CS3.301: Operating Systems and Networks

P. Krishna Reddy and Karthik Vaidhyanathan

E-mail: pkreddy@iiit.ac.in, karthik.vaidhyanathan@iiit.ac.in

http://www.iiit.ac.in/~pkreddy

https://karthikv1392.github.io/

Offices: Data Sciences and Analytics Center (DSAC)/Software Engineering Research Cengter

Course topics

- Introduction (3 hours)
- Process and thread management (6 hours)
- CPU Scheduling (3 hours)
- Process Synchronization (4.5 hours)
- Deadlocks (1.5 hours)
- Memory management (4.5 hours)
- Virtual Memory (4.5 hours)
- File Systems (1.5 hours)
- Protection and Security (1.5 hours)
- Networking (9 hours)

Outline

- History, development and concepts of Operating Systems
- Different kinds of Computer Systems
- Concept of virtual computer

Stored Program Computer

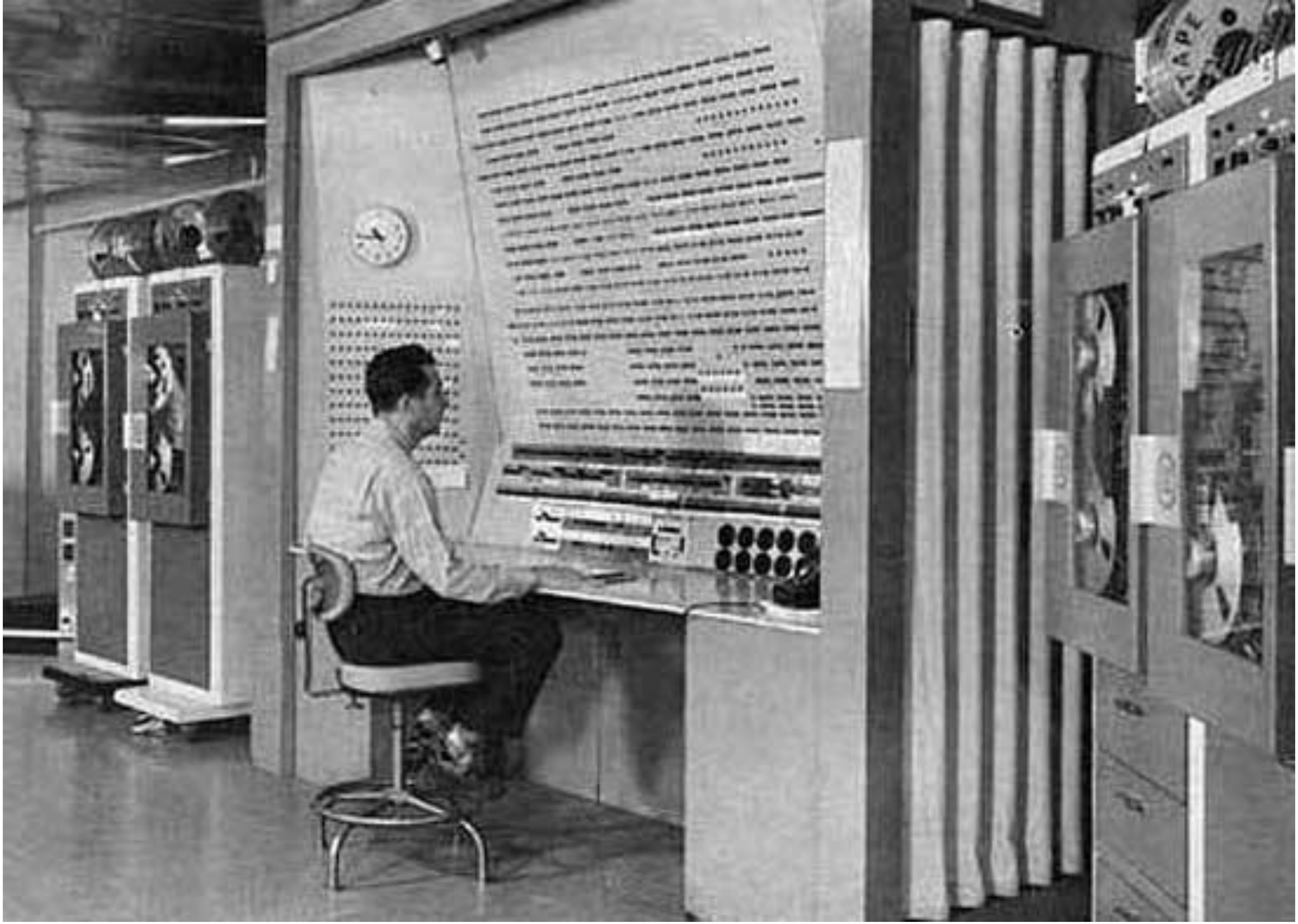
- 1940
  - Computers were designed to perform specific tasks.
  - Modification of the tasks required a great deal of effort and manual labour
- Alan Turing and John von Neumann
  - Proposed the concept of stored program computer.
  - **Machine has both program store and data store and program store provides instructions about what to do on data.**
  - This concept has generated the concept of general purpose computer.
- *Watch the movie*
  - *The Imitation Game*
- *1951*
  - *First general purpose computer*
    - *Machester Mark 1 (ran duriing 1940 to 1949)*
  - *First general purpose commercial computer was available in the market*
    - *Ferranti Mark1*

Early systems (Serial processing)

- **1940-50:**
  - The programmer interacted directly with the computer hardware.
  - Display light, switches, printer, card reader.
  - No OS.
  - Error is displayed through lights.
- **Problems:**
  - Scheduling → Users spend lots of time at the computer.
    - Signup sheet was used.
  - Job Setup time
    - Loading and compiling
      - Mounting and Un-mounting of tapes
      - Setting up of card desks
  - Libraries of functions, linkers, loaders, debuggers, and I/O driver routines were available for all the users.

Early Systems…

- Early computers were (physically) large machines run from a console.
- The programmer would operate the program directly from the console.
  - The program is loaded to the memory from panel of switches, paper tape, and from punched cards.
- As time went on, additional software and hardware were developed.
  - Card readers, line printers, and magnetic tape became common place.
  - Libraries, loaders, and common functions were created.
    - Software reusability.



TAPE

Early Systems…

- The routines that performed I/O were especially became important.
- Device driver: A special subroutine was written for each I/O device.
    - A device driver knows how the buffers, flags, registers, control bits, and status bits for a particular device should be used.
    - Device driver is written once and called from the library.
- Later, compilers for FORTRAN, COBOL and other languages have appeared.

Early Systems…

- Significant amount of setup time.
- Each job consisted of many separate steps:
  - Loading the FORTRAN compiler tape
  - Running the compiler
  - Unloading the compiler tape
  - Loading of assembler tape
  - Running assembler
  - Unloading the assemble tape
  - Loading the object program
  - Running the object program
- If error occurred during any step, you have to start over at the beginning.

Early Systems..

- The setup time was a real problem
- CPU is idle while tapes are being mounted or the programmer was operating the console.
- In the early days, few computers were available and they were expensive (millions of dollars).
  - +operational costs: power, cooling, programmers.
- Main question:

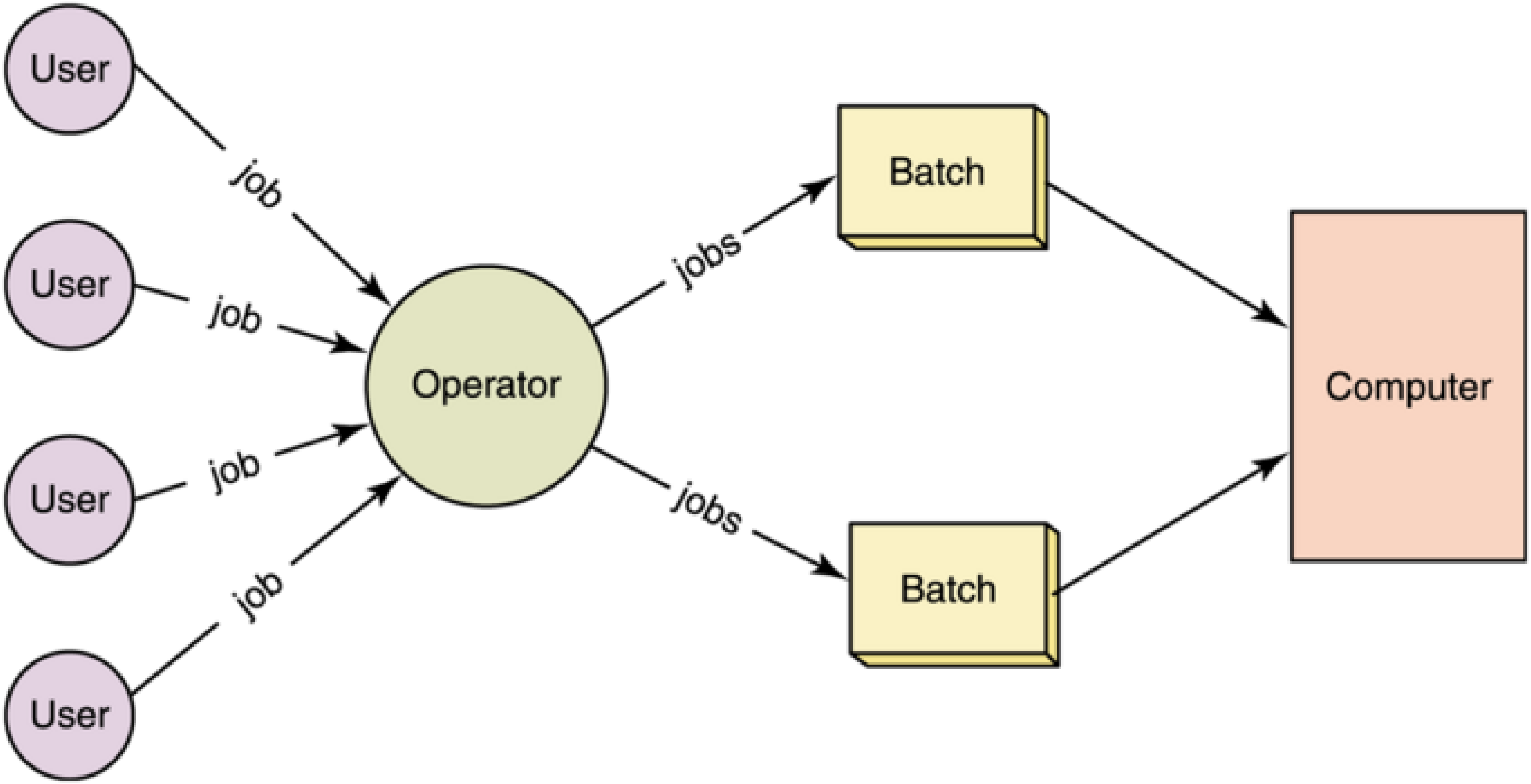
How to increase the utilization of CPU ?

Early Systems…

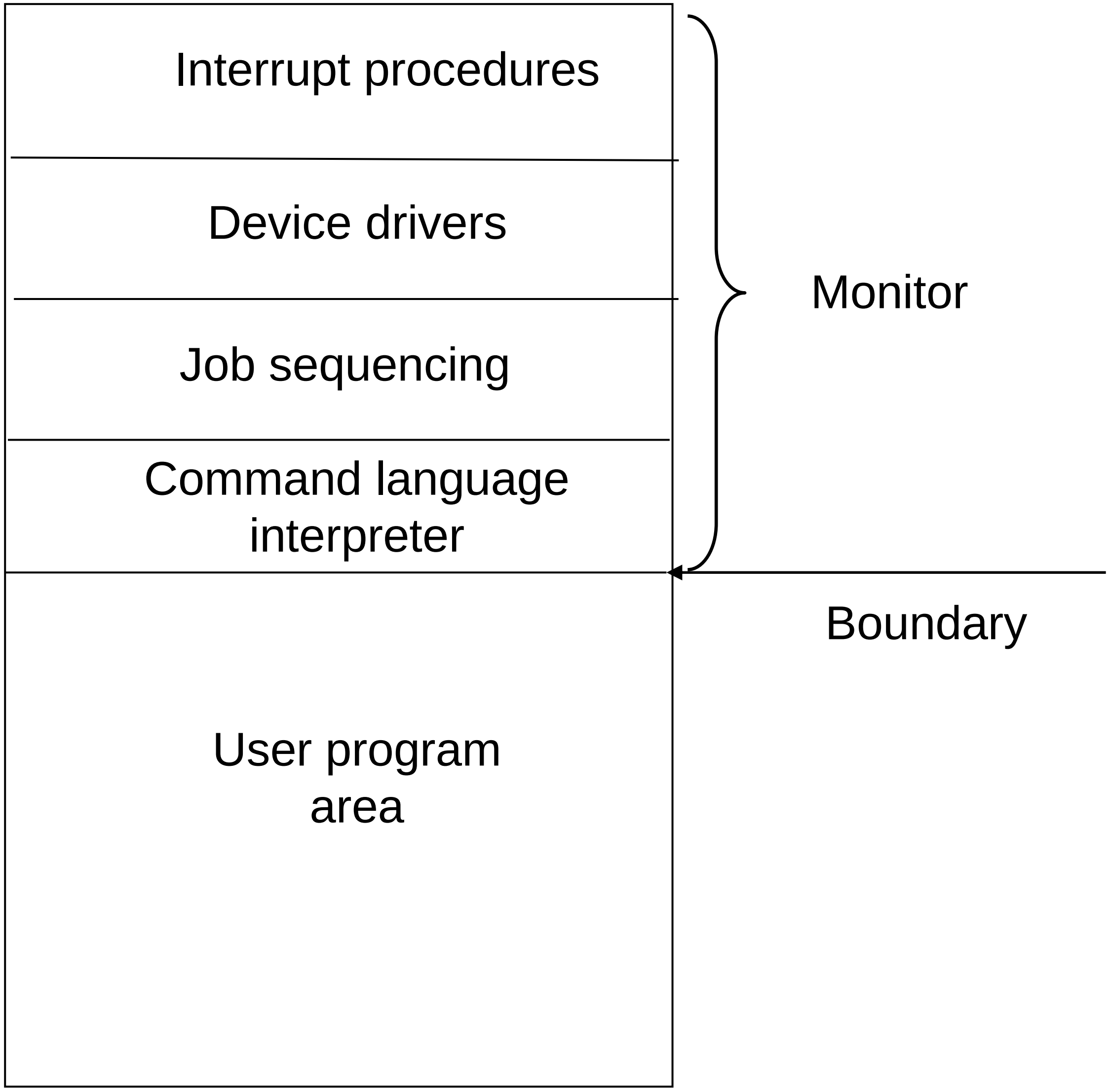
- The solution was two fold.
- First, a professional computer operator was hired.
  - Once the program was finished, he operator could start next job.
  - The operator sets up the job, produces the dump, and starts the next job.
  - The set up time was reduced due to operator’s experience.
- Second, jobs with similar needs were batched together and run through the computer as a group.
  - For example, if there is a FORTRAN job, COBOL job, and FORTRAN job, two FORTRAN jobs were batched together.
- However, during transition time CPU sat idle.
- To overcome this idle time, people developed **automatic job sequencing**.
  - A first rudimentary OS was created
  - A small program called a **resident monitor** was developed.
  - The resident monitor always resided in memory.

Simple Batch Systems (early 1960s)

- In serial systems
  - Machines were very expensive
  - Wasting time was not acceptable.
- To improve usage, the concept of batch OS was developed.
- The main idea is the use of software known as **monitor**.
  - The user no longer has access to machine.
- The user submits the job (tape) to the operator.
- The operator batches the jobs together sequentially, places entire batch as an input device for use by the computer.


User
User
User
User
job
job
job
job
Operator
jobs
jobs
Batch
Batch
Computer

Memory Layout for a Simple Batch System



Resident monitor

Simple Batch Systems..

- At the beginning of any job, the corresponding subroutines and functions are loaded.
- The monitor reads the jobs one at a time from the input device.
- **ALGORITHM FOR MONITOR (or Operating System)**
  - The control is passed to the user's program.
    - Processor is fetching and executing user's instructions.
  - After completion, the control is returned to the monitor program
    - Processor is fetching and executing monitor instructions.

The task performed by CPU when CPU is idle.

```
Loop  no operation
      JUMP loop
```

Features of Batch System

- The batch OS is simply a program.
- It relies on the ability of the processor to fetch instructions from various portions of main memory to seize and relinquish control.
- Hardware features:
  - **Memory protection**: While the user program is running, it must not alter the memory area containing the monitor.
    - If such is the case the processor hardware should detect the error and transfer control to monitor.
  - **Timer:** A timer is used to prevent the single job from monopolizing the system
  - **Privileged instructions**
    - Contains instructions that are only executed by monitor.
    - I/O instructions
    - If a program encounters them the control shifts through monitor..
  - **Interrupts:** It gives OS more flexibility.
    - Relinquishing control and regain control

Features of Batch System

- With batch OS, the machine time alters between execution of user programs and execution of monitor.

- Two overheads
  - Machine time is consumed by the monitor.
  - Memory is consumed by the monitor.

- Still, they improved the performance over serial systems.

Problems with the Batch System

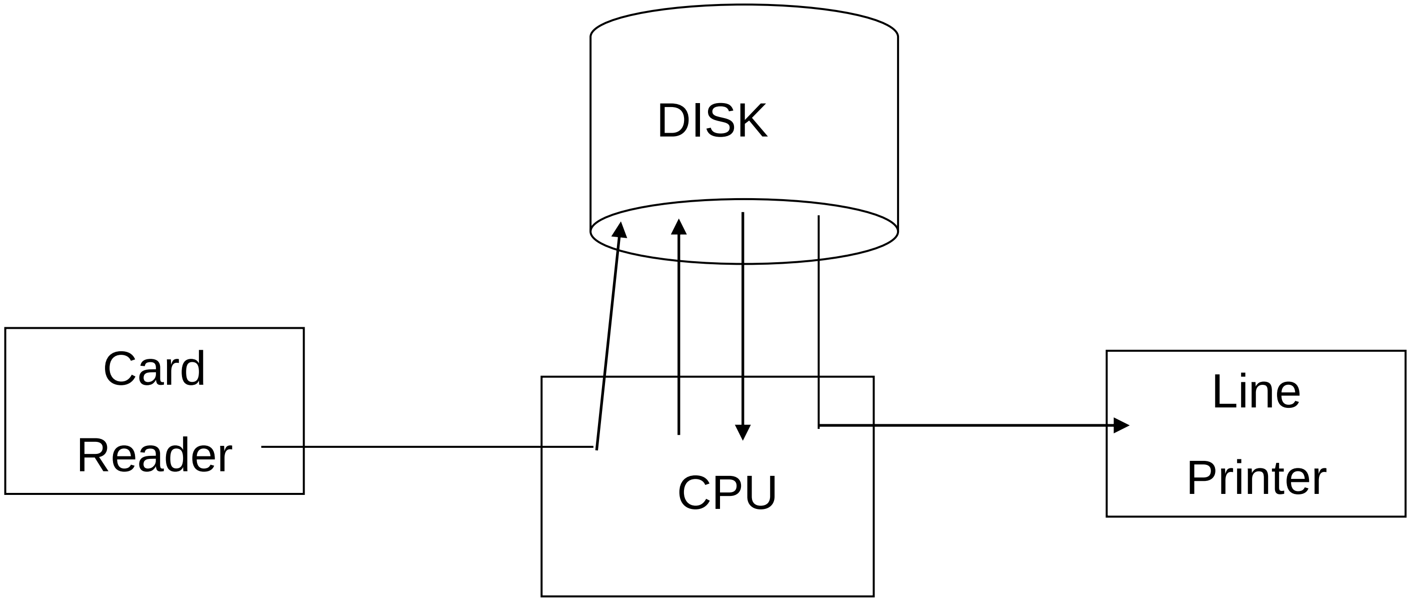
- CPU is idle
  - Speed of mechanical devices is very slower than those of electronic devices.
- CPU works in a microsecond range
  - Thousands of instructions/second
- A card reader may read 1200 cards per minute
  - (20 cards per second)
- CPU speed has increased at a faster rate.
- Tape technology improved the performance little-bit.
- Main perceived problem
  - Turn-around time: up to two days
  - CPU often **underutilized**
    - Most of the time was spent reading and writing from tape.

Resident monitor: summary

- Automatic job sequencing
  - Use of control cards
- Job control language
  - Commands
    - Mount this tape
    - Compile
    - Run
- OSs begin to be important.
  - IBM: Fortran monitor system
- Main perceived problems
  - Turn-around time
  - Inexpensive use of expensive hardware
  - CPU is still mostly idle.

Spooling

- The introduction of disk technology helped in this regard.
- Disk technology introduced the **SPOOLing (Simultaneous Peripheral Operations On-Line)**
- Considers disk as a huge buffer.
- Input comes from the disk
- Output goes to the disk.



Advantage of Spooling

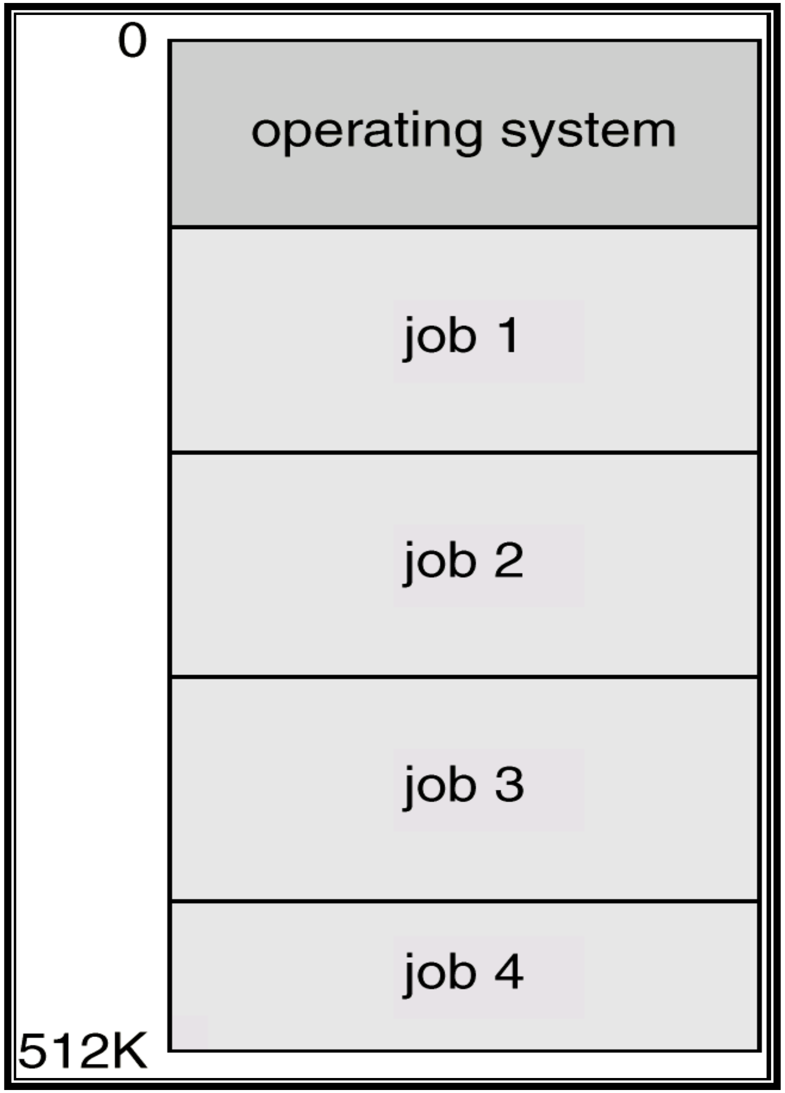
- Reading can be done in advance.
- Output can be stored on the disk.
- Spooling is also used for processing data at remote sites.
- Spooling overlaps the I/O of one job and computation of other jobs.
  - Even printing and reading can overlap.
- Spooling can keep both the CPU and the I/O devices working at higher rates.
- Disk is a random access device.

Multi-programmed Batched Systems (1960s) (or Multi tasking)

- A single user can not keep either CPU or I/O busy.

- Multiprogramming increases CPU utilization by organizing jobs such that the CPU always has one to execute.

- The OS keeps several jobs in memory at a time and CPU is multiplexed among them



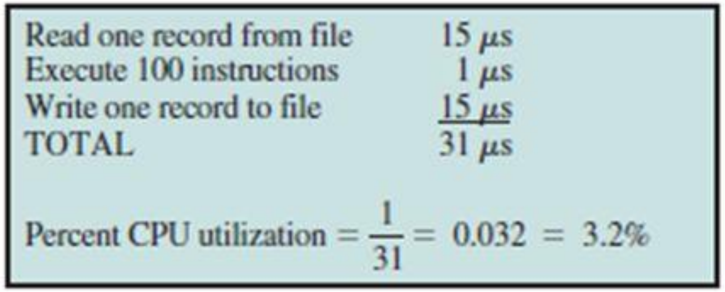
Multi-programmed Batch Systems

- If CPU is executing a job and requires a tape to be mounted
  - In a non multi-programmed system
    - CPU sits idle.
  - In a Multi-programmed system
    - CPU takes up another job.
- **Multiprogramming is the first instance when the OS started taking decisions.**
- Job scheduling is done by OS.
- Having several programs in the memory requires memory management.

Multi-programmed Batch Systems

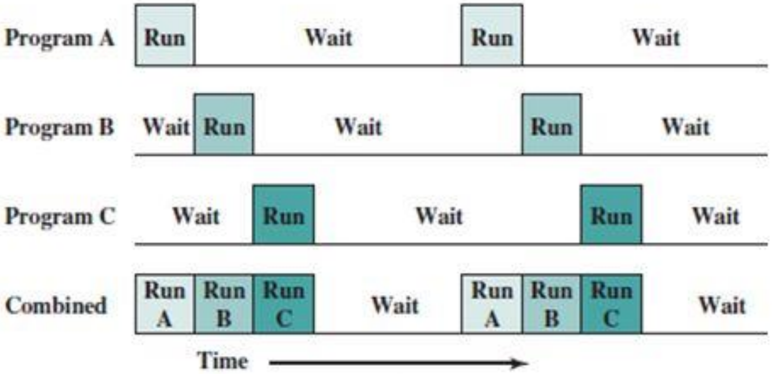
- I/O devices very slow.
- When one program is waiting for I/O, another can use the CPU.

Example:



Read one record from file	15 µs
Execute 100 instructions	1 µs
Write one record to file	15 µs
TOTAL	31 µs

$$\text{Percent CPU utilization} = \frac{1}{31} = 0.032 = 3.2\%$$



Multiprogramming with three programs

OS Features Needed for Multiprogramming

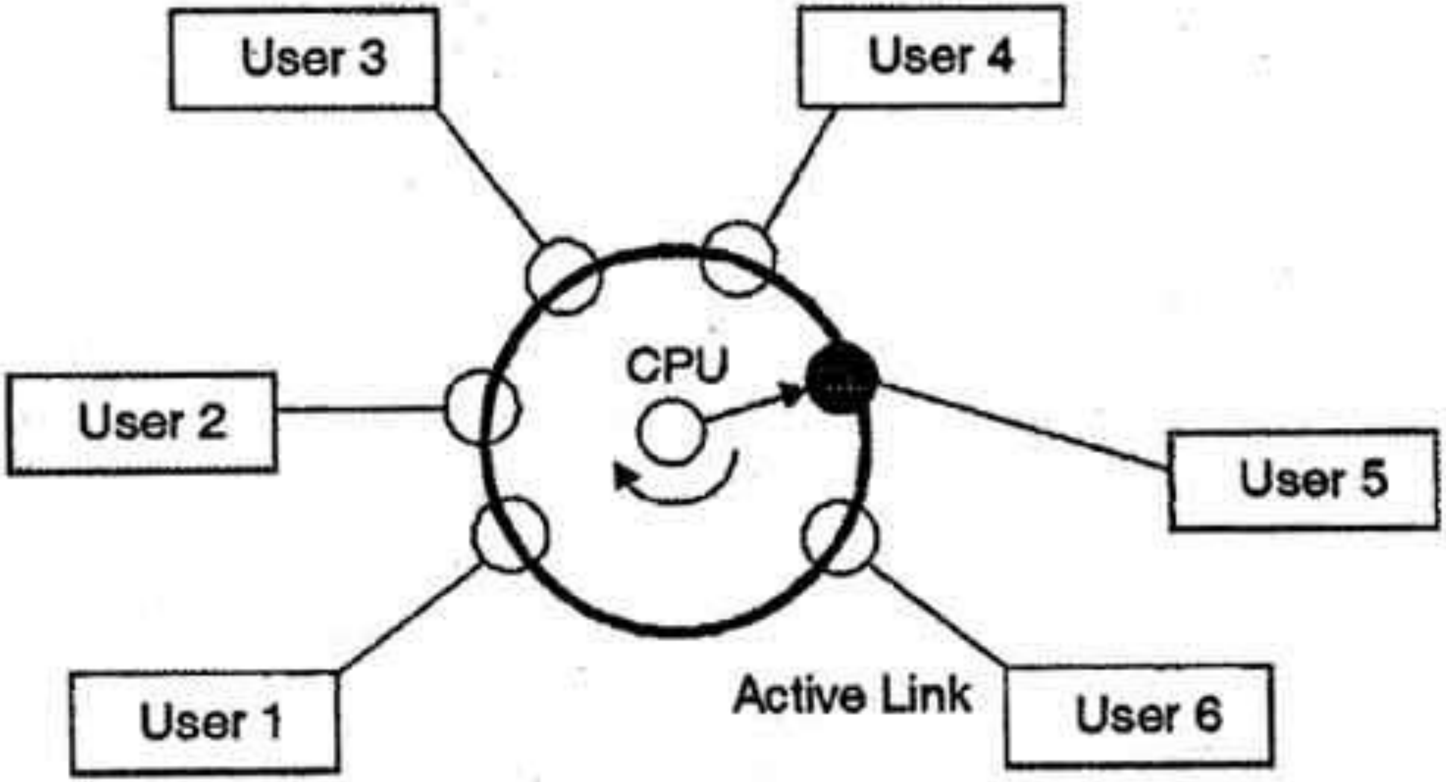
- I/O routine supplied by the system.
- Memory management –
  - the system must allocate the memory to several jobs.
- CPU scheduling –
  - the system must choose among several jobs ready to run.
- Allocation of devices.

Time-Sharing Systems–Interactive Computing

- With multiprogramming
  - Utilization is okay.
  - But, response time was a problem.
- Timesharing:
  - Programs could interact with user.
- Programs
  - Could wait for I/O for arbitrary time
    - CPU switched to another job.
    - However, resident jobs took up valuable memory
      - Needed to be swapped out to disk
      - Virtual memory.
- Time-sharing systems were developed to provide **interactive use of a computer system** at a reasonable cost.

Time-Sharing Systems–Interactive Computing

- A time sharing system uses **CPU scheduling and multi-programming** to provide each user with a small portion of a time shared computer.
- A program that is loaded into a memory and is executing is commonly known as a **process**.
- In timesharing system, a process executes for only a short time.
    - I/O is at people speeds, but OS can switch rapidly.
- A time-shared OS system allows the many users to share the computer simultaneously.
- It gives the impression that the user has own computer, whereas actually a computer is shared among many users.



Time-sharing is a technique which enables many people, located at various terminals, to use a particular computer system at the same time.

TIME SHARING SYSTEMS

About Modern OSs

- **Multiprogramming and timesharing** are the central themes of modern OSs.
- Multiprogramming and timesharing requires
  - CPU scheduling
  - Process synchronization and communication
  - Deadlock detection
  - Memory management and protection
  - Virtual memory: A program is bigger than physical memory
  - Online file systems.
  - Disk management
  - Security and protection
  - Real-time and multimedia support
- In this course, we will discuss
  - key concepts/algorithms /ideas developed since 1950 on the preceding aspects/issues.
  - networking protocols.