Principles of Operating Systems

CS 446/646

3. Memory Management

a. Goals of Memory Management

b. Partitioning

c. Linking & Loading

- ✓ From object codes to executable in memory
- ✓ Loading: binding logical references to physical addresses
- ✓ Linking: weaving logical addresses together

d. Simple Paging & Segmentation

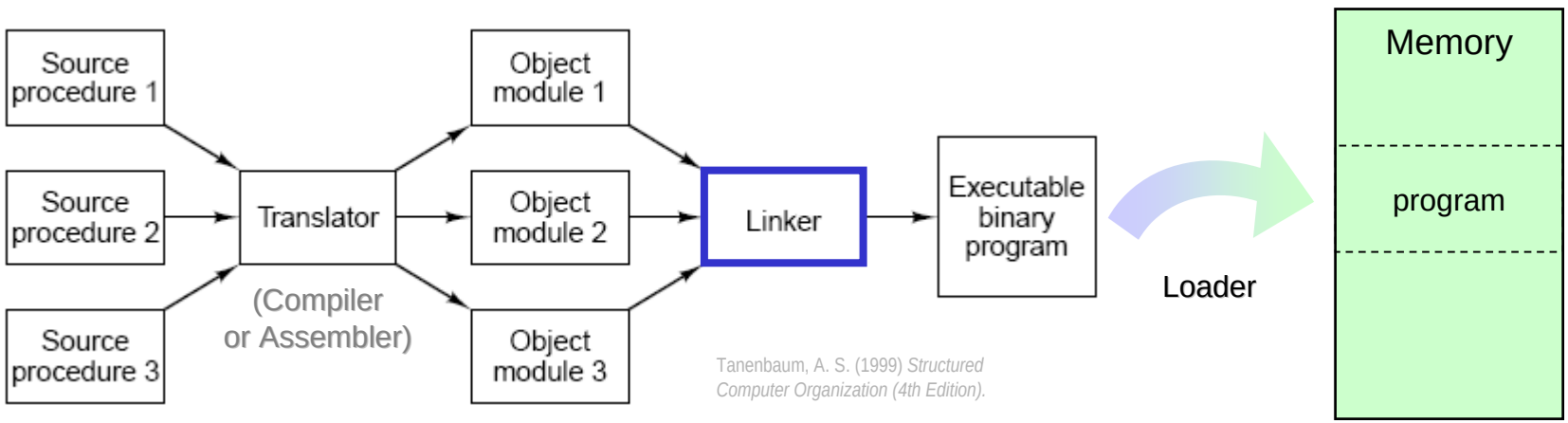
e. Virtual Memory

f. Page Replacement Algorithms

3.c Linking & Loading

From object codes to executable

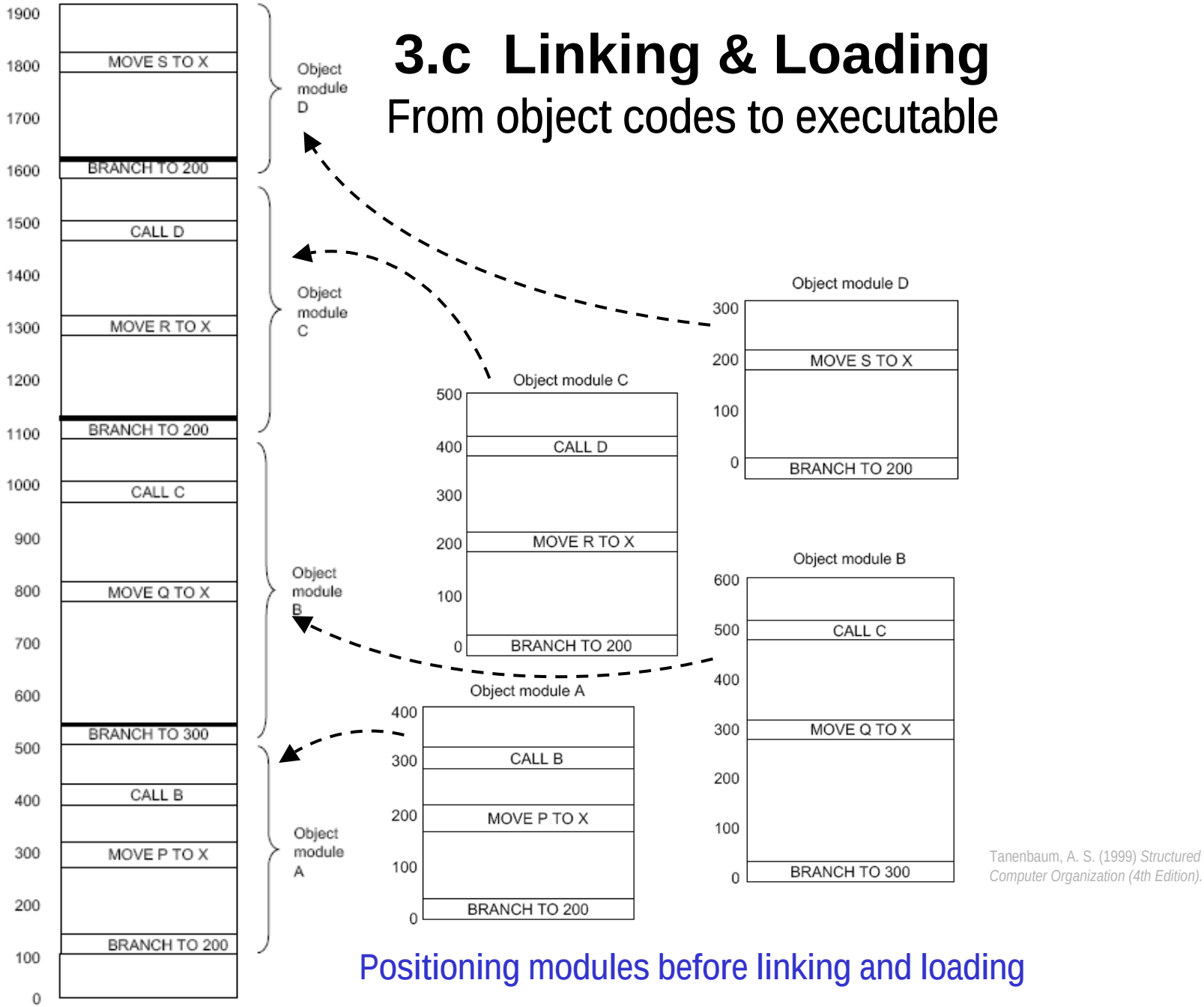
- Linkers and loaders
  - ✓ a **linker** or "link editor" is a program that takes a collection of object modules (created by compilers or assemblers) and combines them into a single executable program
  - ✓ a **loader** places the linked program in memory, possibly translating (relocating) addresses on the way



From source code to memory, via translating, linking and loading

3.c Linking & Loading

From object codes to executable

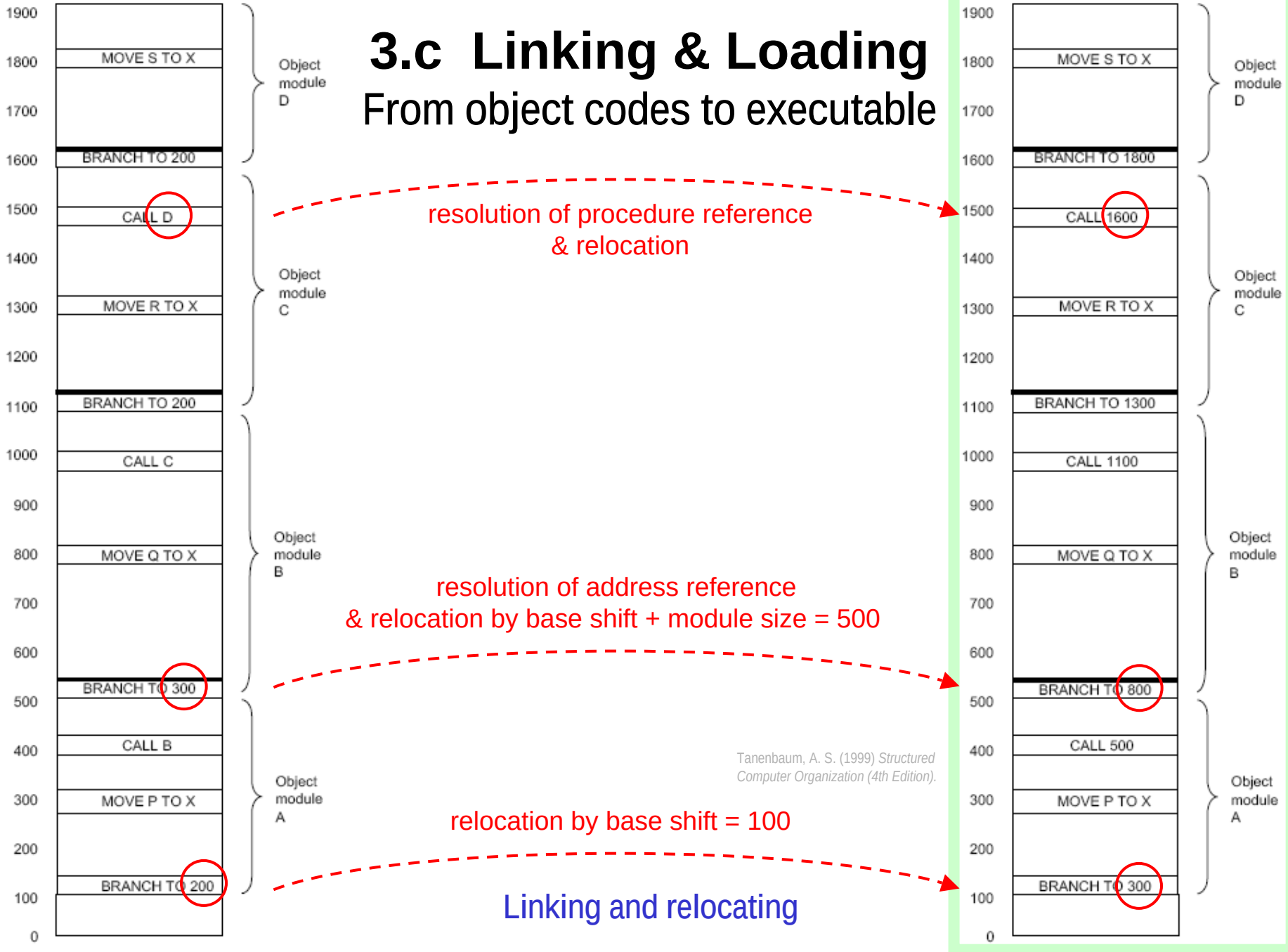


Tanenbaum, A. S. (1999) *Structured Computer Organization (4th Edition).*

Positioning modules before linking and loading

3.c Linking & Loading

From object codes to executable



Tanenbaum, A. S. (1999) *Structured Computer Organization (4th Edition).*

3.c Linking & Loading

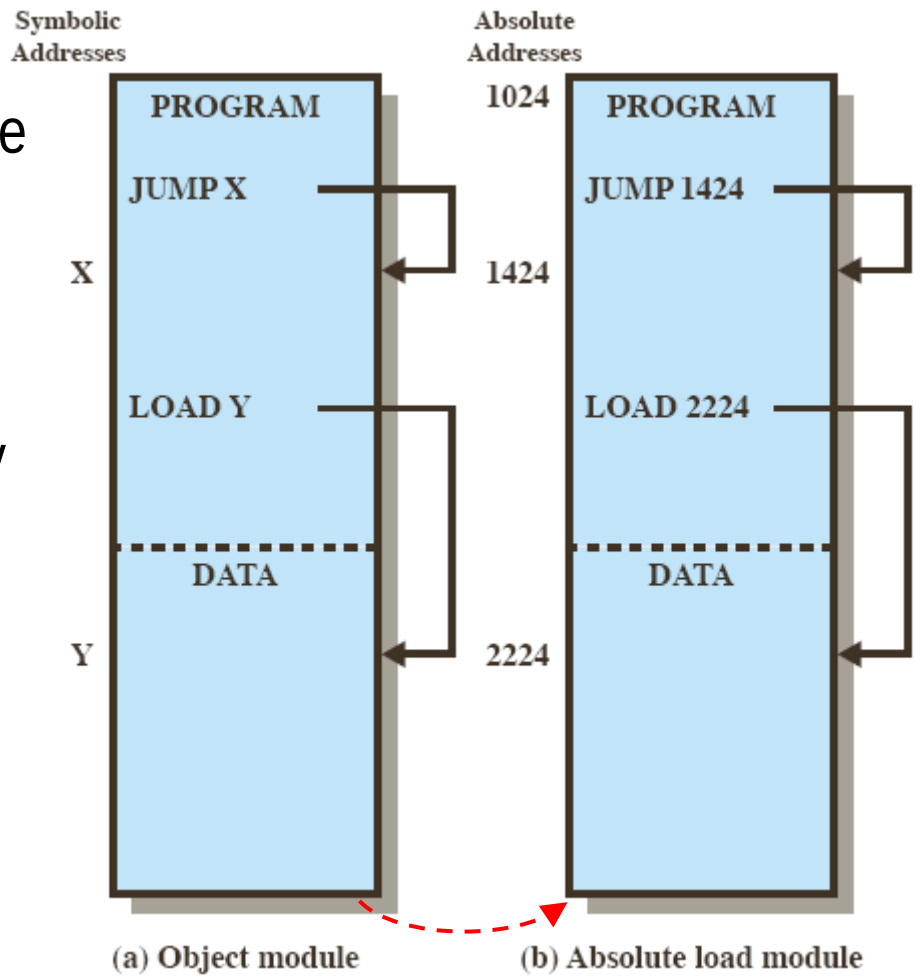
Loading: binding logical references to physical addresses

- Loading involves binding instructions and data to physical memory addresses
  - ✓ once an executable is finished compiling or is stored on disk, the loader places it in memory
  - ✓ modern systems allow a user process image to reside in any part of physical memory
  - ✓ three approaches to loading:
    - 👎 absolute loading = binding can be done beforehand, at compile time *(writing once)*
    - ✌ relocatable loading = binding is done <u>at</u> load time *(rewriting)*
    - 👍 dynamic runtime loading = binding is postponed until execution time *(not writing)*

3.c Linking & Loading

Loading: binding logical references to physical addresses

- Absolute loading
  - ✓ requires that a given module always be loaded into the same location in memory
  - ✓ thus, the compiler can bind symbolic addresses directly to absolute addresses
  - ✓ this was the case of the `.COM` format programs in MS-DOS
  - → *not acceptable: prevents swapping and/or multi-tasking, etc.*



Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

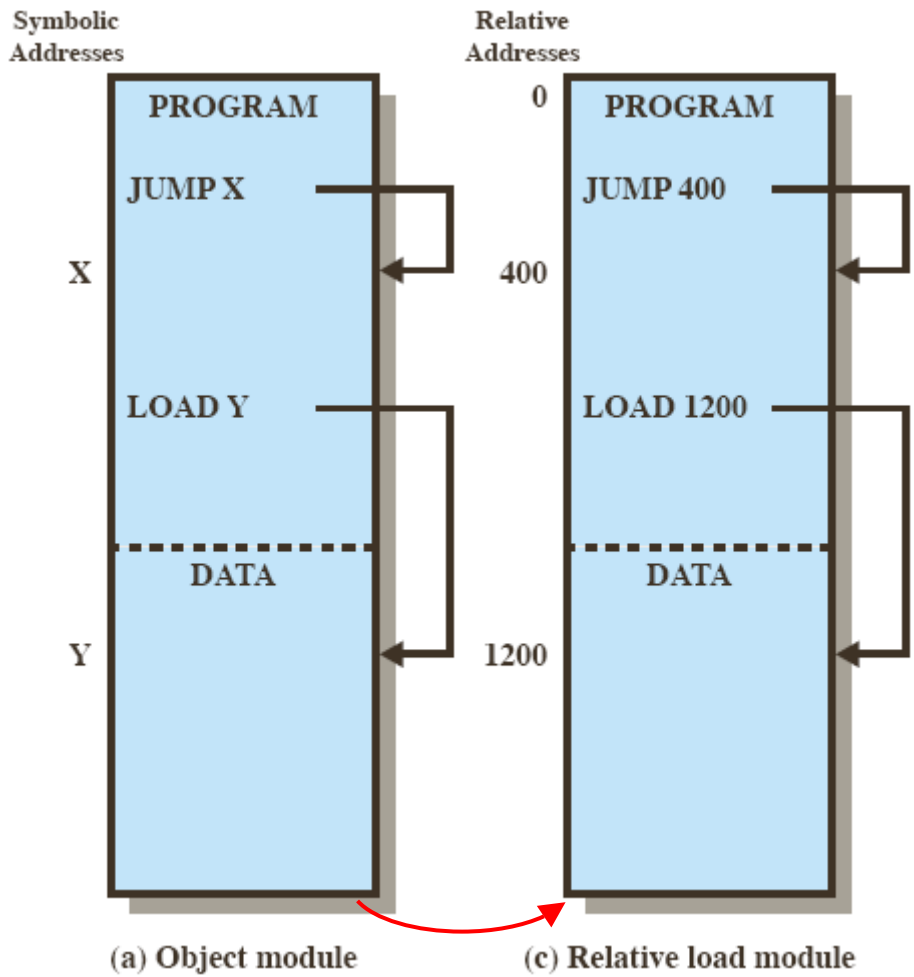
Absolute load module

3.c Linking & Loading

Loading: binding logical references to physical addresses

Relocatable loading

- ✓ we need modules that can be located and relocated anywhere in memory
- ✓ for this, the compiler must produce relative addresses
- ✓ then the task of the loader is basically to add one or several fixed offset(s) to all address references
- ✓ problem: swapping in and out requires delocating and relocating every time



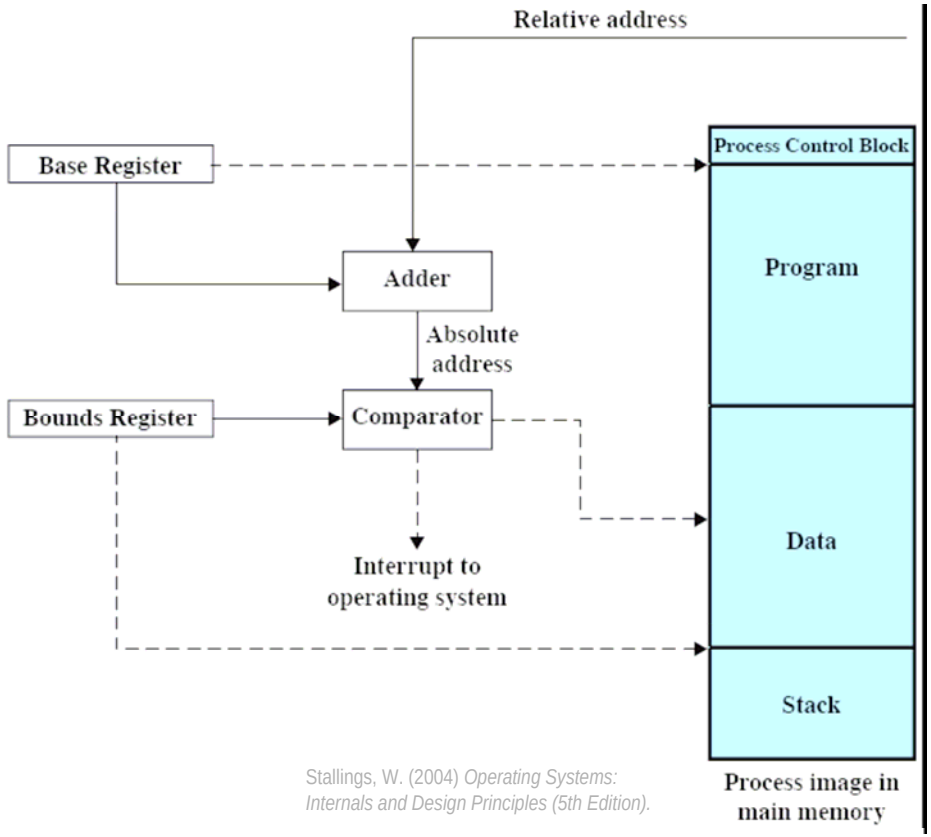
Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Relocatable load module

3.c Linking & Loading

Loading: binding logical references to physical addresses

- Dynamic runtime loading
  - ✓ physical address binding does not happen until the very last moment, when the instruction is executed
  - ✓ done by special processor hardware (combined with protection)
  - ✓ gives the most freedom



Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

Hardware support for relocation

3.c Linking & Loading

Linking: weaving logical addresses together

- Static linking

3.c Linking & Loading

Linking: weaving logical addresses together

- Dynamic load-time linking

3.c Linking & Loading

Linking: weaving logical addresses together

- Dynamic runtime linking

Principles of Operating Systems

CS 446/646

3. Memory Management

a. Goals of Memory Management

b. Partitioning

c. **Linking & Loading**

- ✓ From object codes to executable in memory
- ✓ Loading: binding logical references to physical addresses
- ✓ Linking: weaving logical addresses together

d. **Simple Paging & Segmentation**

e. **Virtual Memory**

f. **Page Replacement Algorithms**

Principles of Operating Systems

CS 446/646

3. Memory Management

a. Goals of Memory Management

b. Partitioning

c. Linking & Loading

d. Simple Paging & Segmentation

- ✓ Paging
- ✓ Hardware support for paging
- ✓ Segmentation

e. Virtual Memory

f. Page Replacement Algorithms

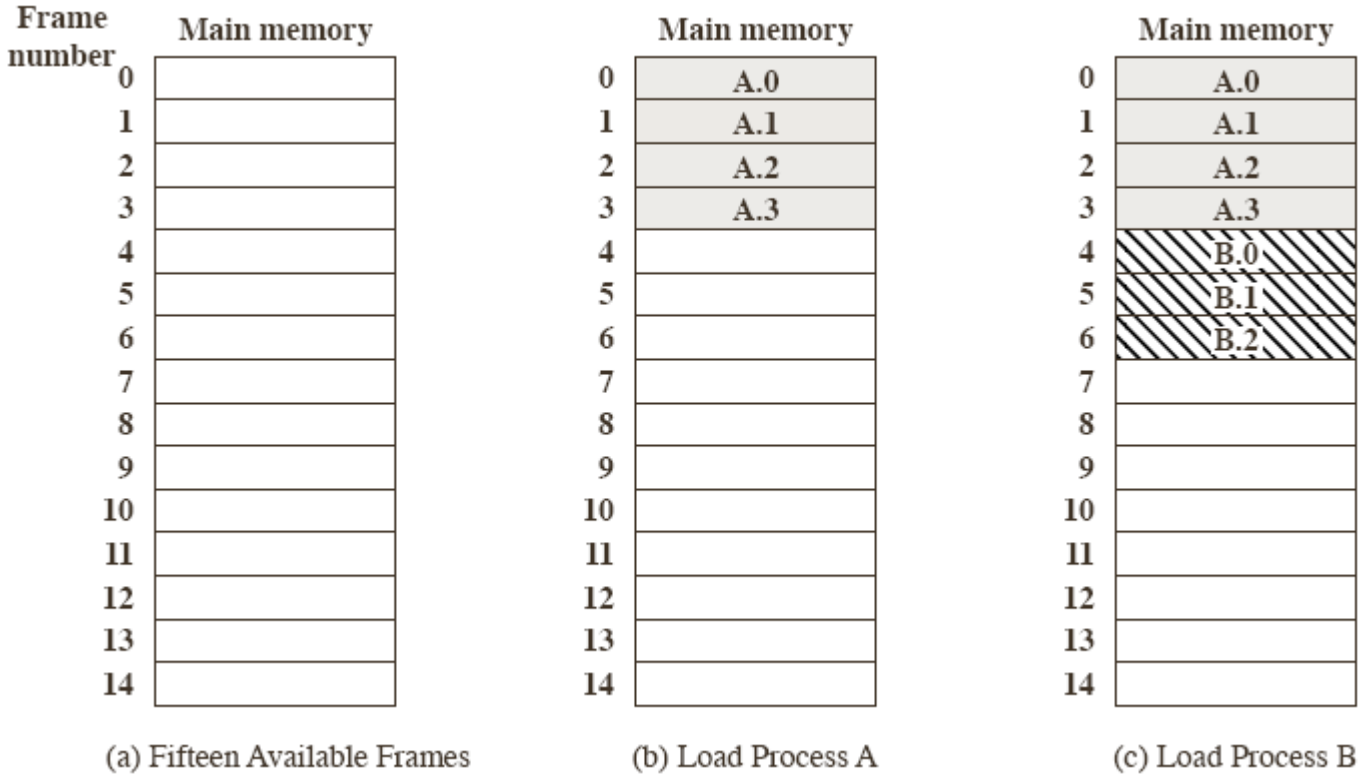
3.d Simple Paging & Segmentation

Paging

- (Sub)divide and conquer
  - ✓ new idea: partition process images, too, so that their physical domain doesn't have to be contiguous anymore
    - memory is partitioned into small, equal-size chunks
    - process images or also subdivided into the same size chunks
  - ✓ the chunks of a process are called **pages** and chunks of memory are called **frames**
  - ✓ the O/S maintains a page table for each process

3.d Simple Paging & Segmentation

Paging



(a) Fifteen Available Frames

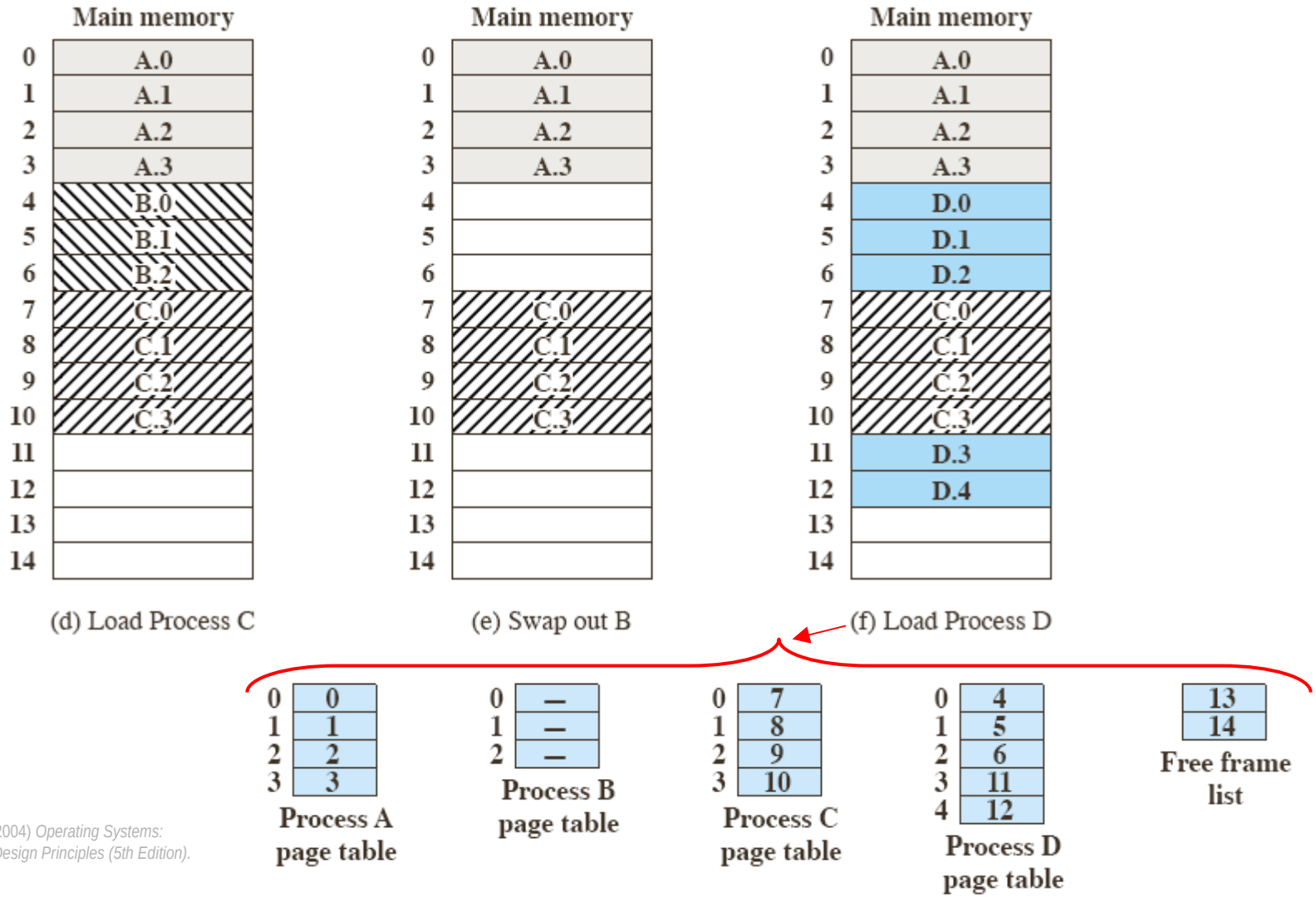
(b) Load Process A

(c) Load Process B

Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

3.d Simple Paging & Segmentation

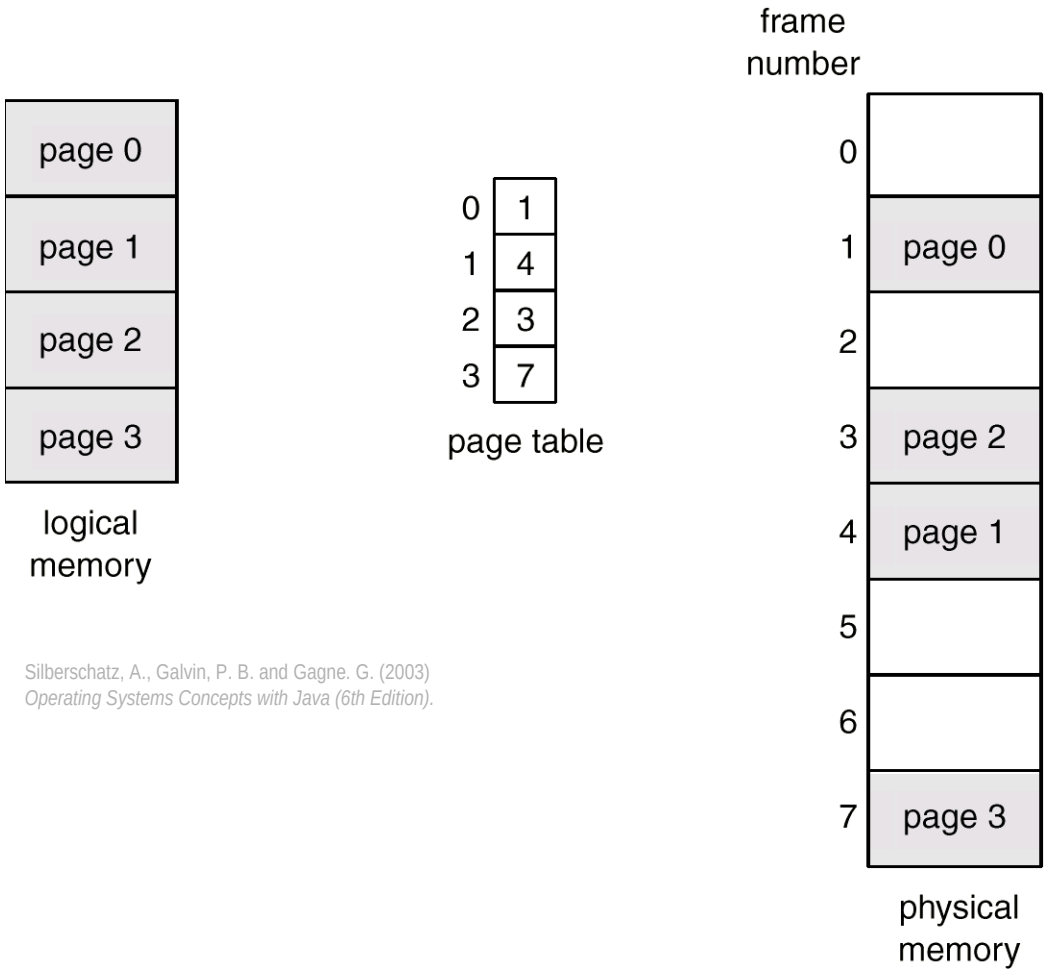
Paging



Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

3.d Simple Paging & Segmentation

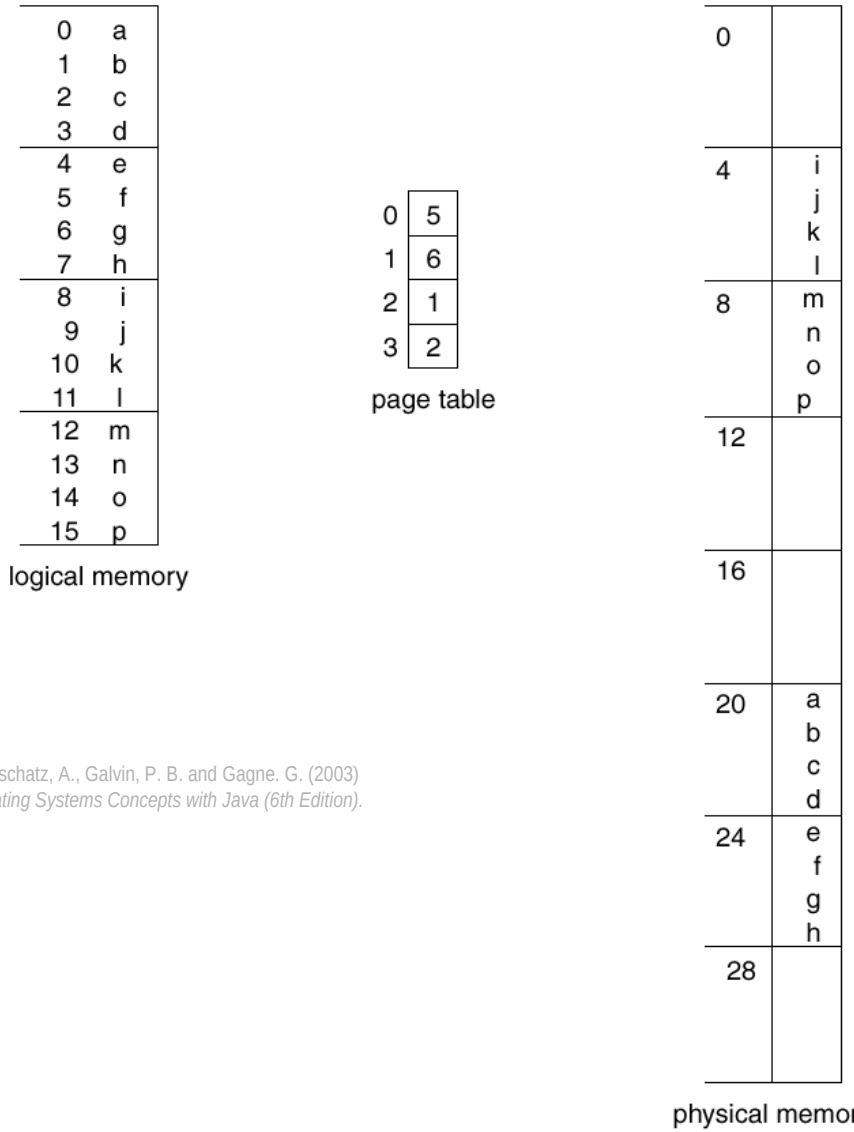
Paging



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

3.d Simple Paging & Segmentation

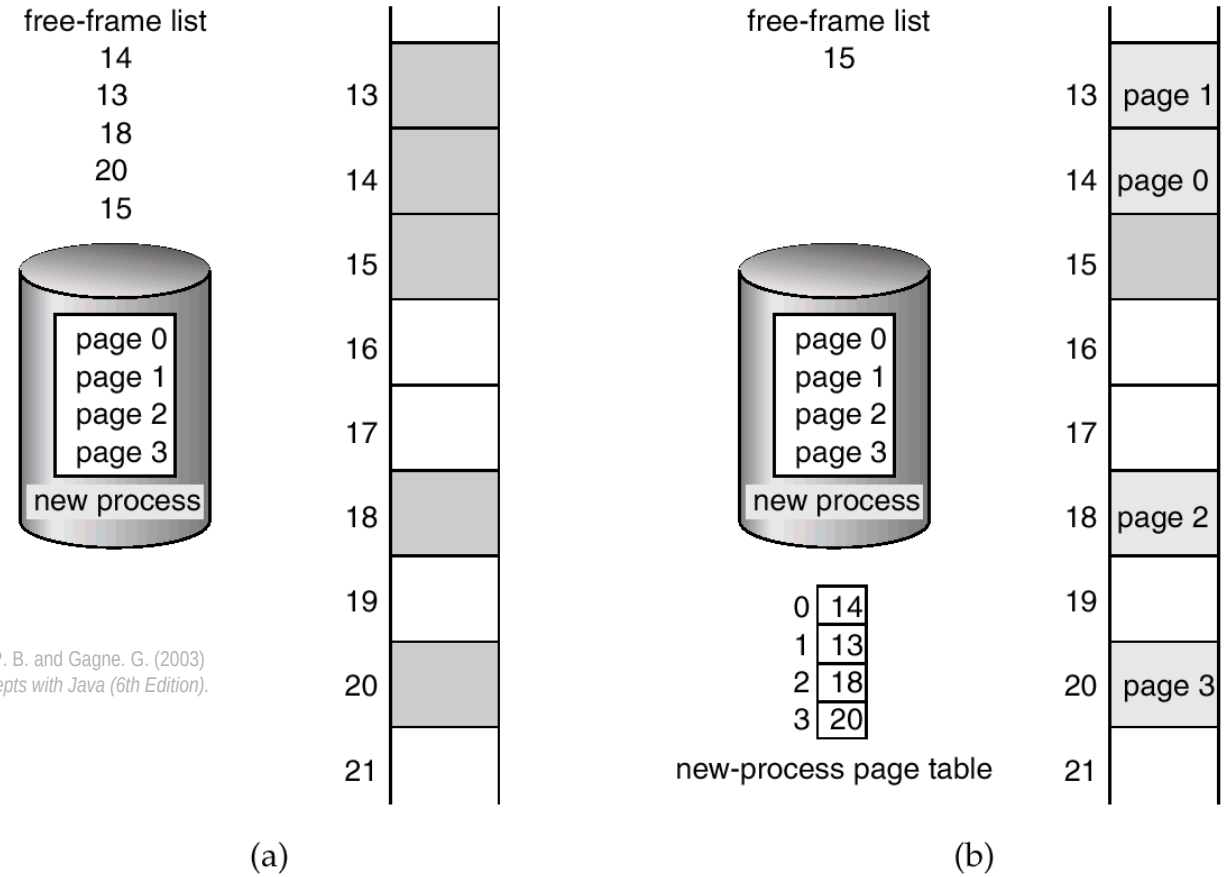
Paging



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

3.d Simple Paging & Segmentation

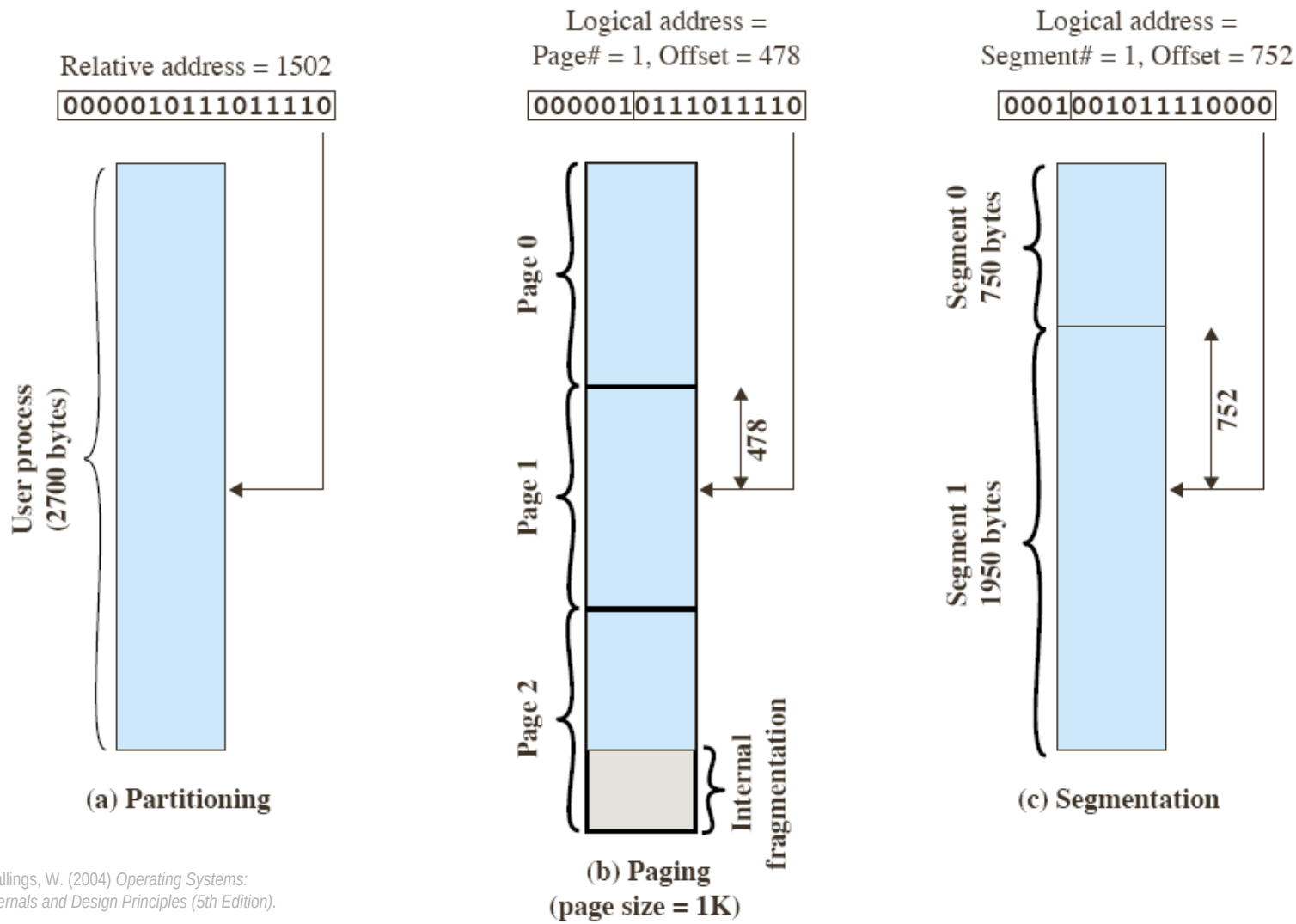
Paging



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

3.d Simple Paging & Segmentation

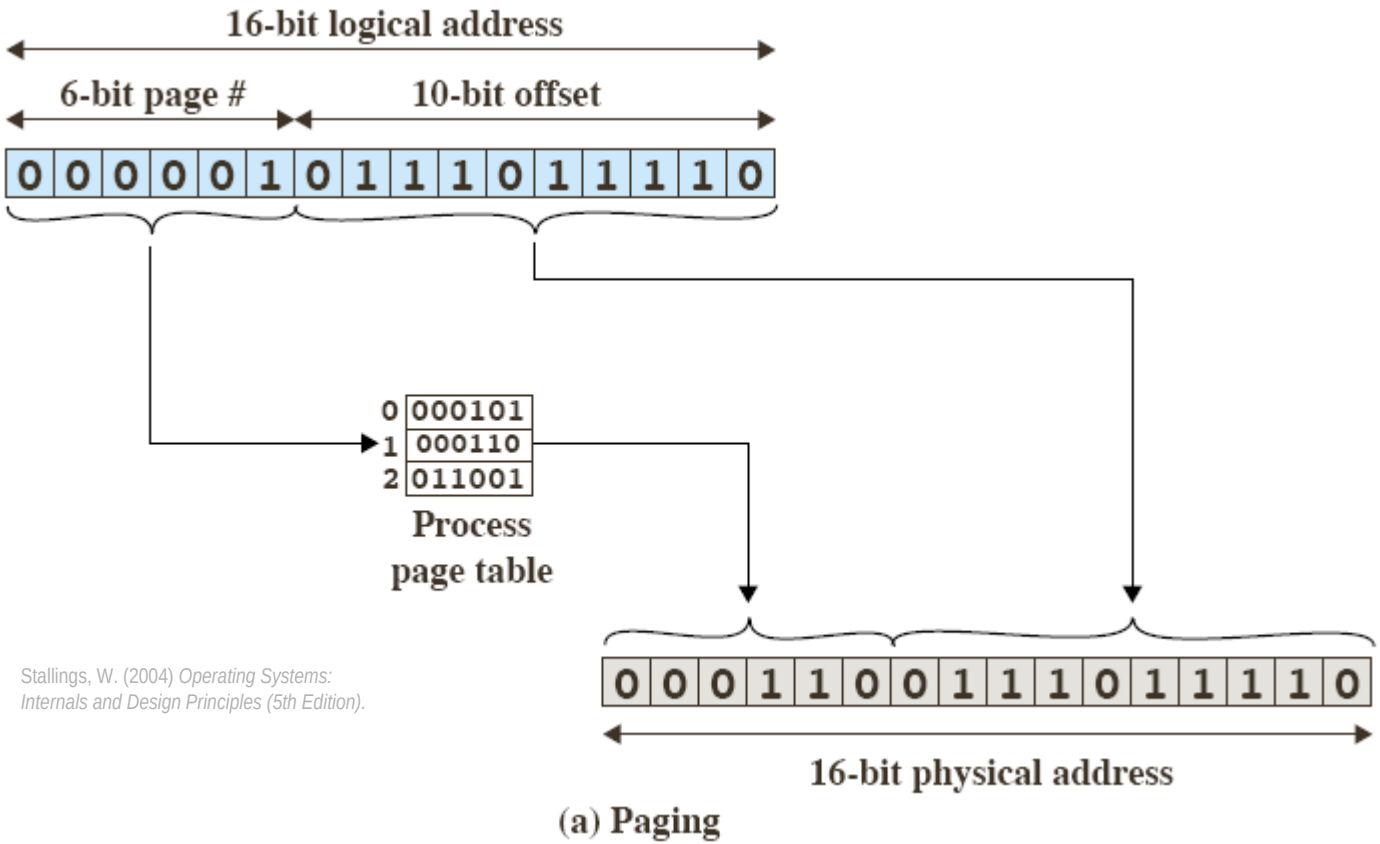
Hardware support for paging



Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

3.d Simple Paging & Segmentation

Hardware support for paging

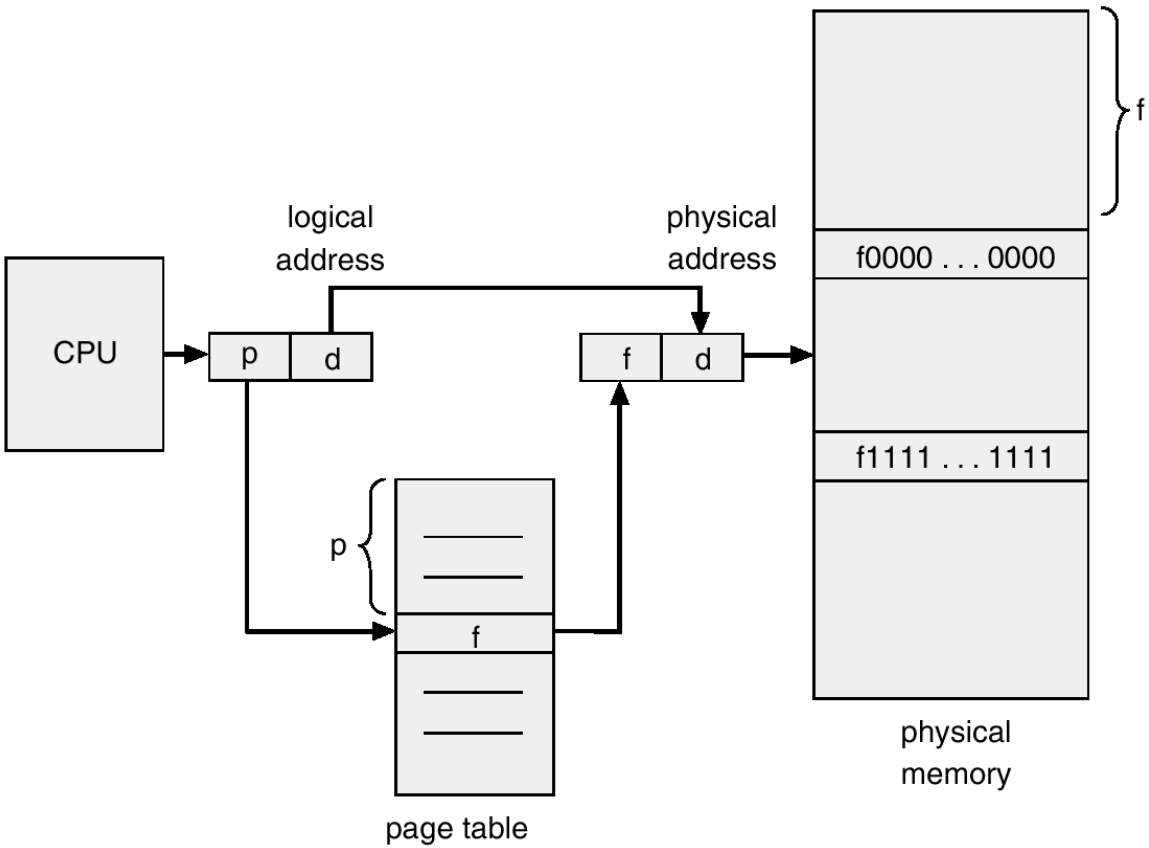


(a) Paging

Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

3.d Simple Paging & Segmentation

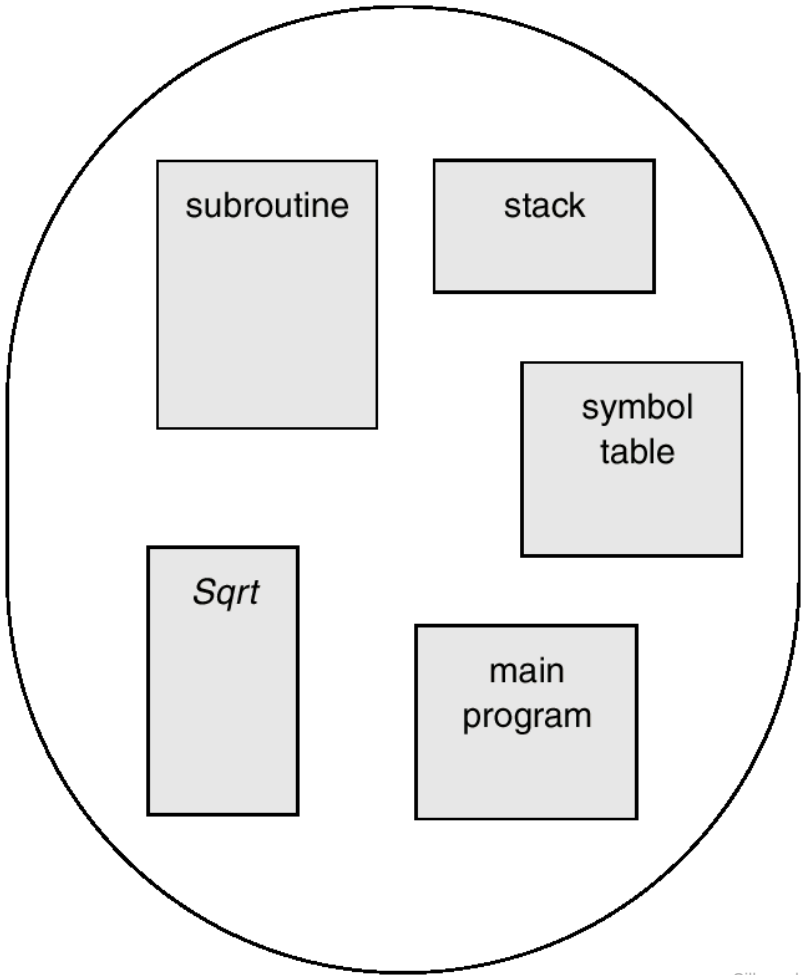
Hardware support for paging



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

3.d Simple Paging & Segmentation

Segmentation



logical address space

Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

3.d Simple Paging & Segmentation

Segmentation

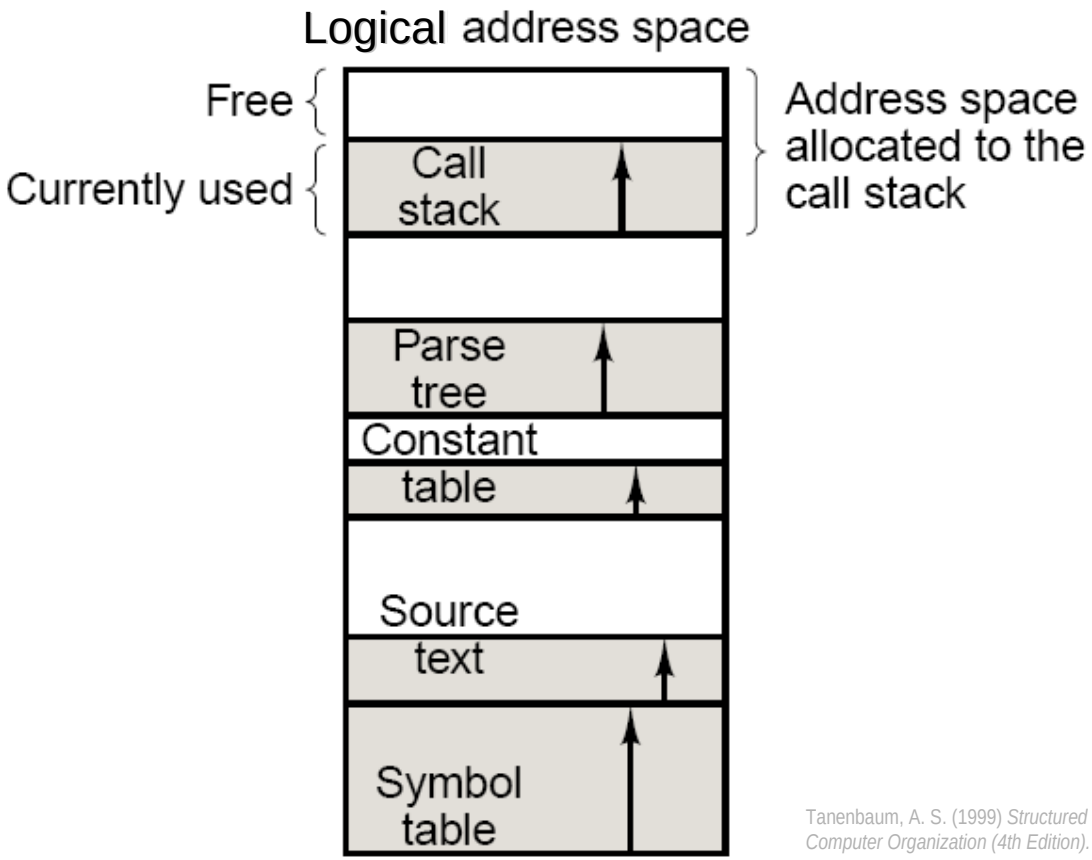


Figure 6-7. In a one-dimensional address space with growing tables, one table may bump into another.

3.d Simple Paging & Segmentation

Segmentation

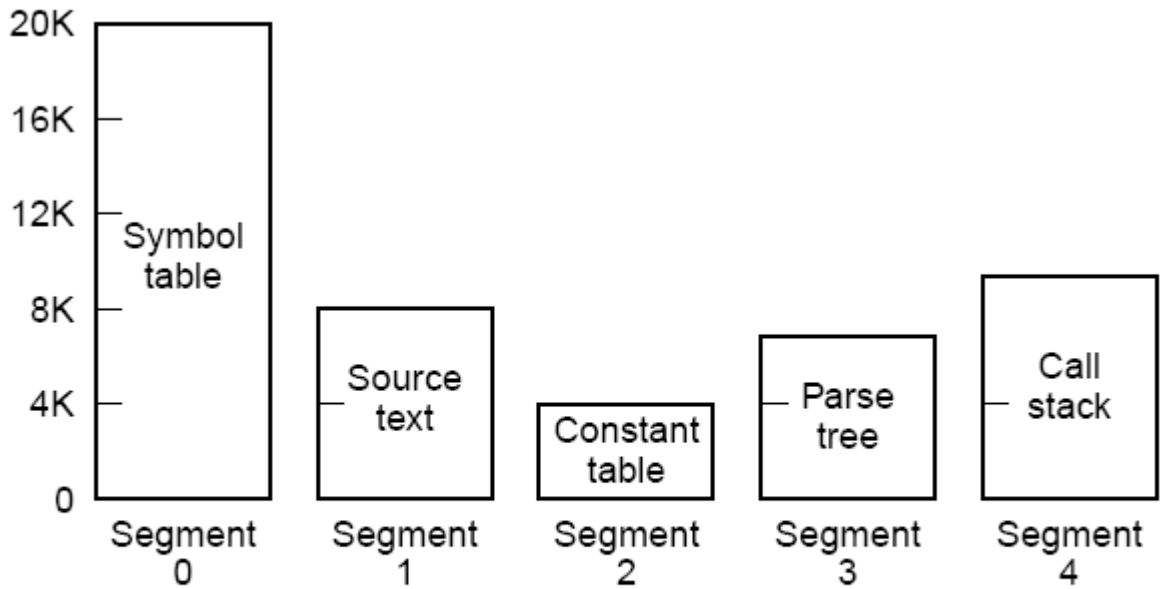
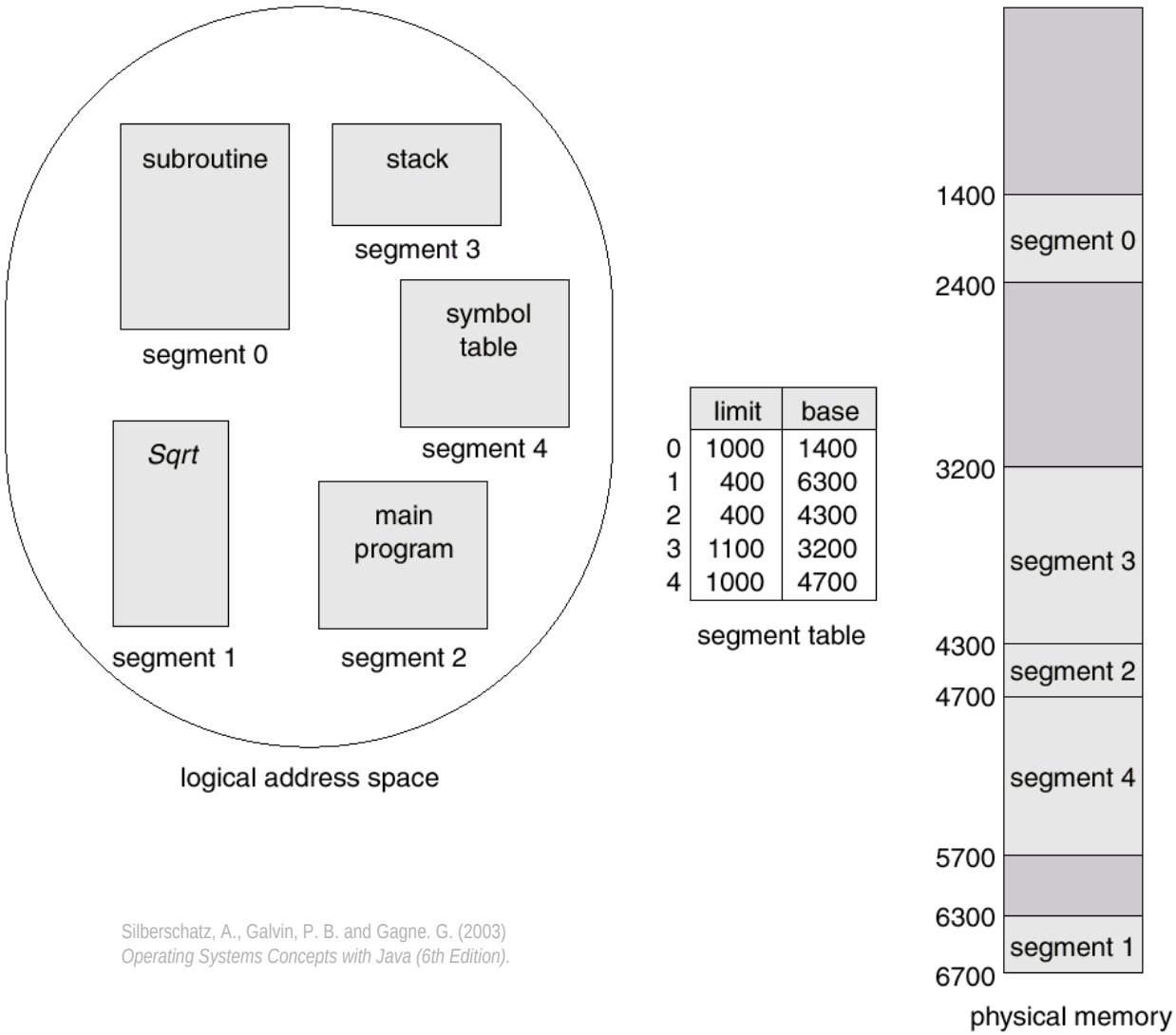


Figure 6-8. A segmented memory allows each table to grow or shrink independently of the other tables.

Tanenbaum, A. S. (1999) *Structured Computer Organization (4th Edition).*

3.d Simple Paging & Segmentation

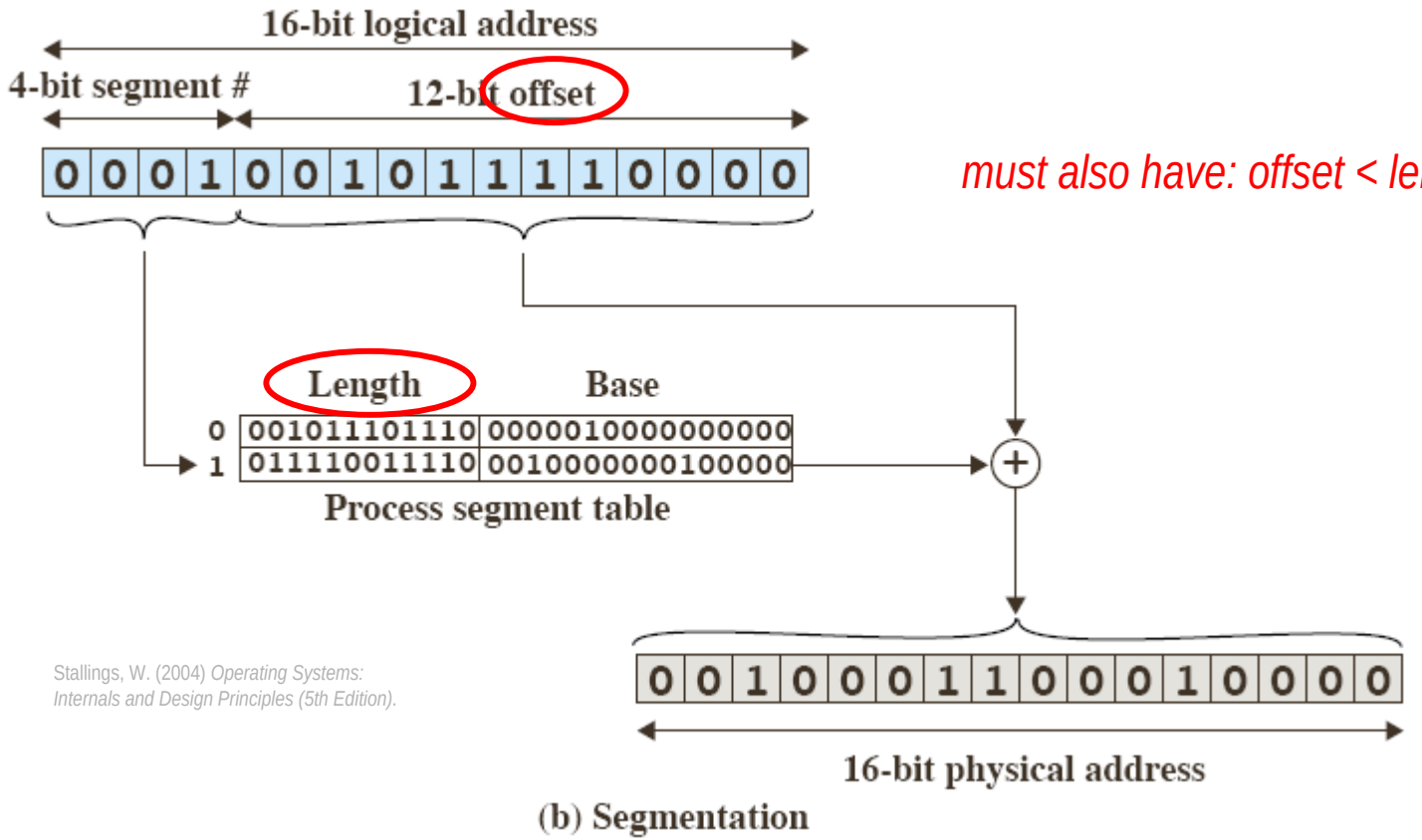
Segmentation



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

3.d Simple Paging & Segmentation

Segmentation



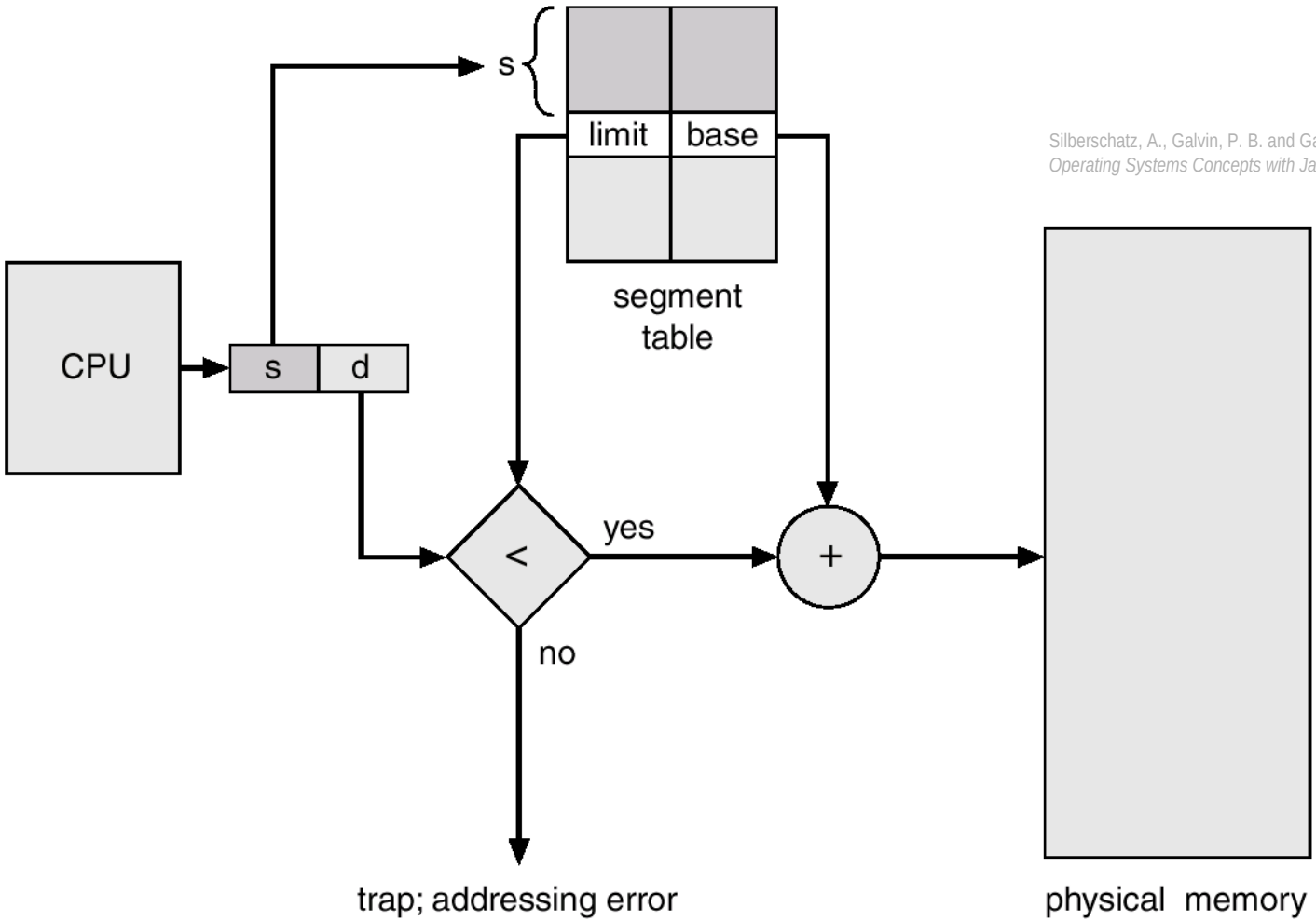
(b) Segmentation

must also have: offset < length

Stallings, W. (2004) *Operating Systems: Internals and Design Principles (5th Edition).*

3.d Simple Paging & Segmentation

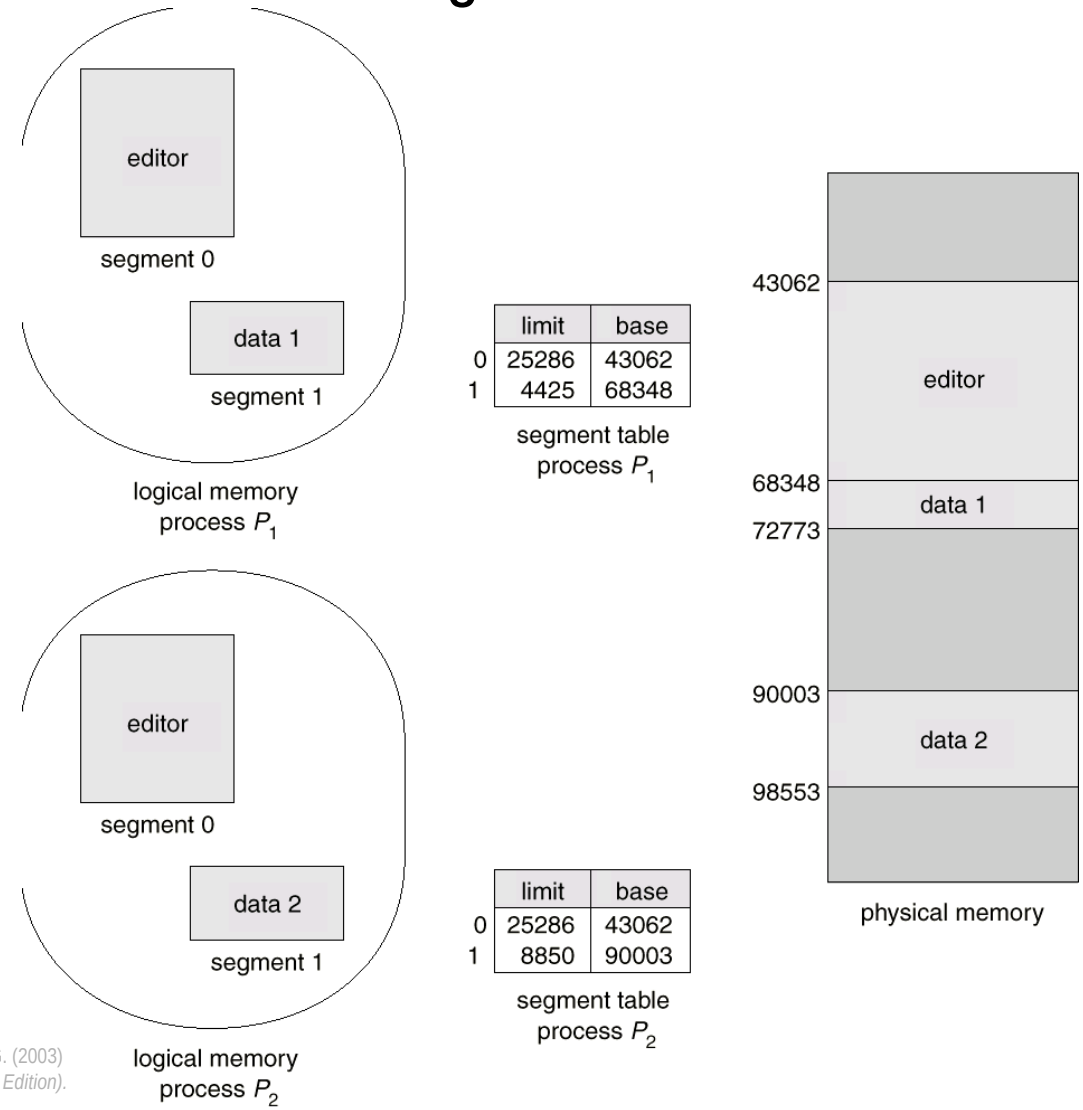
Segmentation



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003)
Operating Systems Concepts with Java (6th Edition).

3.d Simple Paging & Segmentation

Segmentation



Silberschatz, A., Galvin, P. B. and Gagne. G. (2003) *Operating Systems Concepts with Java (6th Edition).*

Principles of Operating Systems

CS 446/646

3. Memory Management

a. Goals of Memory Management

b. Partitioning

c. Linking & Loading

d. Simple Paging & Segmentation

- ✓ Paging
- ✓ Hardware support for paging
- ✓ Segmentation

e. Virtual Memory

f. Page Replacement Algorithms