Operating System Notes (Lecture 1)

WHAT OPERATIONS DO

- Mainframe is big computers (To maximize resource utilization)
- Assemply Language is used to write an operating system
- Kernal - The seed (heart of the oprating system)
- For a computer to run, it needs to have an initial program to run. Bootstrap program is loaded at powerup or reboot
- One program running all the time is called the kernel
- System Call - Software may trigger an interrupt by executing a special operation
- ROM (Read-Only Memory) - All bootstrap programs are stored in the rom
- RAM (Random Access Memory) - Rewritable memory that run most of the compulter's program

Storage Name	Storage Definition
bit	Contains 0 1 values
byte	8 bits
kilobyte	1,024 bytes
Megabyte	$1,024^2$ bytes
Gigabyte	$1,024^3$ bytes
Terabyte	$1,024^4$ bytes

- DRAM (Dynamic random-Access Memory) - Main memory is implemented in a semiconductor technology

* Important *

- The operating system in such cases is designed to maximize resource utilization
- A computer system has many resources that may be required to solve a problem: CPU Time, memory space, file-storage space, I/O devices and so on

- 1. Main memory is usually too small to store all needed programs and data permanently
  2. Main memory is a volatile storage device that losses it's content when power is turned off

- CACHE - System assumes you will use it again (Most used, Most expensive) information in use is copied
- Multiprogram - runs multiple programs
- Software interrupts (exception or trap)
- Dual Mode - Operation allows OS to protect itself and other system components

Operating Systems (Lecture 4)

Account : 20
cs 4440 s 20
gbjWqWdF

cs1.calstatela.edu

Learning C

```
#include <stdio.h>
int main(void){
printf("Hello, World!");
return 0;
}
```

Editors

nano
vim
emacs

- Semantics - Meaning of operation
- Syntax - Symbol of operation

Steps:

1. nano hello.c
2. Text ...
3. ctrl X (save)
4. To compile: gcc -o hello hello.c
5. Execute: ./hello

Tips

^X
ctrl X

Filename

To Compile

gcc -o hello.c

%d
↑
format

current directory

Script

Operating System Concept Notes (Chapter 3)

Process - which is a program in execution

- Potentially, all these processes can execute concurrently, with the CPU multiplexed among them

Program Counter - Tells you the location of the next istruction

Stack - which contains temporary data (such as a function parameters, return address, and local variables

Heap - which is memory allocated during runtime ; "new" keyword lets memory allocated

- A program becomes an executable file when memory is loaded into memory

Process State

New - The process is being created

Running - Instructions are being executed

Waiting - The process is waiting for some event to occur (an I/O device)

Ready - The process is waiting to be assigned to a processor

Terminated - The process has finished execution

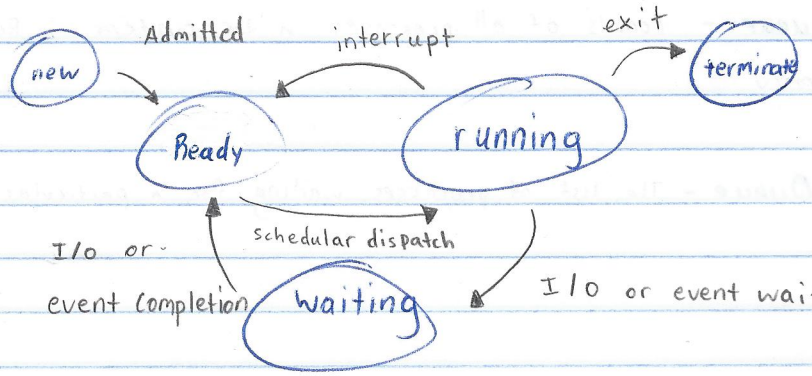
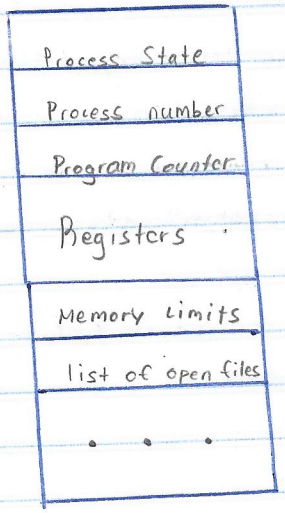


Diagram of process state

CPU Registers - accumalators, index registers, stack pointers

CPU-scheduling information - priority pointers

Memory-Management information - base and limit registers and the page tables



Process control Block (PCB)

Accounting Information - Amount of CPU and real time used

I/o status information - list of I/o devices allocated, open Files

Job queue - Consits of all processes in the system ; Residing in main memory

Device Queue - The list of processes waiting for a particular I/o device

Schedulers

Long-Term Scheduler: Selects processes from this pool and loads them into memory

Short-Term Schedular: Selects processes that are ready to execute and allocates CPU

I/O Bound Process: Spends more of it's time doing I/O than it spends doing computations

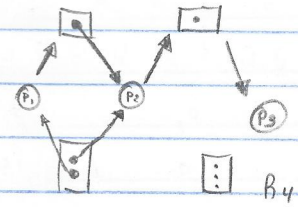
Context: The process running on the CPU so that it can restore that context when it's processing is done

Context Switch: Switching the CPU to another process

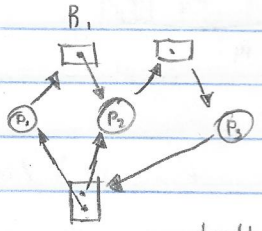
October 31, 2017 Lecture 21 Chapter 7

$R = \{R_1, R_2, R_3, R_4\}$ set { }

edges



Not deadlocked



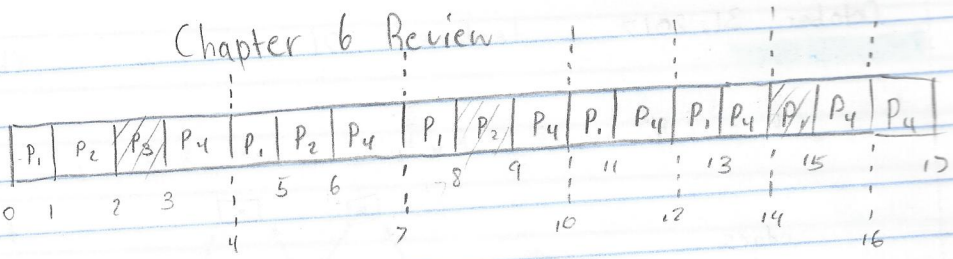
deadlock

if there is no cycle (circular), no deadlock
if there is, then there maybe deadlock

All should happen simultaneously

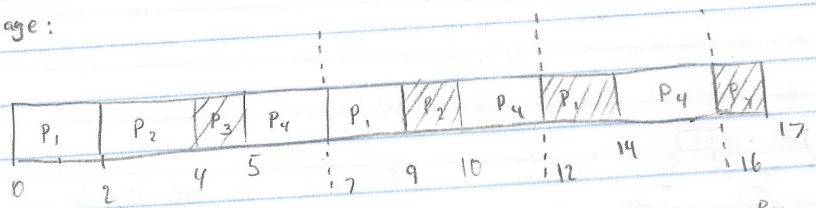
Bankers Algorithm: On Final *

Chapter 6 Review



Finished: P_3 P_2 P_1 P_4

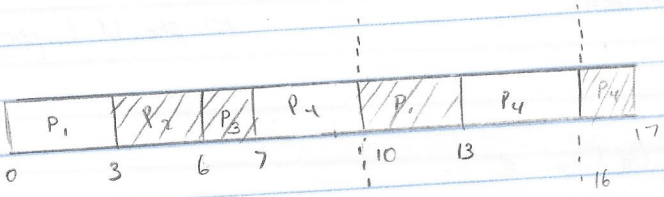
Turn Around Time Average: $(3 + 9 + 15 + 17) / 4 = 11$



Turn around time average: P_3 P_2 P_1 P_4

$(5 + 10 + 14 + 17) / 4 = 11.5$

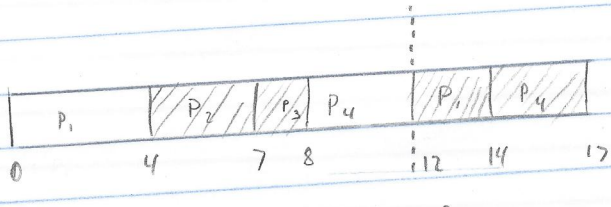
Quantum T: 3



Turn around time average: P_2 P_3 P_1 P_4

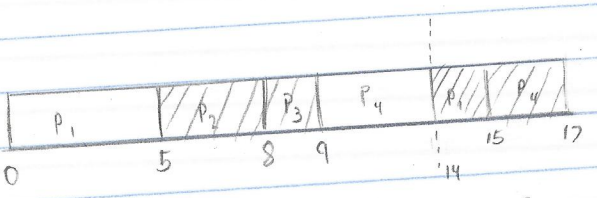
$(6 + 7 + 13 + 17) \div 4 = 10.25$

Quantum T: 4



Turn around time average: P_2 P_3 P_1 P_4

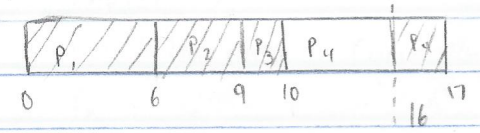
$(7 + 8 + 14 + 17) \div 4 = 11.5$



Turn around time average: P_2 P_3 P_1 P_4

$(8 + 9 + 15 + 17) \div 4 = 12.25$

Quantum T: 6



P_1 P_2 P_3 P_4

$(6 + 9 + 10 + 17) \div 4 = 10.5$

Chapter 7

SAFE STATE?

R = 12		Max need	Currently holds	Currently Need
	P_0	10	5	5
	P_1	4	2	2
	P_2	9	2	7
			9	

$12 - 9 = 3 + 2$ ← P_1 works

$5 + 5 = 10$ P_1 executes then

Sequence: $\langle P_1, P_0, P_2 \rangle$

SAFE STATE?

	Max need	Curr	Currently Need
P_0	10	5	5
P_1	4	2	2
P_2	9	3	6
		10	

$12 - 10 = 2$

$2 + 2 = 4 \geq 5, 6$

Then deadlock...

BANKERS ALGORITHM

Currently Allocated	A	B	C
P_0	0	1	0
P_1	2	0	0
P_2	3	0	2
P_3	2	1	1
P_4	0	0	2
	7	2	5

	A	B	C
	10	5	7
	7	2	5
	3	3	2

Max each needs	A	B	C
	7	5	3
	3	2	2
	9	0	2
	2	2	2
	4	3	3

Need still needs	A	B	C
	7	4	3
	1	2	2
	6	0	0
	0	1	1
	4	3	1

Chapter 7 Homework (7.3, 7.22, 7.23)

	Allocation	Max	Available	Need?	
	A B C D	A B C D	A B C D	A B C D	
P_0	0 0 1 2	0 0 1 2	1 5 2 0	0 0 0 0	x
P_1	1 0 0 0	1 7 5 0		0 7 5 0	
P_2	1 3 5 4	2 3 5 6		1 0 0 2	x
P_3	0 6 3 2	0 6 5 2		0 0 2 0	x
P_4	0 0 1 4	0 6 5 6		0 6 4 2	
	2 9 10 11				

P_0 → P_2 → P_3 → P_4

$1520 > 0000$ → $1532 > 1002$ → $2886 > 0020$ →

$1520 + 0012 = 1532$ → $1532 + 1354 = 2886$ → $2886 + 0632 = 2\,14\,11\,8$

C

$\langle P_0, P_2, P_3, P_1, P_4 \rangle$

	Allocation	Max	Available	Need?	
	A B C D	A B C D			
P_0	0 0 1 2	0 0 1 2	1 5 2 0	0 0 0 0	
P_1	1 4 2 0	1 7 5 0	− 0 4 2 0	0 3 3 0	
P_2	1 3 5 4	2 3 5 6	1 1 0 0	1 0 0 2	
P_3	0 6 3 2	0 6 5 2		0 0 2 0	
P_4	0 0 1 4	0 6 5 6		0 6 4 2	
	2 13 12 12				

deadlocked...

P_0 .... $\langle P_0, P_2, P_1, P_3, P_4$

$1100 > 0000$ $1112 > 1002$

$1100 + 0012 = 1112$ $1112 + 1354 = 2466$

Process	Allocation	Max	Need
	A B C D	A B C D	A B C D
P_0	3 0 1 4	5 1 1 7	2 1 0 3
P_1	2 2 1 0	3 2 1 1	1 0 0 1
P_2	3 1 2 1	3 3 2 1	0 2 0 0
P_3	0 5 1 0	4 6 1 2	4 1 0 2
P_4	4 2 1 2	6 3 2 5	2 1 1 3

Available

0 3 0 1

~~0301 > 2103~~

~~0301 > 1001~~

0301 > 0200

P_2 0301

+ 3121

3422

P_1 3422 > 1001

3422

+ 2210

5632

P_3 5632 > 4102

5632

+ 0510

5 11 4 2

~~5 11 4 2 > 2 1 0 3~~

~~5 11 4 2 > 2 1 1 3~~

Deadlock...

Available

(1, 0, 0, 2)

P_1 1002 > 1001

1002

+ 2210

3212

P_2 3212 > 0200

3212

+ 3121

6333

P_0 6333 > 2103

6333

+ 3014

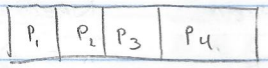
9347

$\langle P_1, P_2, P_0, P_3, P_4 \rangle$

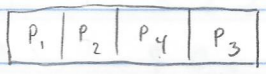
Chapter 6 Review

#Ex 6.16 pg 308

FCFS

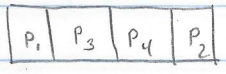


SJF

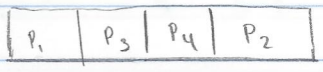


	Arrival Time	Burst
P_1	0	2
P_2	1	1
P_3	2	8
P_4	3	4

Nonpreemptive Priority



Ex:



$P_3 > P_2$ priority

RR

Bigger # the bigg

Turnaround

P	FCFS	SJF	Priority	RR
P_1	2-0=2	2-0=2	2	2
P_2	3-1=2	3-1=2	14	2
P_3	11-2=9	13	10-2=8	13
P_4	15-3=12	4	11	8

Turn Around Time = Terminated Time - Arrival Time

Waiting

P	FCFS	SJF	Priority	RR
P_1	2-2=0	0	0	0
P_2	2-1=1	1	13	1
P_3	1	5	0	5
P_4	8	0	7	4

Waiting Time = Turnaround Time - Burst Time

7.23 Homework Continued

	Allocation	Max	Allocation	Need
	A B C D	A B C D	A B C D	A B C D
P_0	2 0 0 1	4 2 1 2	3 3 2 1	2 2 1 1 x
P_1	3 1 2 1	5 2 5 2		2 1 3 1
P_2	2 1 0 3	2 3 1 6		0 2 1 3
P_3	1 3 1 2	1 4 2 4		0 1 1 2 x
P_4	1 4 3 2	3 6 6 5		2 2 3 3

P_0 3321 > 2211

3321
+ 2001
5322

P_3 5322 > 0112

5322
+ 1312
6634

P_1 6634 > 2131

6634
+ 3121
9755

< P_0, P_3, P_1, P_2, P_4 >

B. P_1 (1, 1, 0, 0)

	Allocation	Max	Available	Need
	A B C D	A B C D	3321	A B C D
P_0	2 0 0 1	4 2 1 2	-1100	2 2 1 1 x
P_1	4 2 2 1	5 2 5 2	2221	1 0 3 1
P_2	2 1 0 3	2 3 1 6		0 2 1 3
P_3	1 3 1 2	1 4 2 4		0 1 1 2
P_4	1 4 3 2	3 6 6 5		2 2 3 3

P_0 2221 > 2211 P 4221 > !

2221
+ 2001
4221

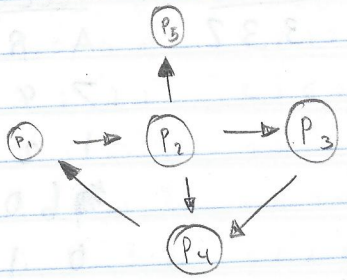
Deadlock

Practice for Quiz

	Allocation	Max	Available	Need	
	A B C	A B C	3 3 2	A B C	
P_0	0 1 0	7 5 3		7 4 3	
P_1	2 0 0	3 2 2		1 2 2	P_1
P_2	3 0 2	9 0 2		600	
P_3	2 1 1	~~222~~ 222		0 1 1	P_3
P_4	0 0 2	4 3 3		4 3 1	

3 3 2 > 1 2 2 — 5 3 2 > 0 1 1 — < P_1, P_3, P_2, P_0, P_4 >

$$\begin{array}{r} 332 \\ +200 \\ \hline 532 \end{array} \qquad \begin{array}{r} 532 \\ +211 \\ \hline 743 \end{array}$$



	Allocation	Max	Available	Need	
	A B C D	A B C D	A B C D	A B C D	
P_0	0 0 1 2	0 0 1 2	~~1 5 2 0~~	0 0 0 0	✗
P_1	1 4 2 0	1 7 5 0	1 1 0 0	0 3 3 0	
P_2	1 3 5 4	2 3 5 6		1 0 0 2	✗
P_3	0 6 3 2	0 6 5 2		0 0 2 0	✗
P_4	0 0 1 4	0 6 5 6		0 6 4 2	✗

$P_1 (0,4,2,0) \rightarrow 1420$

P_0 P_2 P_3 ... ✓

$1100 > 0000$

$$\begin{array}{r} 1100 \\ +\ 0012 \\ \hline 1112 \end{array}$$

$1112 > 1002$

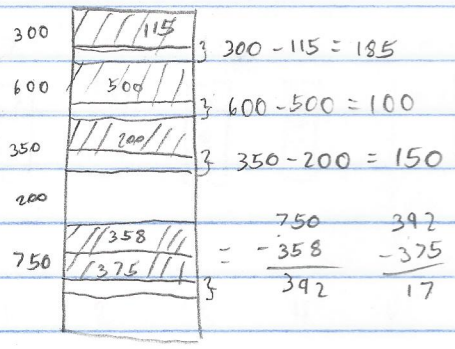
$$\begin{array}{r} 1112 \\ +\ 1354 \\ \hline 2466 \end{array}$$

$2466 > 0020$

$$\begin{array}{r} 2466 \\ +\ 0632 \\ \hline 21098 \end{array}$$

11/16/17

First Fit



Left over allocation will be set to

Segment Table

segment number	limit	base

*Final

*Google it

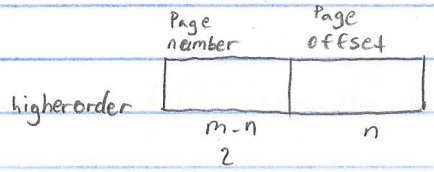
Binary Numbers:

	Page number	offset
0	0 0	0 0
1	0 0	0 1
2	0 0	1 0
3	0 0	1 1
4	0 1	0 0
5	0 1	0 1
6	0 1	1 0
7	0 1	1 1
8	1 0	0 0
9	1 0	0 1
10	1 0	1 0

0 ← page

1

Page Size: $2^n = 2^2 = 4$



Logical Address Space

$2^m = 2^4 = 16$

$m = 4$

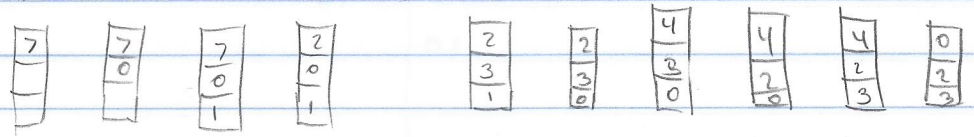
$n = 2$

$m - n = 2$

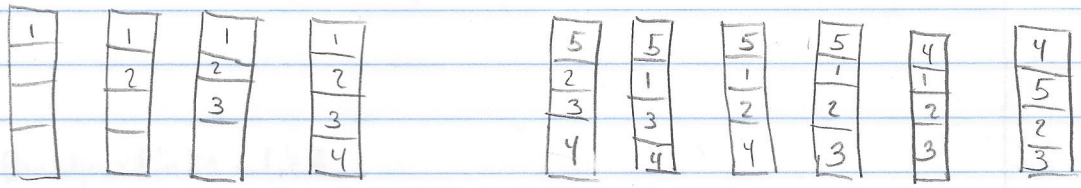
Page Offset = Page size

Page folds FIFO

7 0 1 2 0 3 0 4 2 3 0 3 2 1

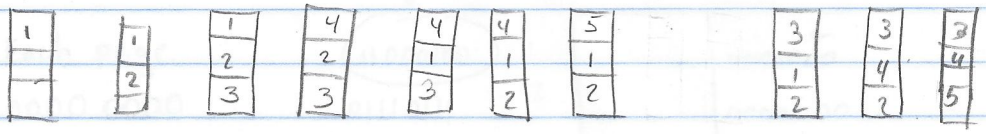


1 2 3 4 1 2 5 1 2 3 4 5



LRU 3 Frames

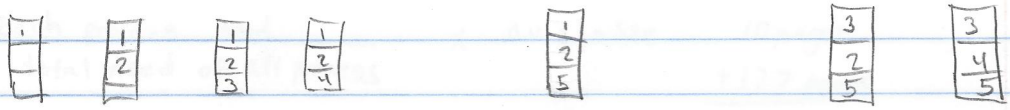
1 2 3 4 1 2 5 1 2 3 4 5



#10

Optimal Page-replacement Algorithm * Best Algorithm

1 2 3 4 1 2 5 1 2 3 4 5



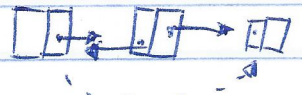
Problem: You don't know the future

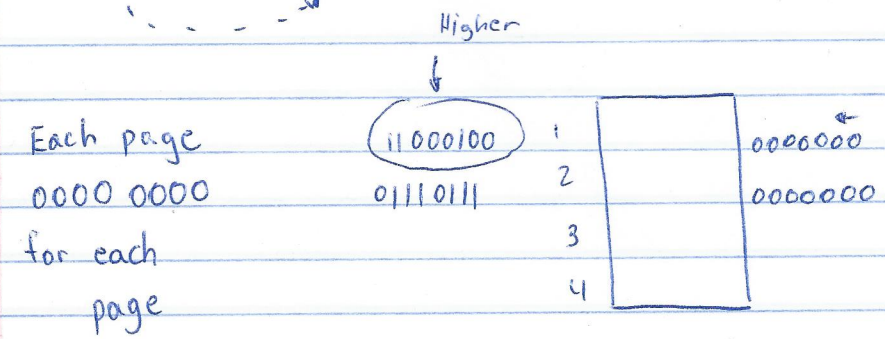
* Initial loading

November 28, 2017

Page Numbers	offset
01	00
	01
	99
02	00

Doubly - Linke - List





Each page
0000 0000
for each
page

Allocation Algorithm

$\frac{\text{Each process Need}}{\text{total need of all process}}$ x avialable

10 pages
+127 pages
137 pages

$\frac{10}{137} \times 62 = 4...$

$\frac{127}{137} \times 62 = 57$