CS342 Operating Systems

Homework 3

Ata Seren

21901575

Section-2

1) In case of sequence P1, P2

P1 allocation <=10 P2 allocation <=5

In case of sequence P2, P1

P2 allocation <=12 P1 allocation <=3

As result:

(0…3, 0...12) and (4…10, 0…5) allocations are safe which is 94 allocations in total.

2)

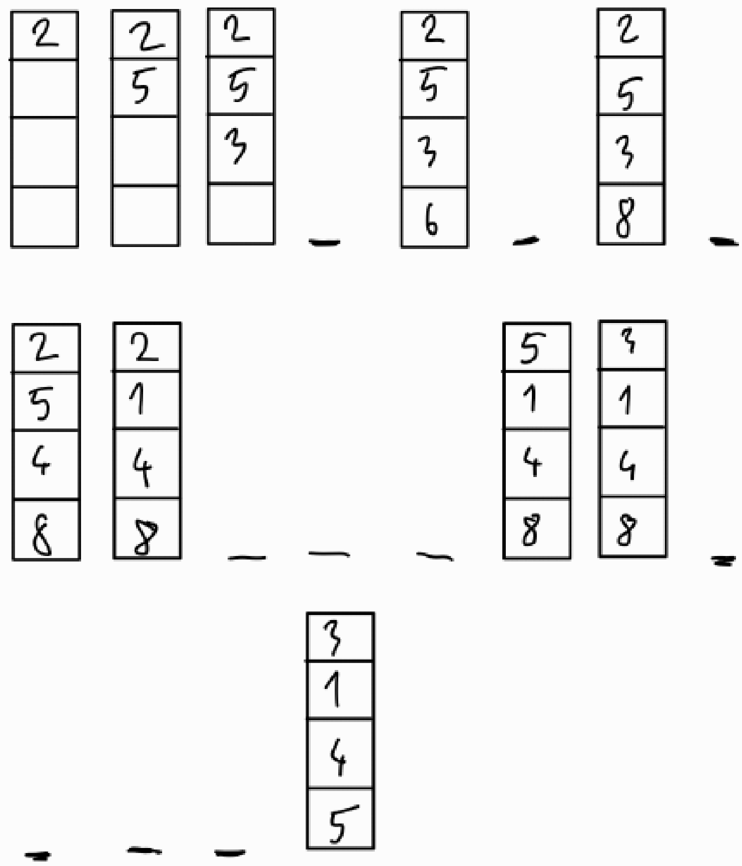
	Alloc	Request	Available
	A B C	A B C	A B C
P1	1 0 0	0 0 1	3 3 0
P2	3 0 1	0 4 0	
P3	0 2 2	3 1 2	
P4	1 0 1	2 3 0	
P5	2 1 0	3 3 3	

After allocations, available resources are 3A, 3B and 0C.

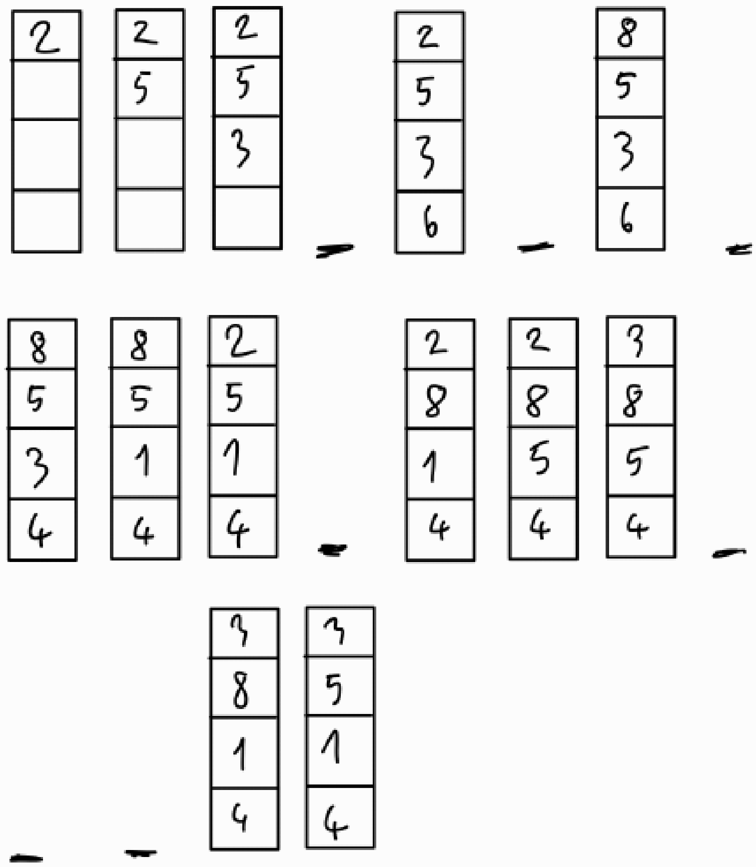
At this moment, only P4 can be completed. After completion, available resources become 4A, 3B and 1C. Then only P1 can be completed and available resources become 5A, 3B and 1C. After that, no processes can be completed because available resources can’t meet any of the requests of P2, P3 or P5. There is a deadlock at the moment.

3)

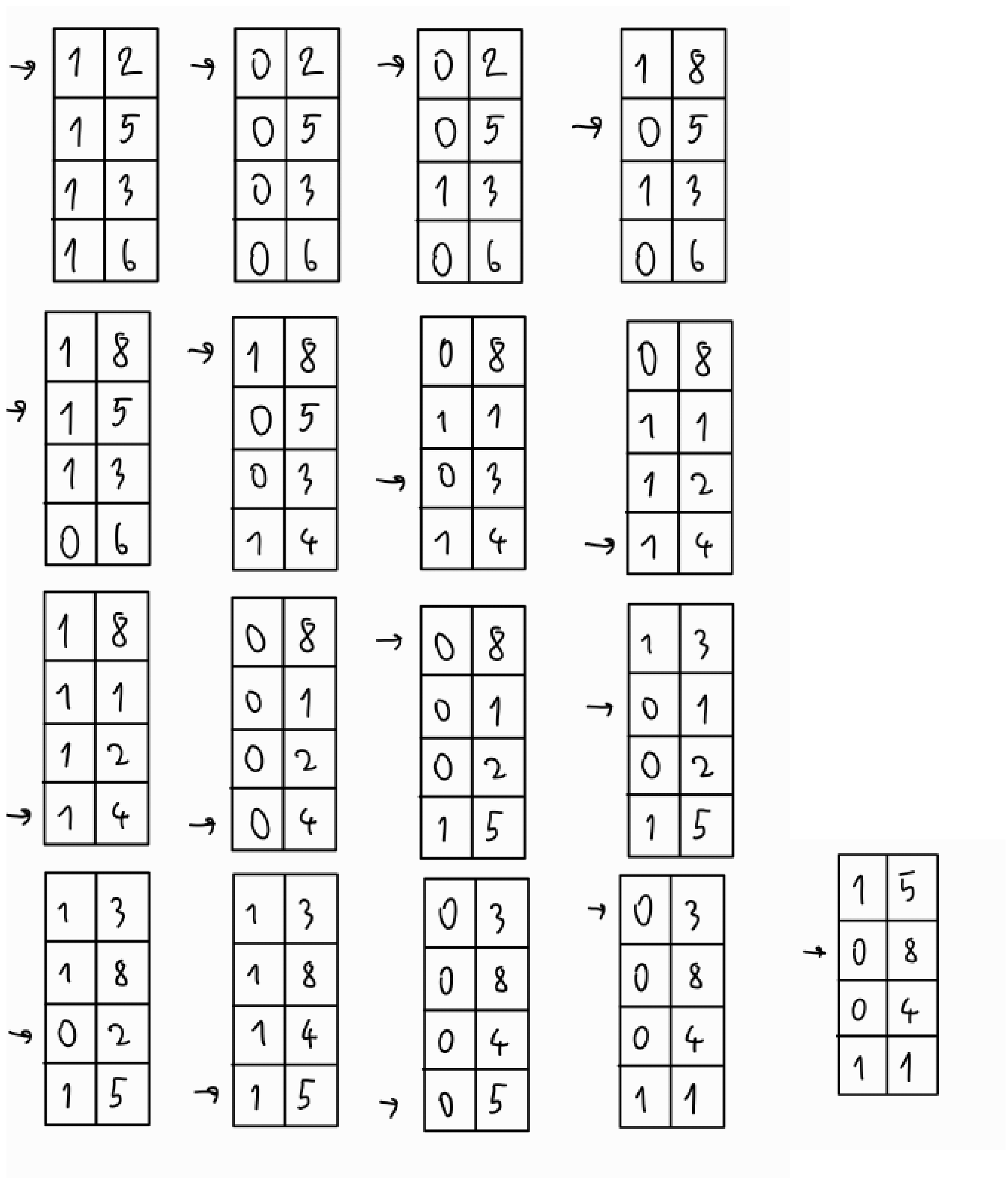
a) Optimal:



b) LRU:



c) Clock:



4)

a)

36 bit addresses can address 2^36 bytes = 64 GB

2^36/2^14 = 2^22 entries

2^22 * 8 bytes (size of an entry) = 2^25 bytes = 32MB

b)

Each second level page table cover: 2^11*16KB = 32MB

Number of second level page tables needed according to the layout:

(64MB + 128MB + 32MB)/32MB = 7 tables

Number of entries in one table: 2^11 = 2048 entries

RAM space for second level page tables: 2048*8*7 = 112KB

RAM space for first level page table: 2^11*8 = 16KB

Final result: 112KB + 16KB = 128KB

c)

Number of physical frames: 256 MB/16KB = 16384 frames = 2^14 frames

Final result: 2^14 *8 = 2^17 = 128 KB

5)

4KB/8 bytes = 512 entries in a block

a)

12 direct pointers address 12 blocks

1 single-indirect pointer addresses 512 = 2^9 blocks

1 double-indirect pointer addresses 512*512 = 2^18 blocks

1 triple-indirect pointer addresses 512*512*512 = 2^27 blocks

Maximum file size = 4KB*(12 + 2^9 + 2^18 + 2^27) = 537921584 KB

b)

For 30KB: 30/4 = 7.5 8 data blocks are needed.

1 index block with single-indirect pointer is enough to keep addresses of data blocks.

For 256KB: 256/4 = 64 data blocks are needed.

1 index block with single-indirect pointer is enough to keep addresses of data blocks.

For 15MB: 15MB/4KB = 3840 data blocks are needed.

3840/ 512 = 7.5   8 second-level index blocks are needed to keep addresses of data blocks. 1 top level index block is enough to keep address of second-level index block. 9 index blocks are needed in total.

For 512MB: 512MB/4KB = 131072 data blocks are needed.

131072/512 = 256 second-level index blocks are needed to keep addresses of data blocks. 1 top level index block is enough to keep addresses of second-level index blocks. 257 index blocks are needed in total.

For 32GB: 32GB/4KB = 8388608 data blocks are needed.

8388608/512 = 16384 third-level index blocks are needed to keep addresses of data blocks.

16384/512 = 32 second-level index blocks are needed to keep addresses of data blocks.

1 top level index block is enough to keep addresses of second-level index blocks. 1 + 32 + 16384 = 16417 index blocks are needed in total.

c) ????

6)

a)

$T_{IO} = T_{seek} + T_{rotation} + T_{transfer}$

Time per rotation = (1/7200)*60*1000 ≈ 8.3 ms

$T_{rotation}$ = 8.3/2 ≈ 4.16 ms

$T_{transfer}$ = 4KB/ (80MB/s) = 0.04882812 ms

T_{IO} = 5 + 4.16 + 0.04882812 = 9.20882812 ms

I/O operations per second ≈ 108.591450179

I/O data rate = 434.365800716 KB/s

b)

Time per rotation = (1/7200)*60*1000 ≈ 8.3 ms

$T_{rotation}$ = 8.3/2 ≈ 4.16 ms

$T_{transfer}$ = 10MB/ (80MB/s) = 0.125 s

T_{IO} = 5 + 4.16 + 125 = 134.16ms

I/O operations per second ≈ 7.45378652355

I/O data rate = 74.5378652355 MB/s

7)

a)

Steady-state random read throughput = NxR MB/s

Time per rotation = (1/15000)*60*1000 ≈ 4 ms

$T_{rotation}$ = 4/2 ≈ 2 ms

$T_{transfer}$ = 4KB/ (100MB/s) = 0.0390625 ms

T_{IO} = 4 + 2 + 0.0390625 = 6.0390625 ms

I/O operations per second ≈ 165.588

I/O data rate ≈ 662.354 KB/s ≈ 0.6469 MB/s = R

Steady-state random read throughput = 5.8221 MB/s

b)

Steady-state random read throughput = (N-1) x S MB/s

Time per rotation = (1/15000)*60*1000 ≈ 4 ms

$T_{rotation}$ = 4/2 ≈ 2 ms

$T_{transfer}$ = 100MB/ (100MB/s) = 1000 ms

T_{IO} = 4 + 2 + 1000 = 1006 ms

I/O operations per second ≈ 0.994

I/O data rate ≈ 99.4 MB/s = S

Steady-state sequential read throughput = 894.632MB/s

8) RAID 4 can recover from single-disk failure. Therefore, if there will be a second disk failure before the recovery of first one, total recovery won't be possible.

Mean time to one of the (9) disks will fail: 50000/9

Prob. of one of the (9) disks will fail: 9/50000

Mean time until one of the remaining disks fails is 50000/8

Prob. of second disk failure before recovery: 48/(50000/8). (8 because we have 8 disks left.)

Prob. of both of these events occur: (9/50000)*(48/(50000/8))

Mean time to data loss (MTTDL): $50000^2/(9*8*48)$

9)

There are 64 pages in a block and 40 of them are invalid. They will just be deleted. The remaining 24 pages must be written to SRAM. After deleting whole block, we can write the desired pages to the block.

Time for:

Reading 24 pages = 24*40 microseconds

Writing them to SRAM = 24*200 microseconds

Deleting the block = 4 milliseconds

Writing desired 10 block = 10*200 microseconds

In total= 11760 microseconds

10)

a)

```
monitor access{
condition cv;
int pidSum = 0;

void request(int i) {
	while ( pidSum + i > M ) {
		cv.wait();
		}
	pidSum = pidSum + i;
	}

	void release (int i)
	{
		sum = sum – i;
		cv.broadcast();
	}
}
```

b)

```
monitor access2{
pthread_mutex_init pidLock;
condition cv;
int pidSum = 0;

void request(int i){
	pthread_mutex_lock(&pidLock);
	pidSum = pidSum + i;
```

```
	while ( pidSum + i > M ) {
		cv.wait();
	pthread_mutex_unlock(&pidLock);
}

void release(int i){
	pthread_mutex_lock(&pidLock);
	pidSum = pidSum - i;
	cv.broadcast();
	pthread_mutex_unlock(&pidLock);
	}
}
```

11)

There are 10 different possible outputs. By inspecting that some values must be printed before or after some values, sequences can be determined.

Outputs:

1. 60 10 80 40 30 45
2. 60 10 40 80 30 45
3. 60 10 40 30 80 45
4. 60 40 30 10 80 45
5. 60 40 10 80 30 45
6. 60 40 10 30 80 45
7. 40 60 10 80 30 45
8. 40 60 10 30 80 45
9. 40 60 30 10 80 45
10. 40 30 60 10 80 45