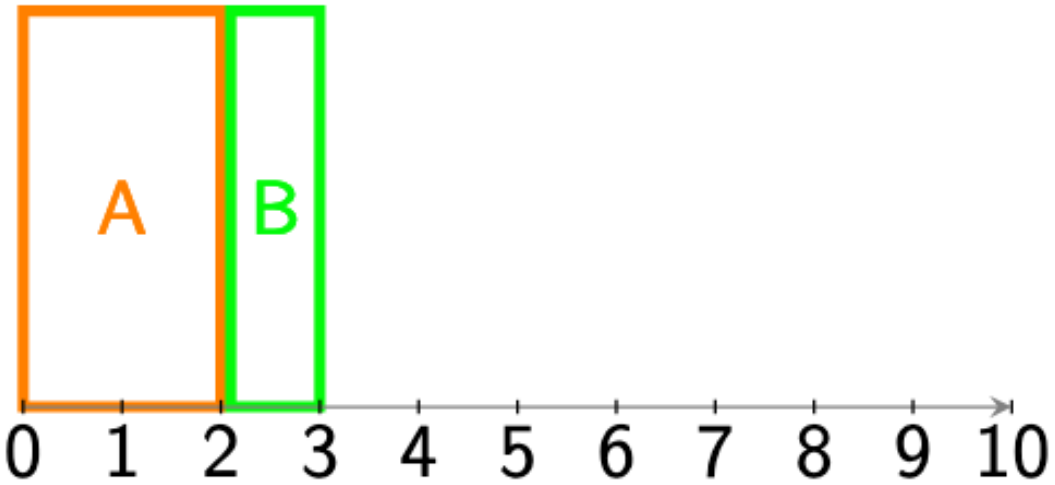
What if we have an interactive system?

Optimizations

- **Turnaround time** optimization wants to get the jobs done as quickly as possible

- **Response time** optimization wants to get the jobs scheduled to run on the CPU as quickly as possible

- Task A arrives at 0 with length 2
- Task B arrives at 1 with length 1
  - B Turnaround: 2
  - B Response: 1

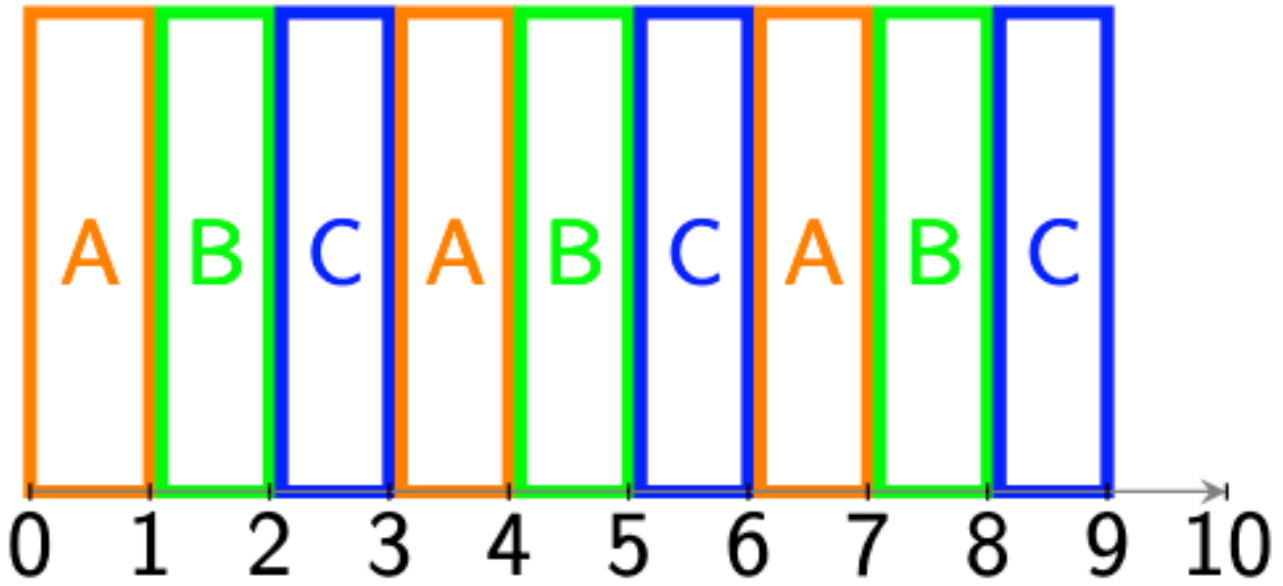


Round Robin

- Every process gets a fixed quantum slice of CPU time
  - Slice must be large enough that it offsets the overhead of context switching (amortizes the cost)
- Preemptive
- Good for response time and fairness
- Bad for turnaround time

RR Example

- Jobs A, B, C
  - Arrive at T=0
  - Length of 3
- Average Response Time
  - (0+1+2)/3 = 1
- How does it compare with FIFO's response time?
  - (0 + 3 + 6) / 3 = 3



I/O Considerations

- We want to maximize the utilization of our resources
- Use blocked time for a job to overlap other jobs that are ready
- Keep the disk and CPU busy (doing valuable work)

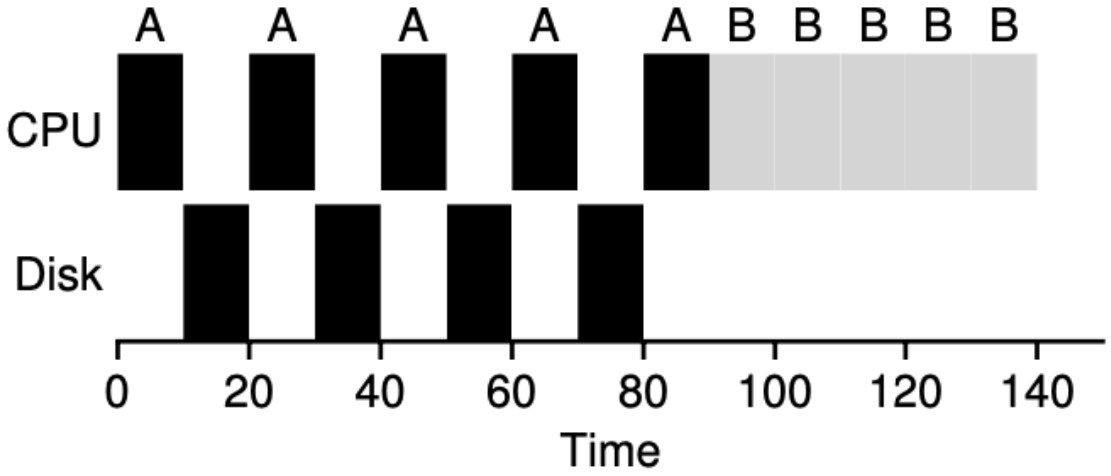


Figure 7.8: **Poor Use Of Resources**

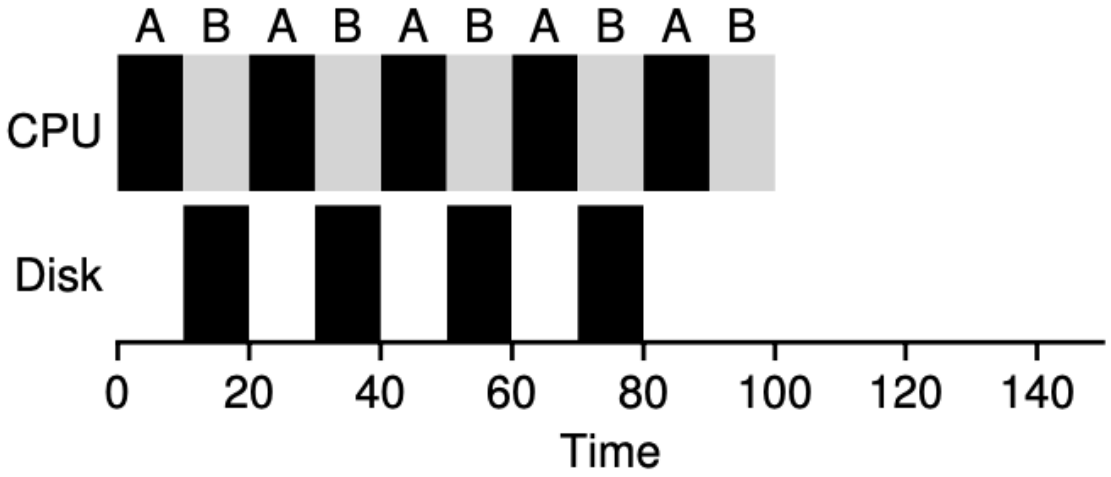


Figure 7.9: **Overlap Allows Better Use Of Resources**