Lecture 18 CS 347/333 13.9.22

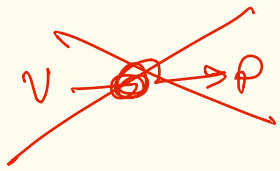
Midsem: 15th Thu. 11 am. – 1 pm. LA 201, 202

⊛ memory management

- eviction — free, return/exit
- swapping — memory pressure
- allocation — malloc, mmap

v 2 p

proc → sz



on page fault need a free page

no free page.

- usually, OSes handle this pro-actively. [kswapd]

⊛ when to evict?
Which pages to evict?
where to allocate on VA?
how much to free?

⊛ eviction of physical pages (swapping)

- page replacement policy
  eviction

⇒ challenges: (i) semantic gap — process priority is unknown to OS
importance

(ii) size of metadata.

(iii) once a VA is issued OS cannot intercept.

→ mov 0xabcde964, %eax

pte flags — access bit.

- a bit that is set by the MMU on page access.
- OS checks / resets this bit to determine recency / frequency of usage / access.

---

policies.

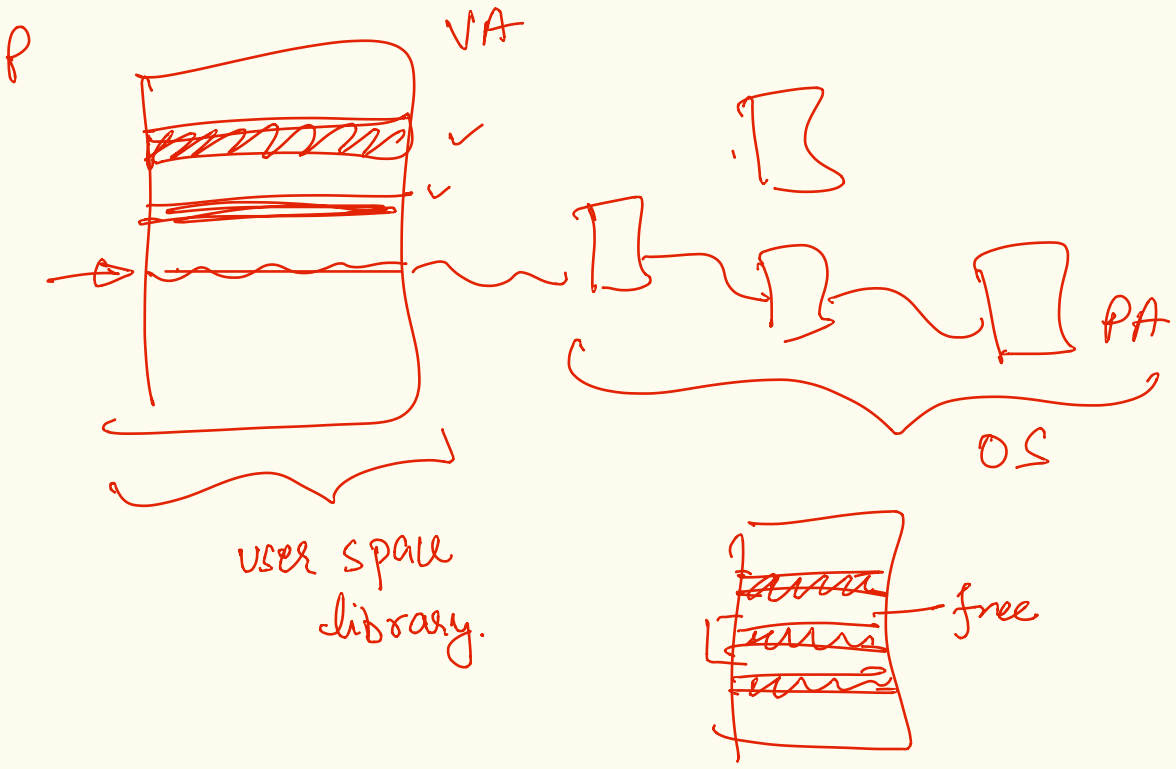
1. FIFO — first in first out — first physical page mapped is evicted first.
2. LRU — least recently used
3. LFU — least frequently used.

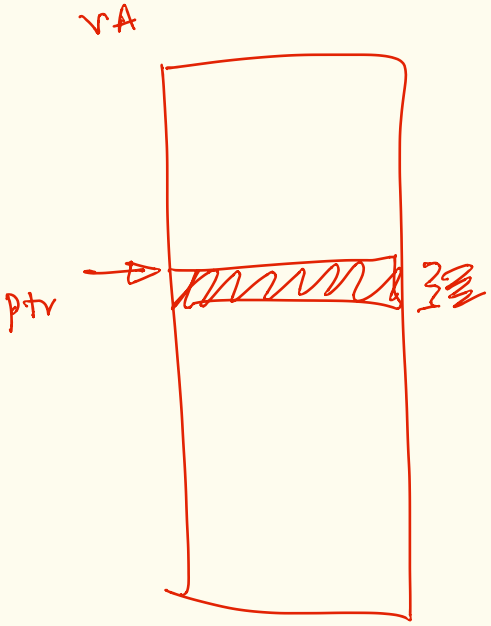
HW numericals

✳ malloc & free

- user process call ⇒ system call.

```
void * malloc (size);
```

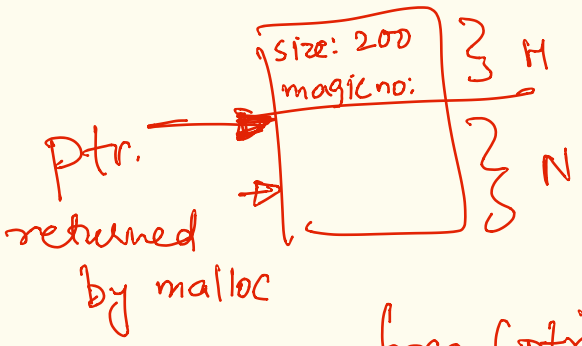





```
ptr = (char*) malloc (10 * sizeof(char));
}
free (ptr);
```

malloc (N) ⇒ VA allocation of size N+H

H: size of header.



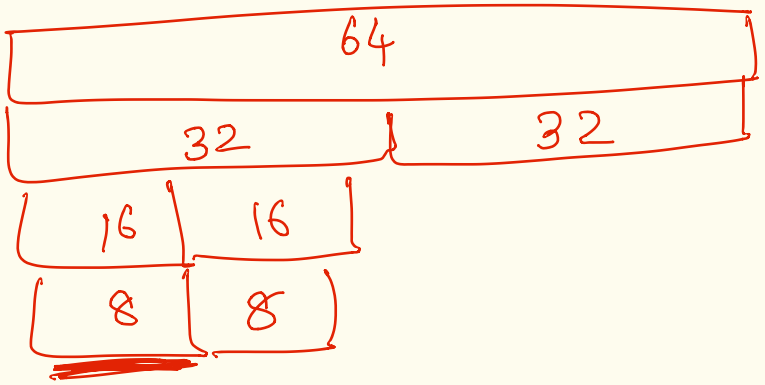
magic no: 123456

free (ptr) ⇒ step back 1 offset
verify magic no.
step back 1 offset
get size
update VA usage map
release (N+H) on VA
system call to update page table

Buddy allocator

all allocations on a VA happen in powers of 2

allocate the smallest unit of power of 2 that fits request.



splitting & coalescing.