# Systems 3
Memory Management II

Marcel Waldvogel

Department of Computer and Information Science
University of Konstanz

Winter 2019/2020

Chapter Goals

- Going beyond physical memory
- How to simplify memory management for the kernel?
- How can this be represented with the existing page table structure?
- How to select which pages to replace on memory pressure?
- What are the (logical) segments of a program?

Memory development

| 1:1 Computer:Program | Partitioning | Virtual Addresses and Pages | Virtual Memory |
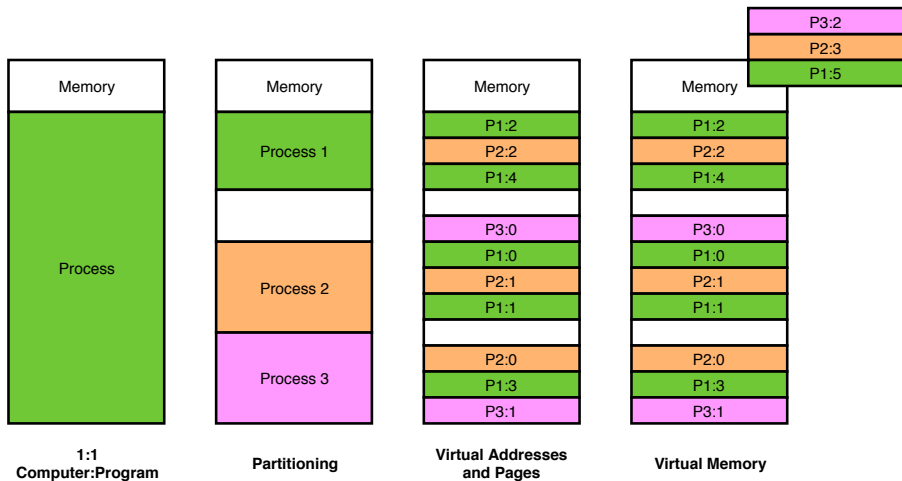
Evolution

Present Bit

Index	Present	Modified	Frame / Info
0	1	1	1234
1	1	0	2600
2	0	-	File #123, block 883
3	0	-	File #123, block 884
4	1	0	1536
5	0	-	Really invalid
...	...	...	...

Overcoming Memory Limits

Approach	Mechanism
Layout	Increase
Static	Swapping
Paged (virtual memory)	Paging

Which page should be replaced?

The three most simple replacement algorithms are:

- Optimal replacement
- Not recently used (NRU) replacement
- First-in, first-out (FIFO) replacement

But there are more algorithms. See next slides.
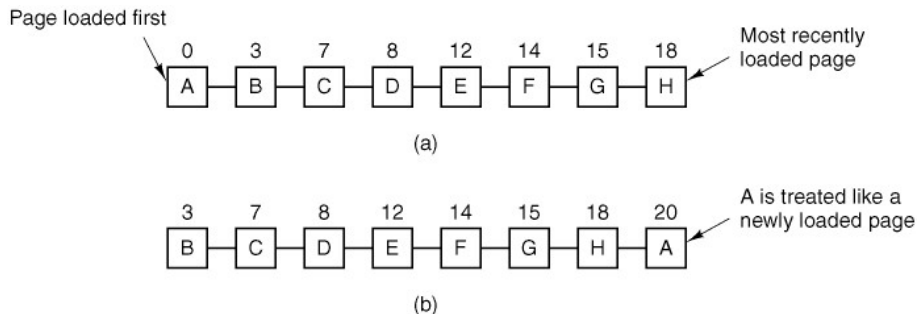
Second Chance Replacement

Page loaded first

0	3	7	8	12	14	15	18	Most recently loaded page
A	B	C	D	E	F	G	H	

(a)

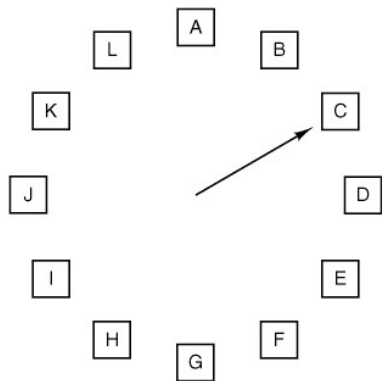| 3	7	8	12	14	15	18	20	A is treated like a newly loaded page
B	C	D	E	F	G	H	A	

(b)

Figure: Operation of second chance. (a) Pages sorted in FIFO order. (b) Page list if a page fault occurs at time 20 and A has its R bit set. The numbers above the pages are their loading times. (Tannenbaum fig. 4-14)

Clock Page Replacement

When a page fault occurs,
the page the hand is
pointing to is inspected.
The action taken depends
on the R bit:
  R = 0: Evict the page
  R = 1: Clear R and advance hand

Figure: The clock page replacement algorithm. (Tannenbaum fig. 4-15)

What is the difference between Second Chance and Clock Page
Replacement?

Least Recently Used

Assumption: Pages that have not been used for ages will probably remain unused for a long time.

Can be implemented in

- Hardware
- Software

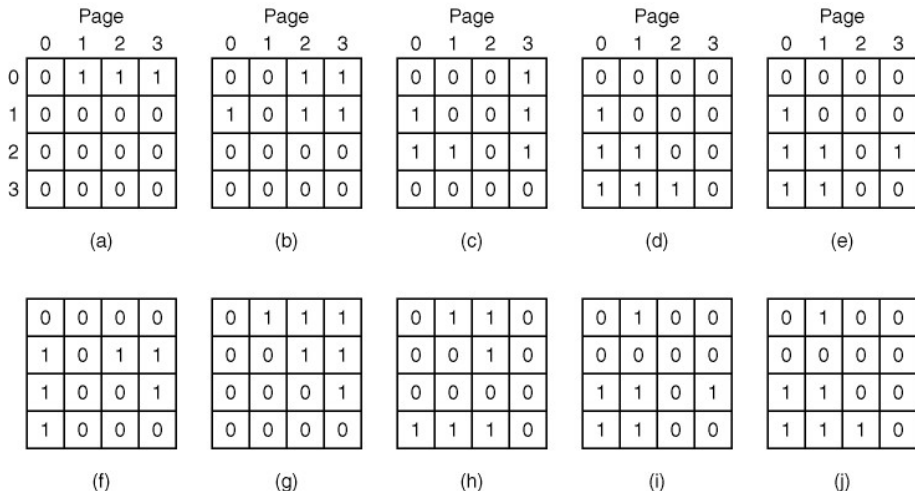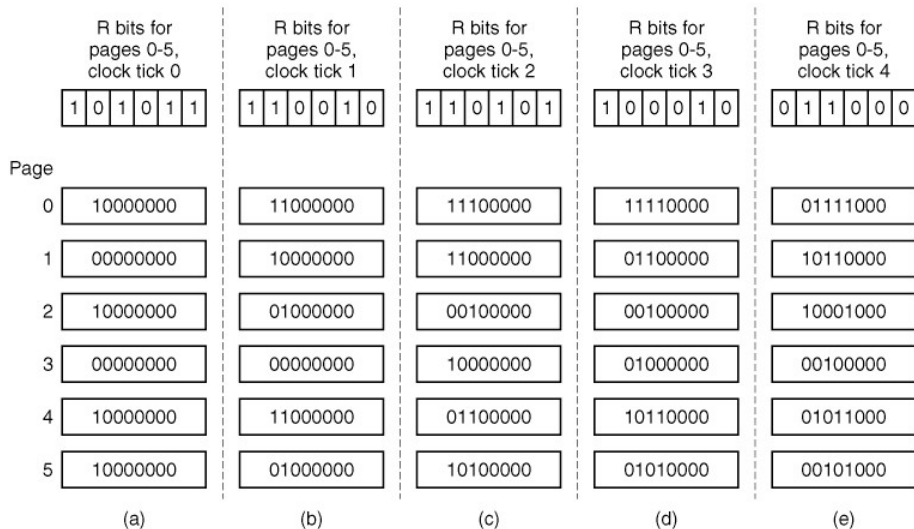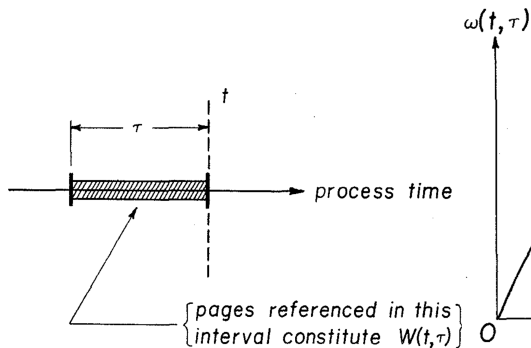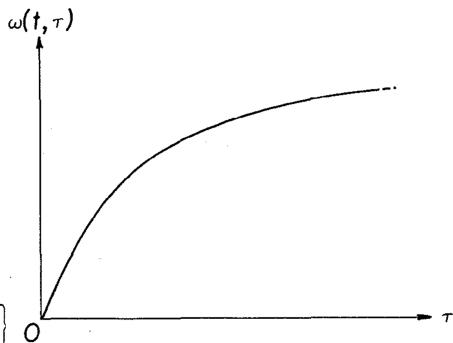Advantages? Disadvantages?

Simulating LRU

Figure: LRU using a matrix when pages are referenced in order 0, 1, 2, 3, 2, 1, 0, 3, 2, 3. (Tannenbaum fig. 4-16)

Simulating LRU

	R bits for pages 0-5, clock tick 0	R bits for pages 0-5, clock tick 1	R bits for pages 0-5, clock tick 2	R bits for pages 0-5, clock tick 3	R bits for pages 0-5, clock tick 4
	1 0 1 0 1 1	1 1 0 0 1 0	1 1 0 1 0 1	1 0 0 0 1 0	0 1 1 0 0 0
Page 0	10000000	11000000	11100000	11110000	01111000
1	00000000	10000000	11000000	01100000	10110000
2	10000000	01000000	00100000	00100000	10001000
3	00000000	00000000	10000000	01000000	00100000
4	10000000	11000000	01100000	10110000	01011000
5	10000000	01000000	10100000	01010000	00101000
	(a)	(b)	(c)	(d)	(e)

Working set[1]

Fig. 2. Definition of $W(t, \tau)$

Fig. 3. Behavior of $\omega(t, \tau)$

This locality results in a (slowly) growing working set. Memory areas, which have not been recently used are less likely to be accessed again soon and could be moved to slower memory.

---

[1]Denning, Peter J. (1968). "The working set model for program behavior". Communications of the ACM. 11(5):323–333
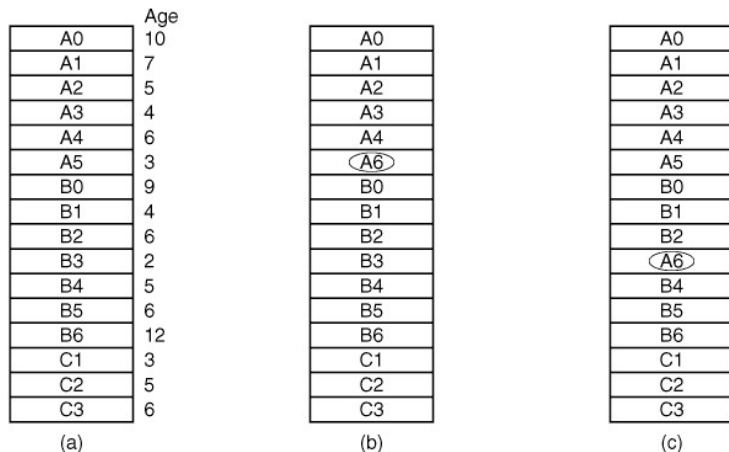
Local vs. Global

Figure: Local vs. global page replacement. (a) Original configuration. (b) Local page replacement. (c) Global page replacement. (Tannenbaum fig. 4-19)
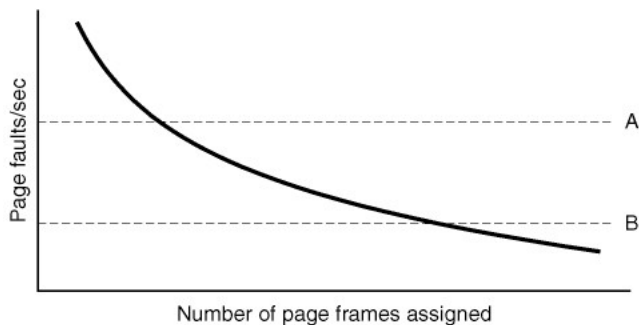
Page Fault Frequency

Figure: Page fault rate as a function of the number of page frames assigned. (Tannenbaum fig. 4-20)

Page size

Page size criteria

- What are arguments for large pages?
- What are arguments for small pages?
- What is a useful range?
- Are there applications with different criteria?

Memory Layout of C Programs

high address		command-line args, env vars, ...
	stack	
	↓	
	↑	
	heap	
	uninitialized data (**bss**)	
	initialized **data**	
	code (**text**)	
low address	illegal	