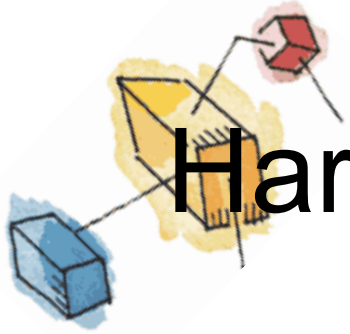


*Operating Systems:
Internals and Design Principles, 6/E*
William Stallings



Chapter 8
Virtual Memory

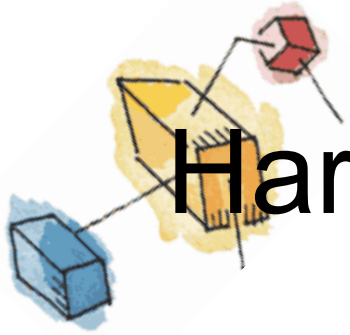
Patricia Roy
Manatee Community College, Venice, FL
©2008, Prentice Hall



Hardware and Control Structures

- Memory references are dynamically translated into physical addresses at run time
 - A process may be swapped in and out of main memory such that it occupies different regions

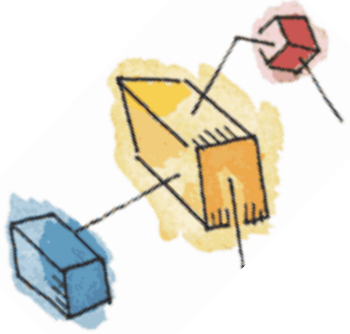




Hardware and Control Structures

- A process may be broken up into pieces that do not need to be located contiguously in main memory
- All pieces of a process do not need to be loaded in main memory during execution

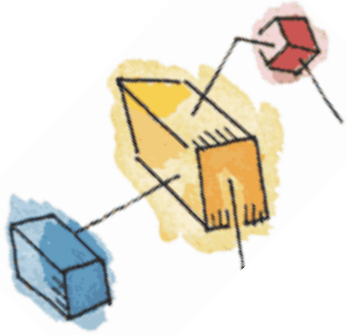




Execution of a Program

- Operating system brings into main memory a few pieces of the program
- Resident set - portion of process that is in main memory
- An interrupt is generated when an address is needed that is not in main memory
- Operating system places the process in a blocking state

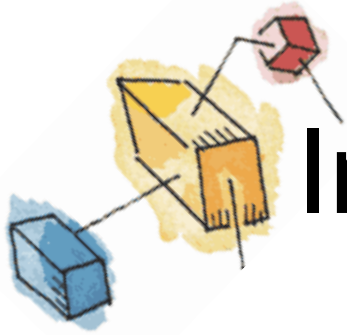




Execution of a Program

- Piece of process that contains the logical address is brought into main memory
 - Operating system issues a disk I/O Read request
 - Another process is dispatched to run while the disk I/O takes place
 - An interrupt is issued when disk I/O complete which causes the operating system to place the affected process in the Ready state

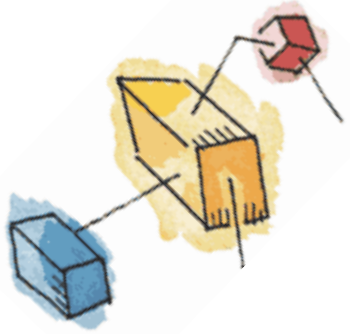




Improved System Utilization

- More processes may be maintained in main memory
 - Only load in some of the pieces of each process
 - With so many processes in main memory, it is very likely a process will be in the Ready state at any particular time
- A process may be larger than all of main memory

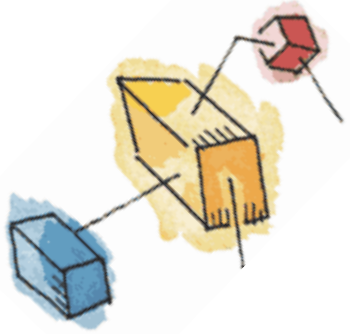




Types of Memory

- Real memory
 - Main memory
- Virtual memory
 - Memory on disk
 - Allows for effective multiprogramming and relieves the user of tight constraints of main memory

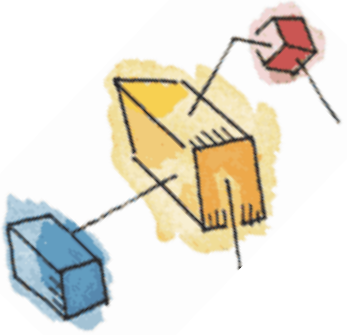




Thrashing

- Swapping out a piece of a process just before that piece is needed
- The processor spends most of its time swapping pieces rather than executing user instructions





Principle of Locality

- Program and data references within a process tend to cluster
- Only a few pieces of a process will be needed over a short period of time
- Possible to make intelligent guesses about which pieces will be needed in the future
- This suggests that virtual memory may work efficiently



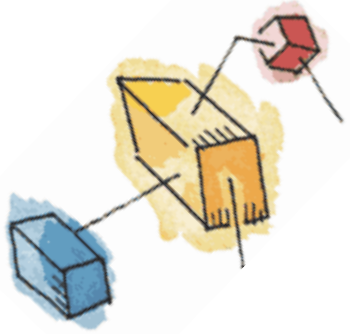


Support Needed for Virtual Memory

- Hardware must support paging and segmentation
- Operating system must be able to management the movement of pages and/or segments between secondary memory and main memory



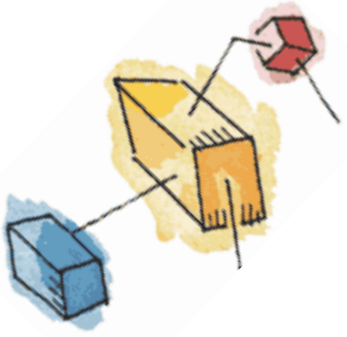
Paging



- Each process has its own page table
- Each page table entry contains the frame number of the corresponding page in main memory
- A bit is needed to indicate whether the page is in main memory or not



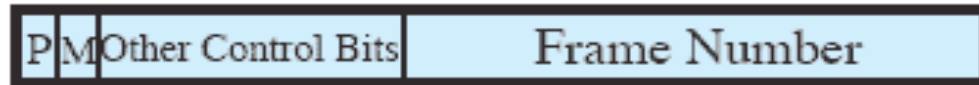
Paging



Virtual Address

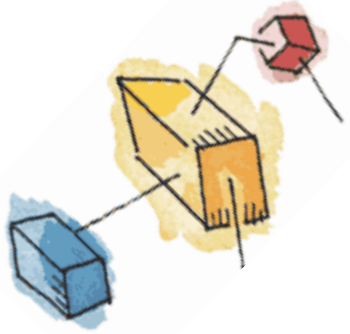


Page Table Entry



(a) Paging only





Modify Bit in Page Table

- Modify bit is needed to indicate if the page has been altered since it was last loaded into main memory
- If no change has been made, the page does not have to be written to the disk when it needs to be replaced



Address Translation

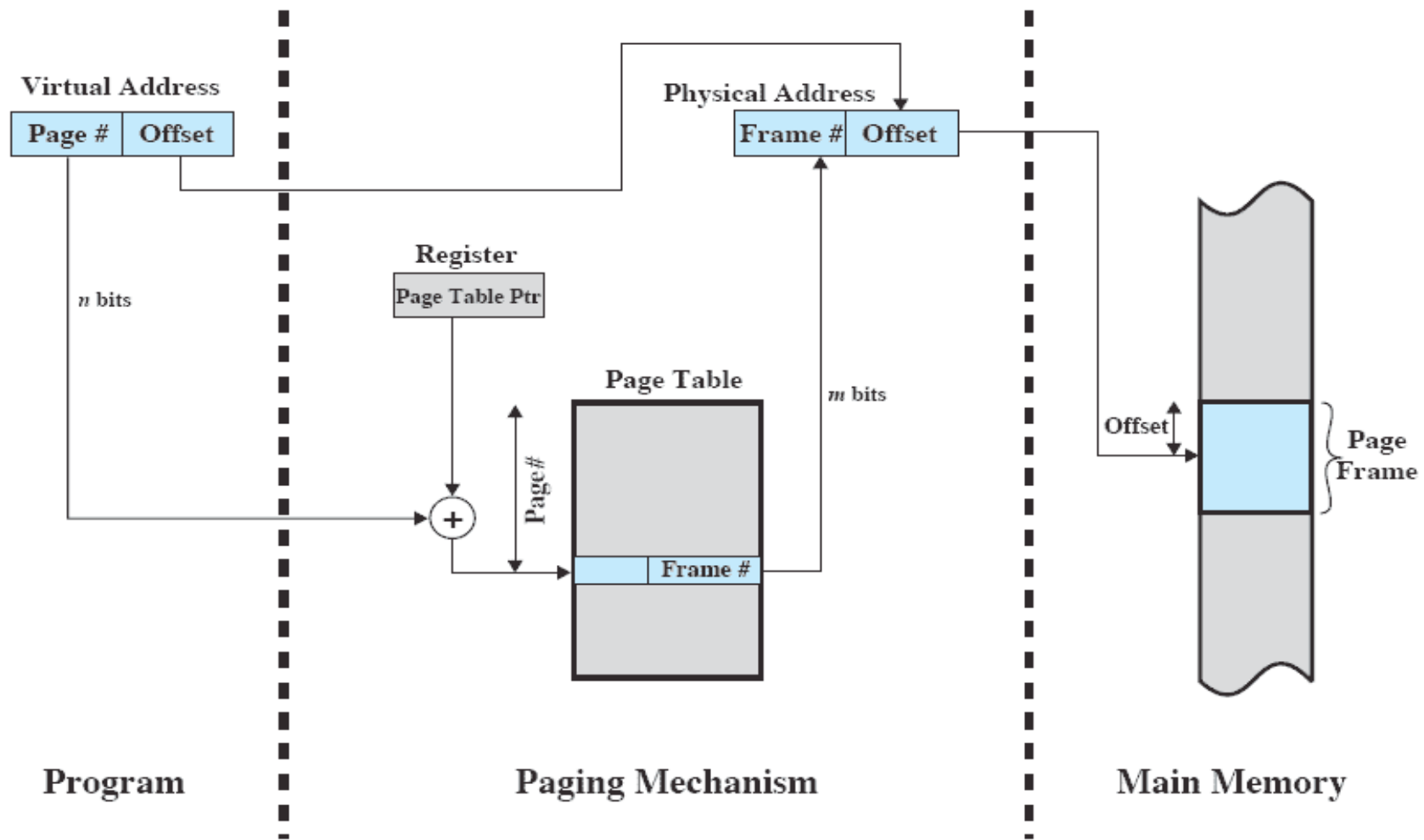
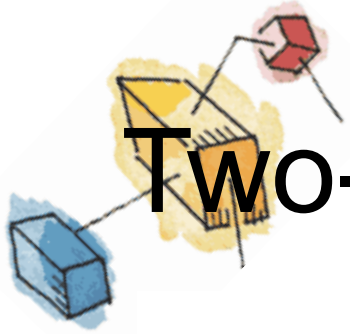


Figure 8.3 Address Translation in a Paging System



Two-Level Hierarchical Page Table

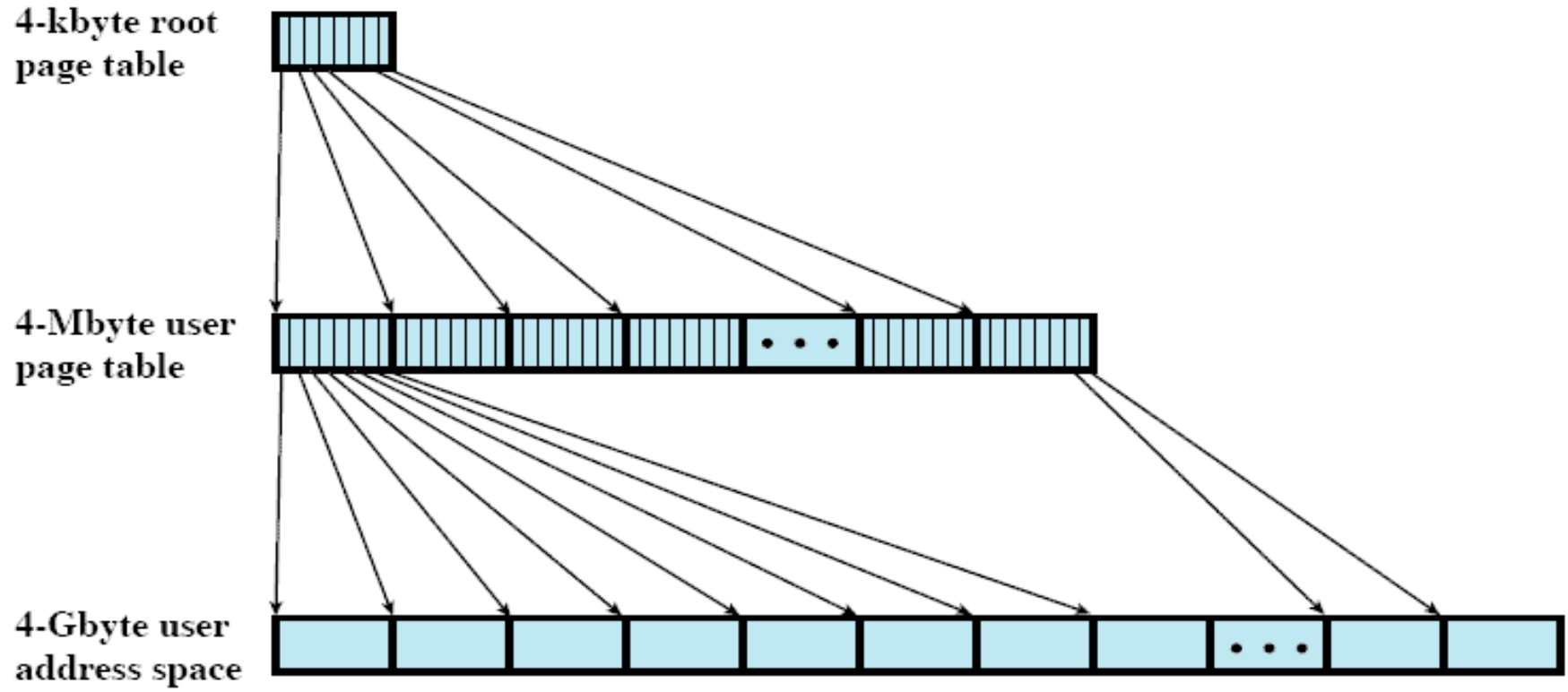
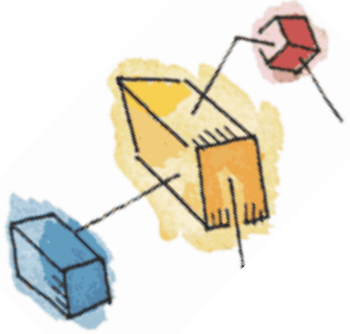


Figure 8.4 A Two-Level Hierarchical Page Table





Page Tables

- Page tables are also stored in virtual memory
- When a process is running, part of its page table is in main memory



Address Translation

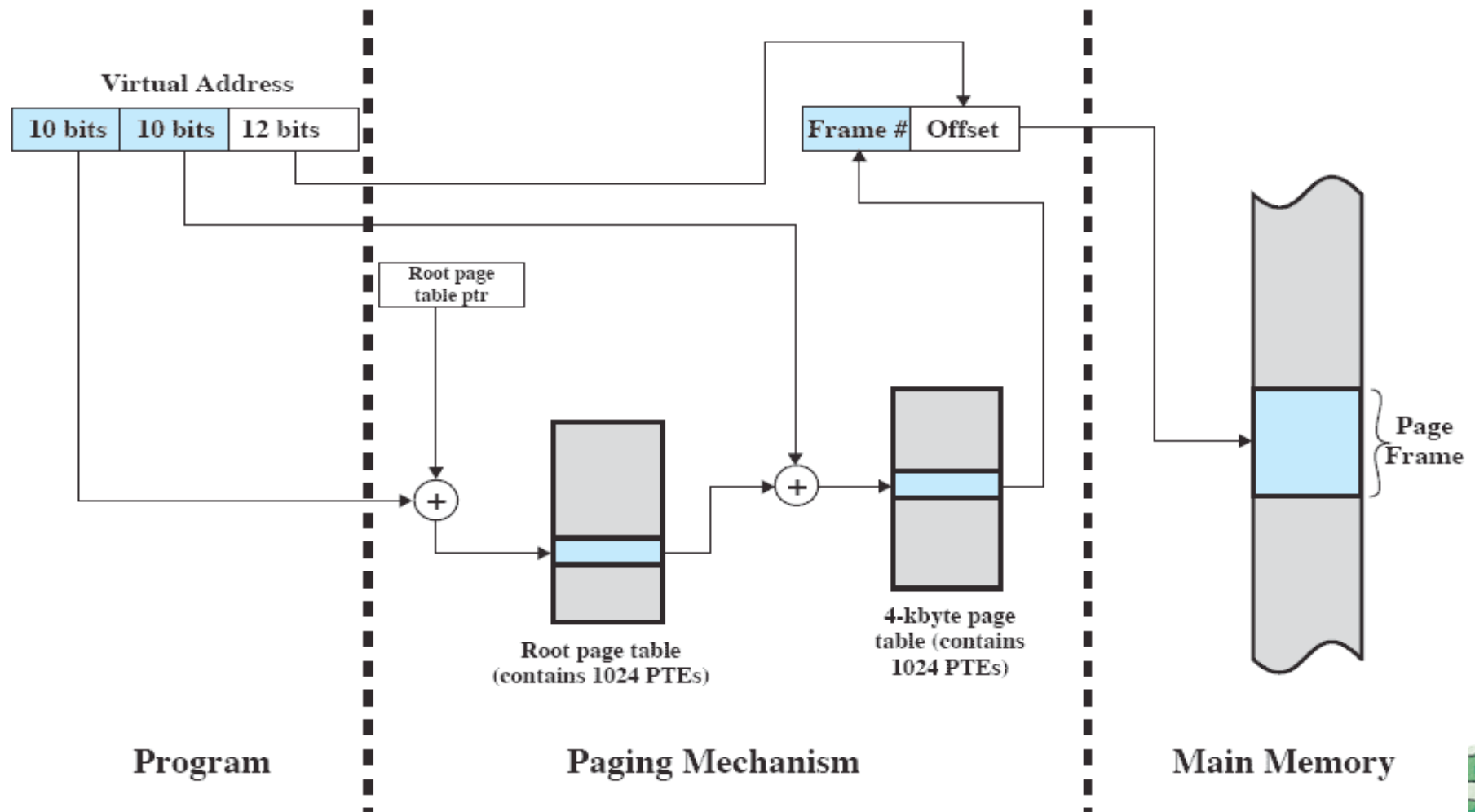
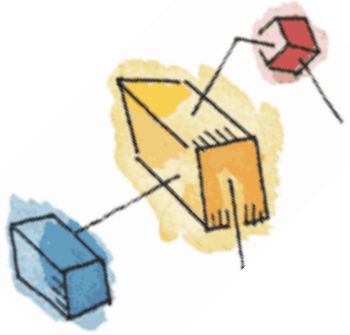


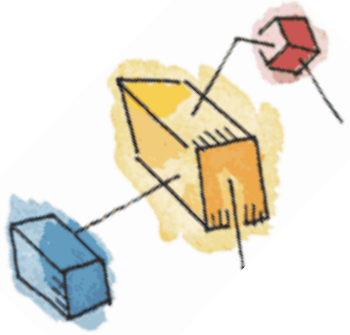
Figure 8.5 Address Translation in a Two-Level Paging System



Inverted Page Table

- Used on PowerPC, UltraSPARC, and IA-64 architecture
- Page number portion of a virtual address is mapped into a hash value
- Hash value points to inverted page table
- Fixed proportion of real memory is required for the tables regardless of the number of processes

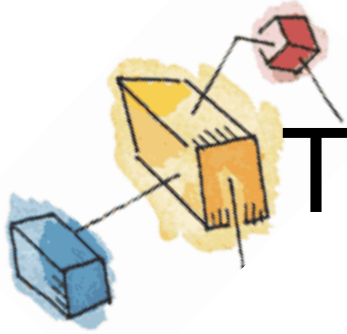




Inverted Page Table

- Page number
- Process identifier
- Control bits
- Chain pointer

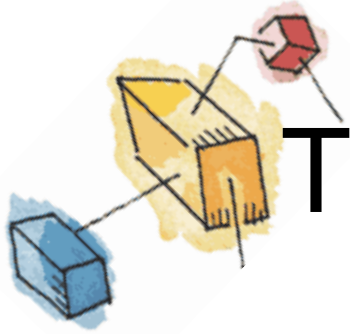




Translation Lookaside Buffer

- Each virtual memory reference can cause two physical memory accesses
 - One to fetch the page table
 - One to fetch the data
- To overcome this problem a high-speed cache is set up for page table entries
 - Called a Translation Lookaside Buffer (TLB)

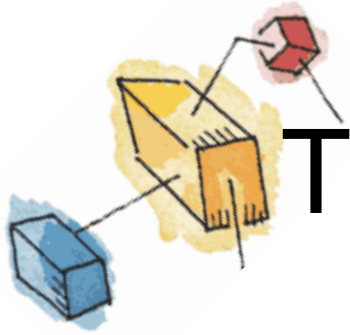




Translation Lookaside Buffer

- Contains page table entries that have been most recently used

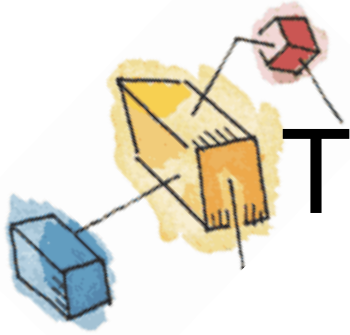




Translation Lookaside Buffer

- Given a virtual address, processor examines the TLB
- If page table entry is present (TLB hit), the frame number is retrieved and the real address is formed
- If page table entry is not found in the TLB (TLB miss), the page number is used to index the process page table

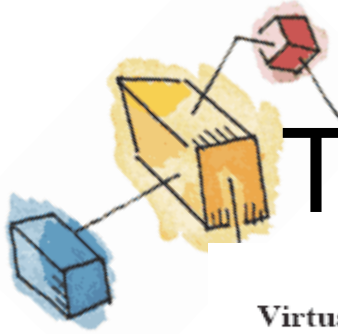




Translation Lookaside Buffer

- First checks if page is already in main memory
 - If not in main memory a page fault is issued
- The TLB is updated to include the new page entry





Translation Lookaside Buffer

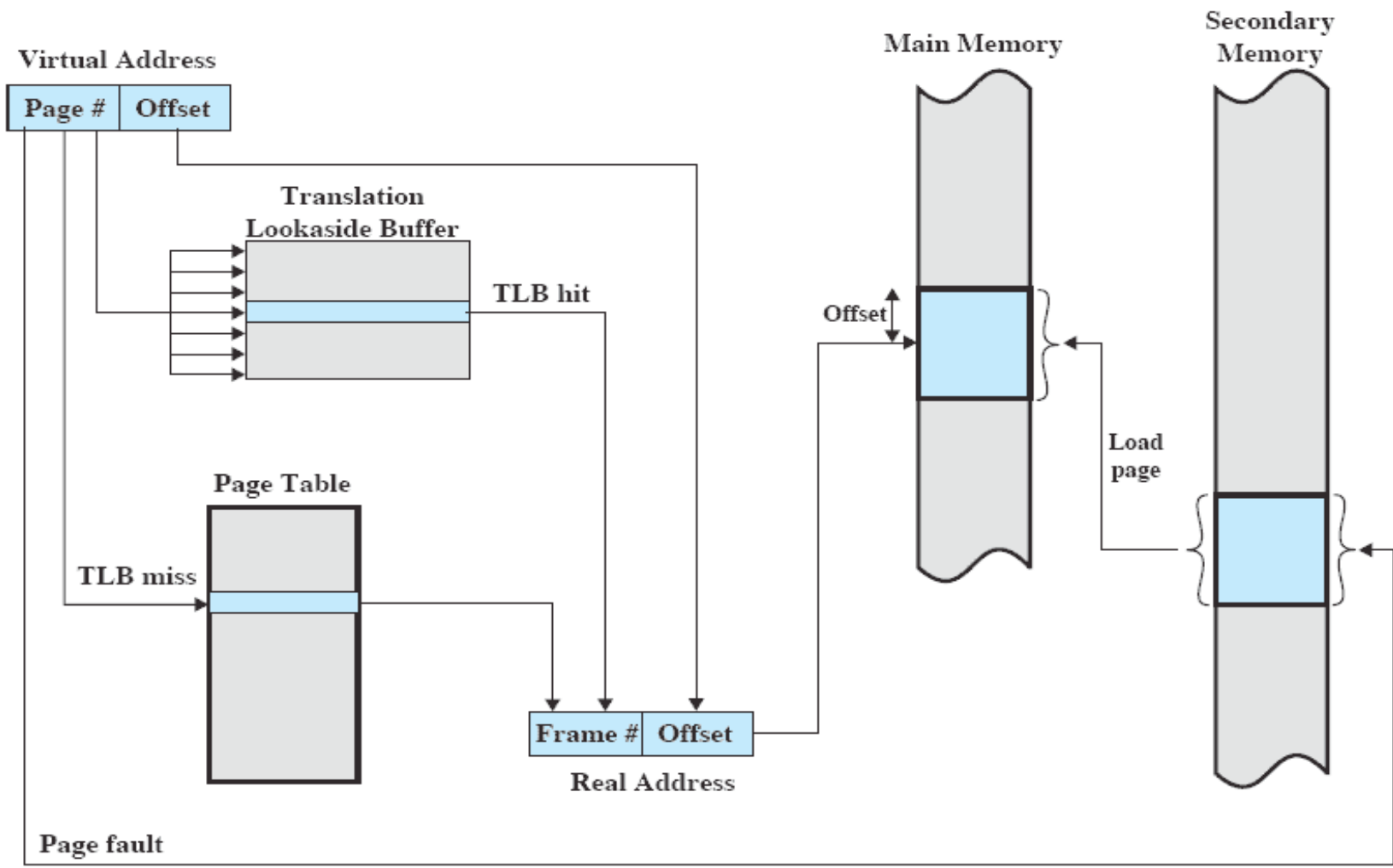
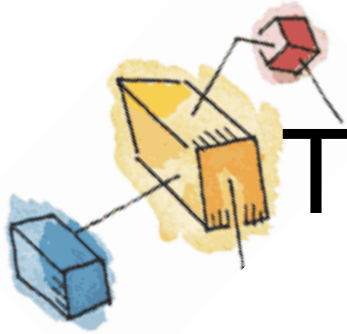


Figure 8.7 Use of a Translation Lookaside Buffer





Translation Lookaside Buffer

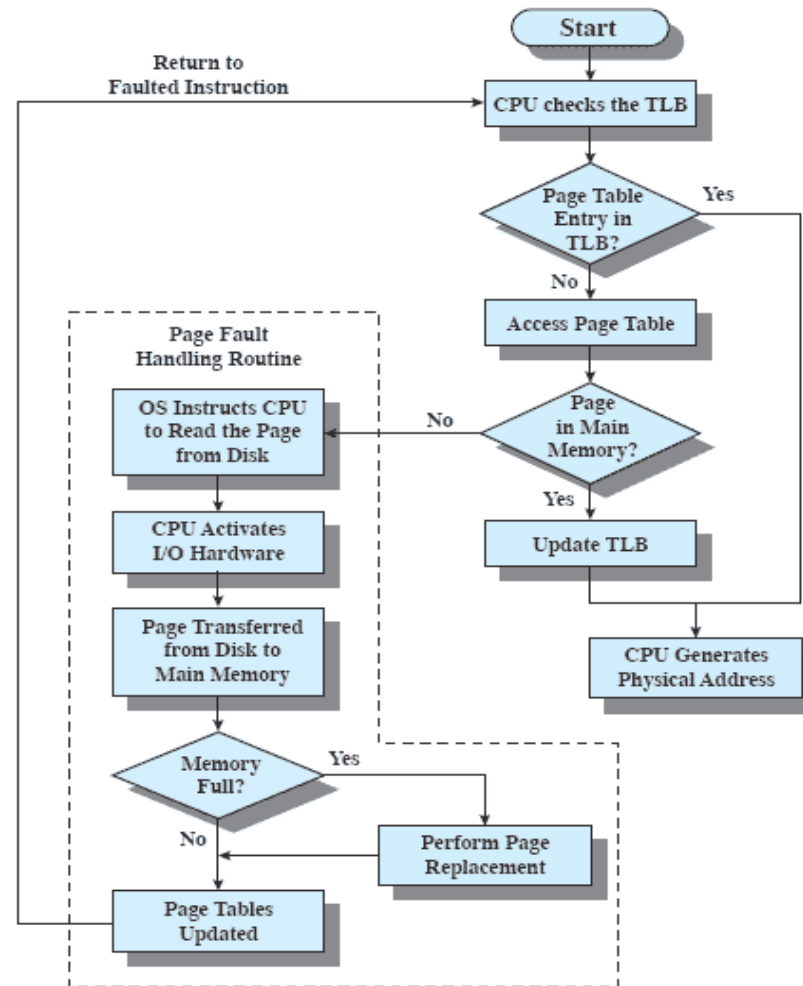
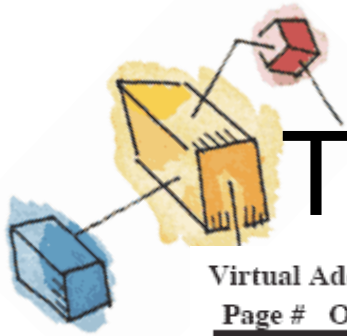


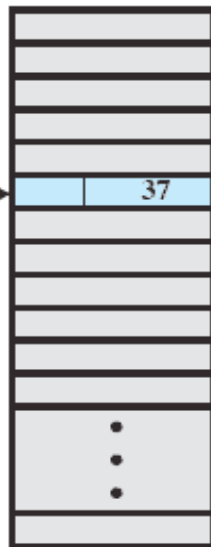
Figure 8.8 Operation of Paging and Translation Lookaside Buffer (TLB) [FURH87]





Translation Lookaside Buffer

Virtual Address
Page # Offset
5 502

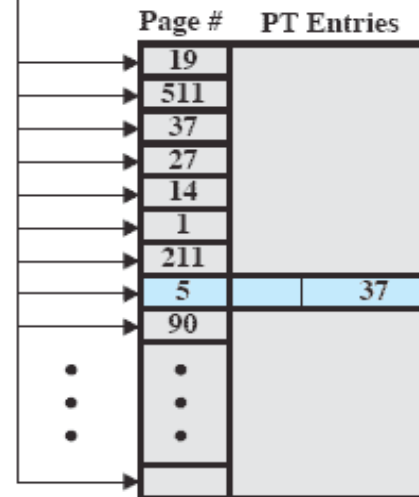


Page Table

37 502
Frame # Offset
Real Address

(a) Direct mapping

Virtual Address
Page # Offset
5 502



Translation Lookaside Buffer

37 502
Frame # Offset
Real Address

(b) Associative mapping



Figure 8.9 Direct Versus Associative Lookup for Page Table Entries





Translation Lookaside Buffer

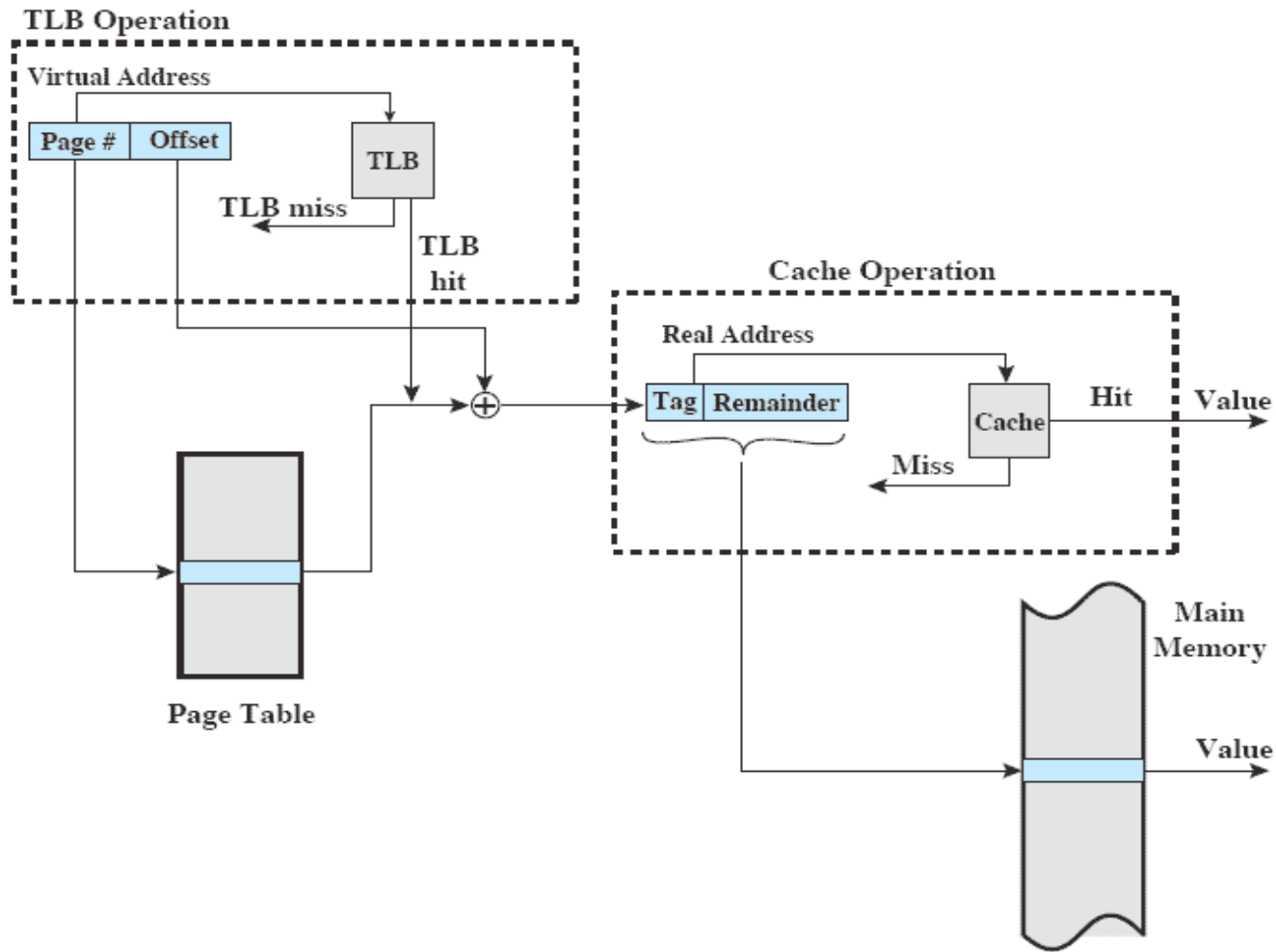
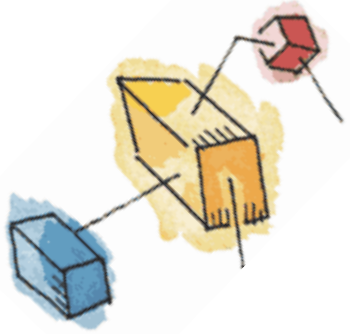


Figure 8.10 Translation Lookaside Buffer and Cache Operation



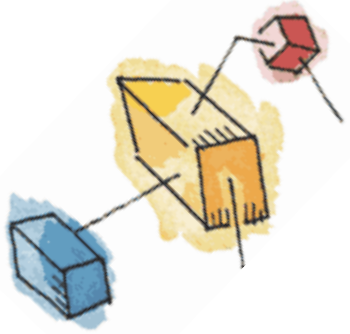


Page Size

- Smaller page size, less amount of internal fragmentation
- Smaller page size, more pages required per process
- More pages per process means larger page tables
- Larger page tables means large portion of page tables in virtual memory

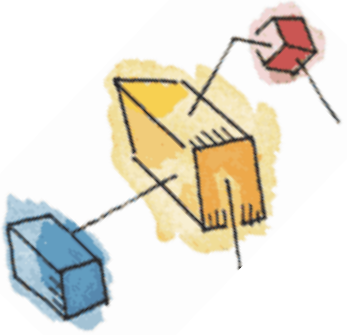


Page Size



- Secondary memory is designed to efficiently transfer large blocks of data so a large page size is better



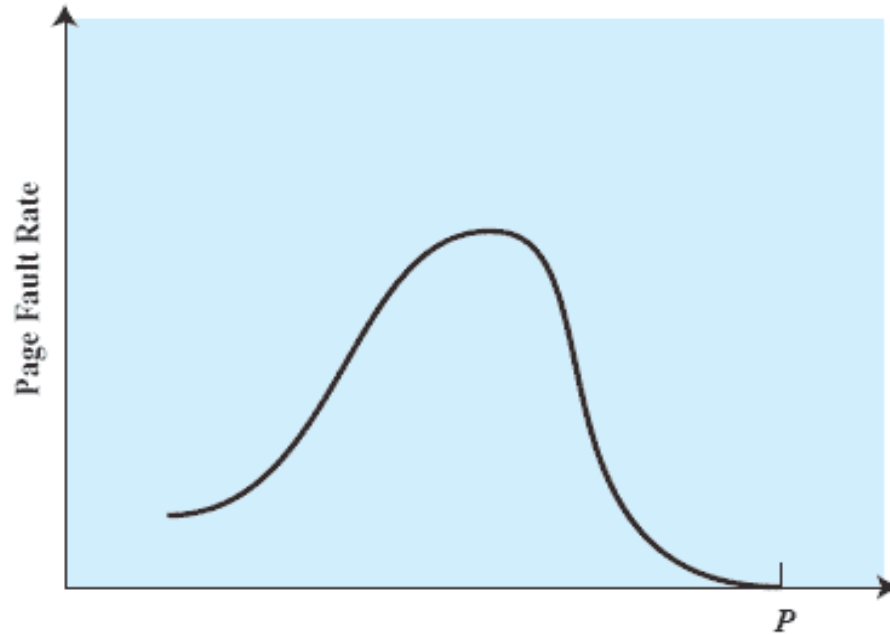
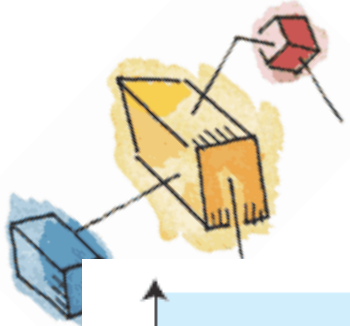


Page Size

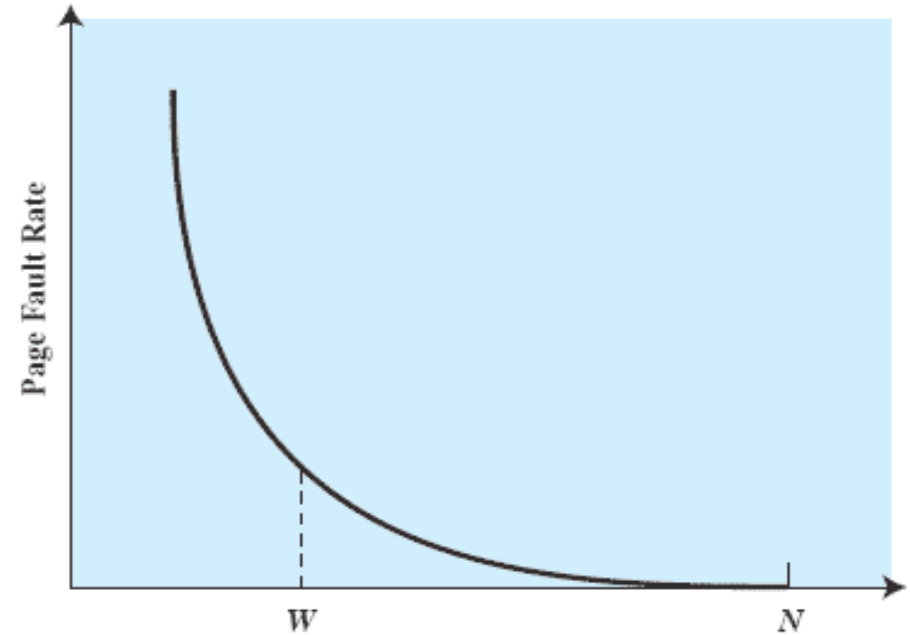
- Small page size, large number of pages will be found in main memory
- As time goes on during execution, the pages in memory will all contain portions of the process near recent references. Page faults low.
- Increased page size causes pages to contain locations further from any recent reference. Page faults rise.



Page Size



(a) Page Size



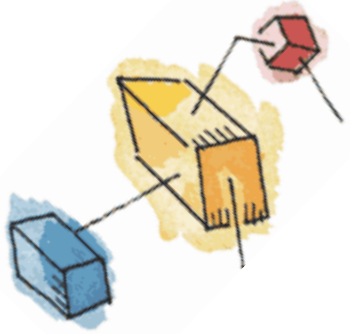
(b) Number of Page Frames Allocated

P = size of entire process

W = working set size

N = total number of pages in process

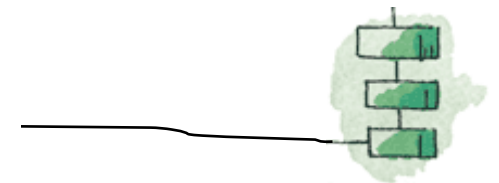
Figure 8.11 Typical Paging Behavior of a Program

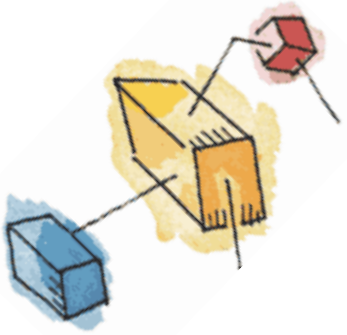


Example Page Size

Table 8.3 Example Page Sizes

Computer	Page Size
Atlas	512 48-bit words
Honeywell-Multics	1024 36-bit word
IBM 370/XA and 370/ESA	4 Kbytes
VAX family	512 bytes
IBM AS/400	512 bytes
DEC Alpha	8 Kbytes
MIPS	4 Kbytes to 16 Mbytes
UltraSPARC	8 Kbytes to 4 Mbytes
Pentium	4 Kbytes or 4 Mbytes
IBM POWER	4 Kbytes
Itanium	4 Kbytes to 256 Mbytes

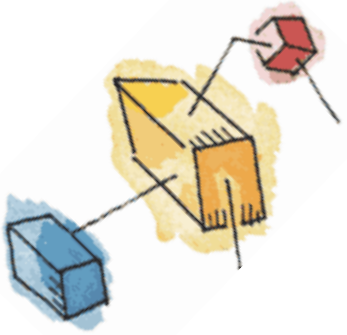




Segmentation

- May be unequal, dynamic size
- Simplifies handling of growing data structures
- Allows programs to be altered and recompiled independently
- Lends itself to sharing data among processes
- Lends itself to protection

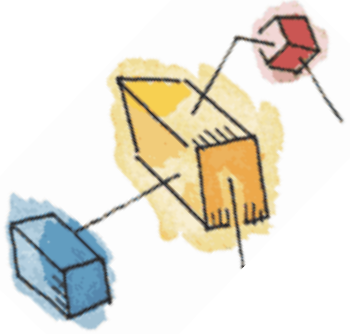




Segment Tables

- Starting address corresponding segment in main memory
- Each entry contains the length of the segment
- A bit is needed to determine if segment is already in main memory
- Another bit is needed to determine if the segment has been modified since it was loaded in main memory





Segment Table Entries

Virtual Address



Segment Table Entry



(b) Segmentation only



Segmentation

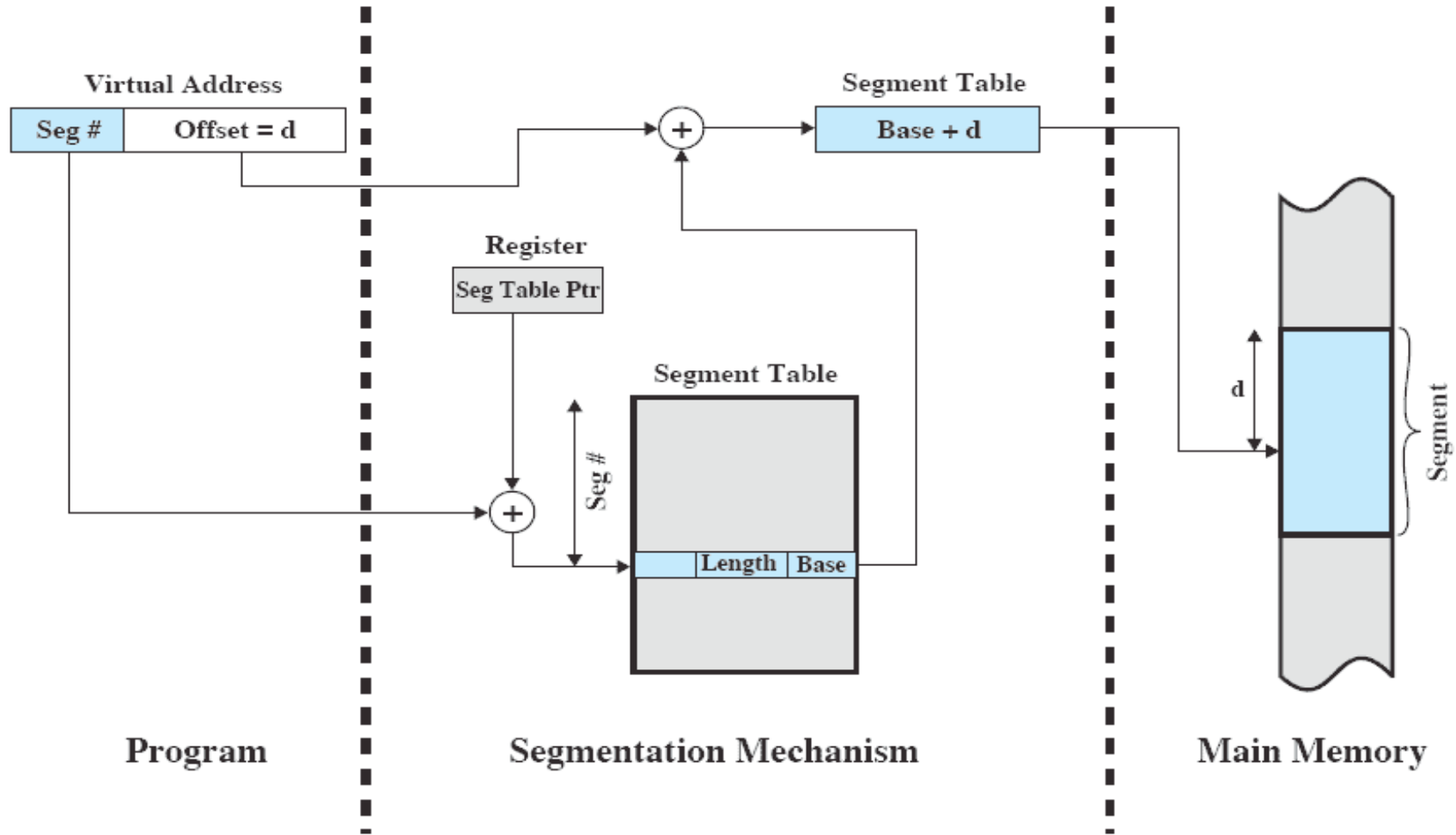
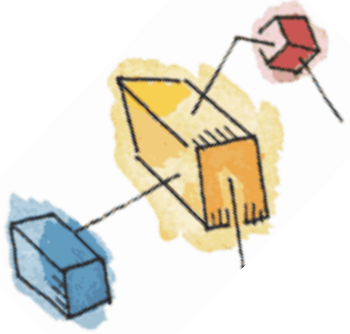


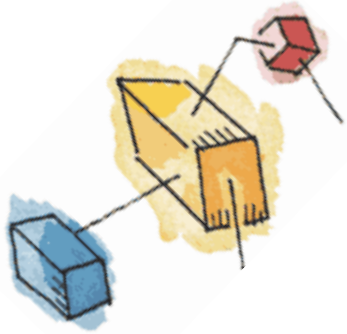
Figure 8.12 Address Translation in a Segmentation System



Combined Paging and Segmentation

- Paging is transparent to the programmer
- Segmentation is visible to the programmer
- Each segment is broken into fixed-size pages





Combined Paging and Segmentation

Virtual Address



Segment Table Entry



Page Table Entry



P= present bit

M = Modified bit

(c) Combined segmentation and paging



Address Translation

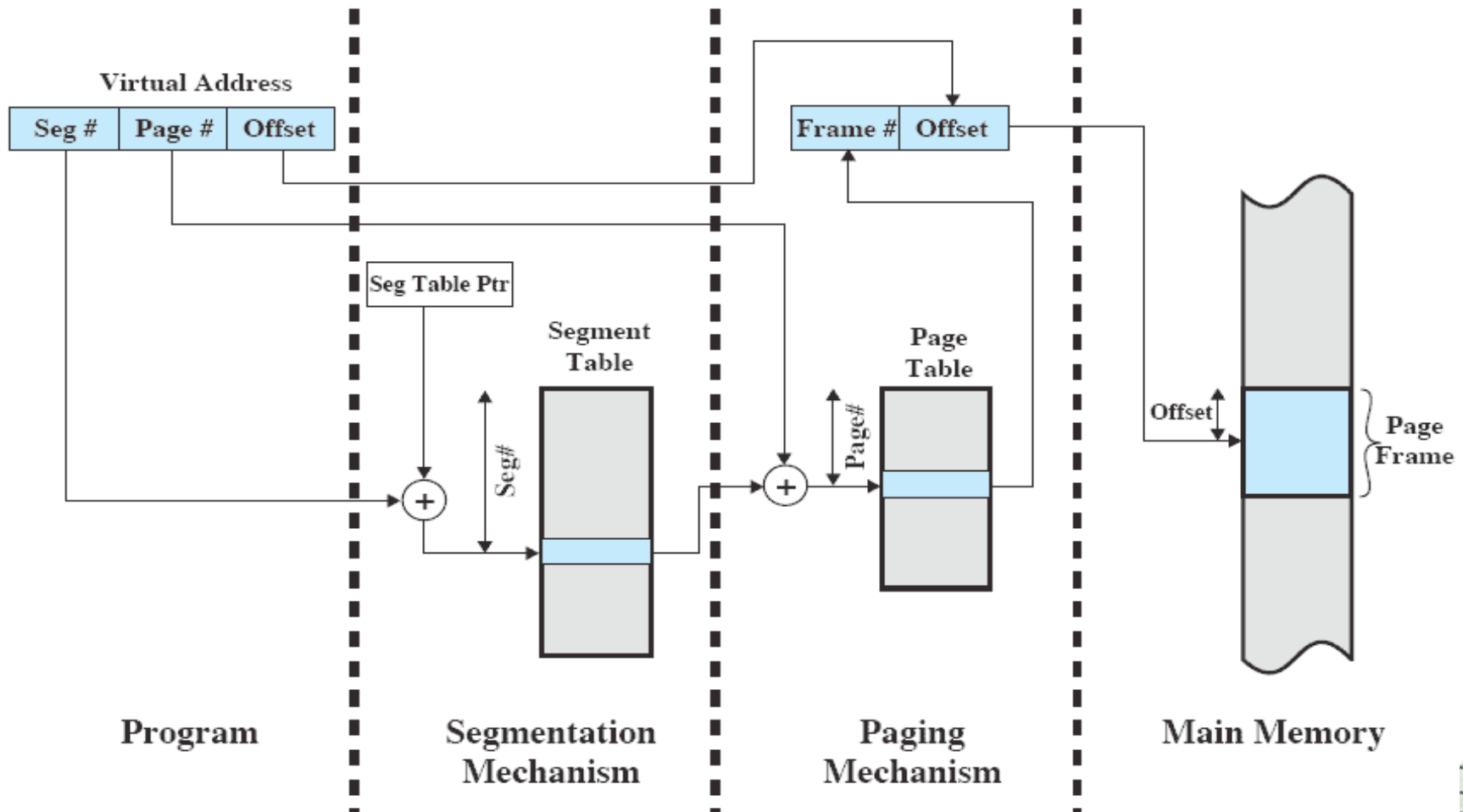
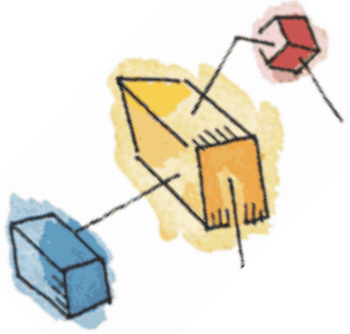


Figure 8.13 Address Translation in a Segmentation/Paging System



Protection Relationships

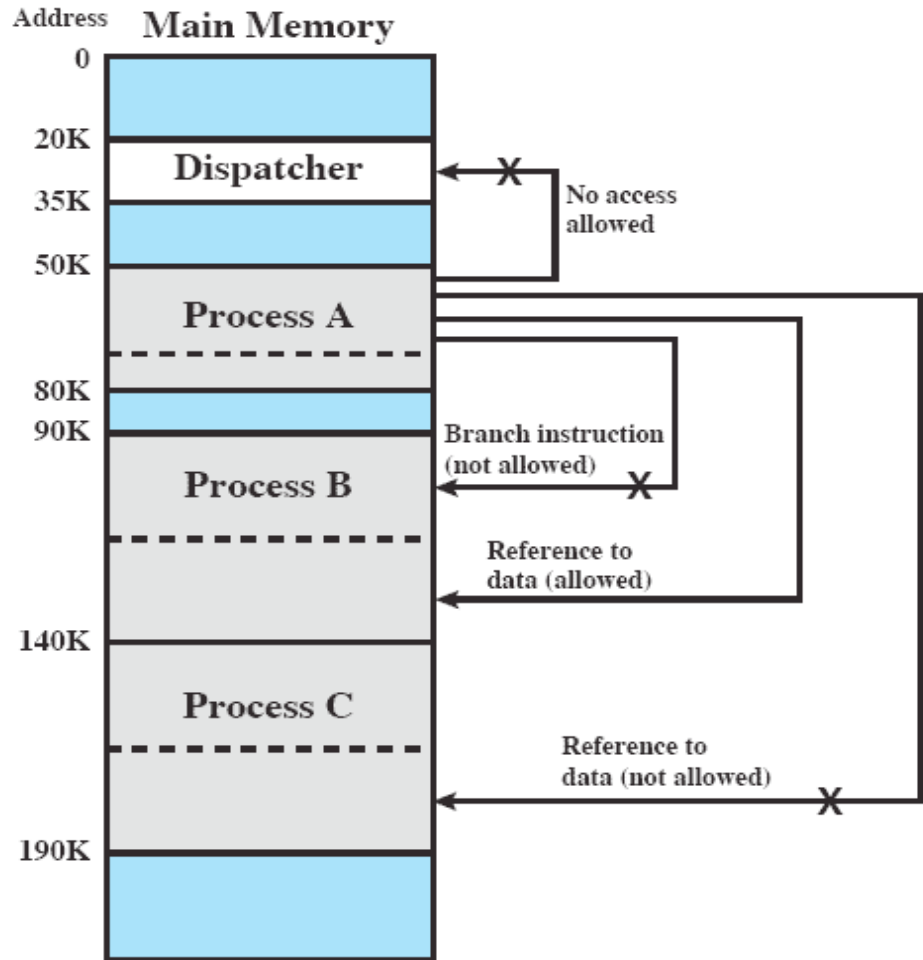
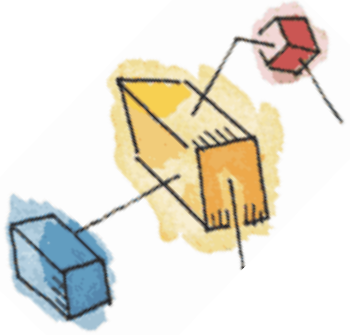


Figure 8.14 Protection Relationships Between Segments

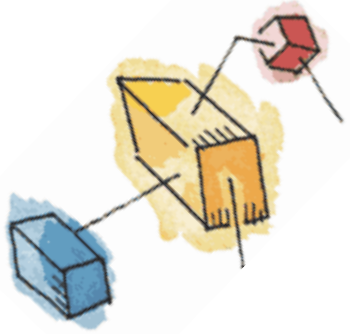




Fetch Policy

- Determines when a page should be brought into memory
- Demand paging only brings pages into main memory when a reference is made to a location on the page
 - Many page faults when process first started
- Prepaging brings in more pages than needed
 - More efficient to bring in pages that reside contiguously on the disk

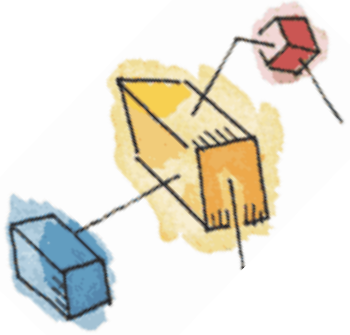




Placement Policy

- Determines where in real memory a process piece is to reside
- Important in a segmentation system
- Paging or combined paging with segmentation hardware performs address translation

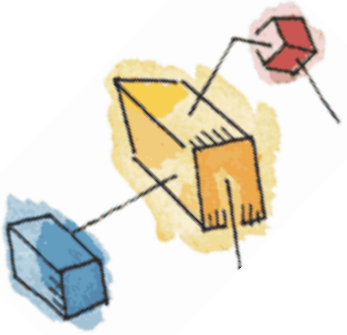




Replacement Policy

- Which page is replaced?
- Page removed should be the page least likely to be referenced in the near future
- Most policies predict the future behavior on the basis of past behavior

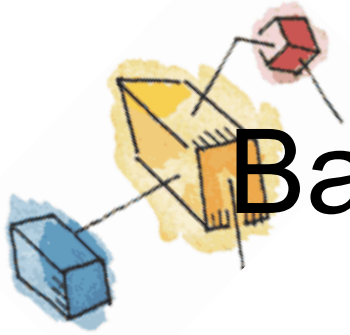




Replacement Policy

- Frame Locking
 - If frame is locked, it may not be replaced
 - Kernel of the operating system
 - Key control structures
 - I/O buffers
 - Associate a lock bit with each frame

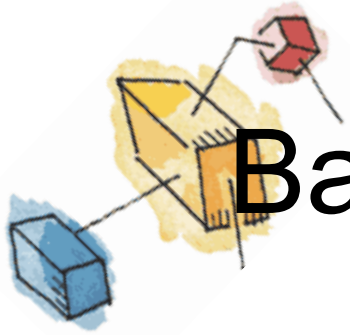




Basic Replacement Algorithms

- Optimal policy
 - Selects for replacement that page for which the time to the next reference is the longest
 - Impossible to have perfect knowledge of future events

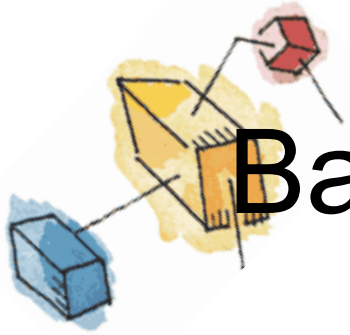




Basic Replacement Algorithms

- Least Recently Used (LRU)
 - Replaces the page that has not been referenced for the longest time
 - By the principle of locality, this should be the page least likely to be referenced in the near future
 - Each page could be tagged with the time of last reference. This would require a great deal of overhead.

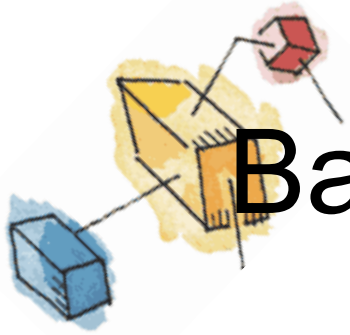




Basic Replacement Algorithms

- First-in, first-out (FIFO)
 - Treats page frames allocated to a process as a circular buffer
 - Pages are removed in round-robin style
 - Simplest replacement policy to implement
 - Page that has been in memory the longest is replaced
 - These pages may be needed again very soon



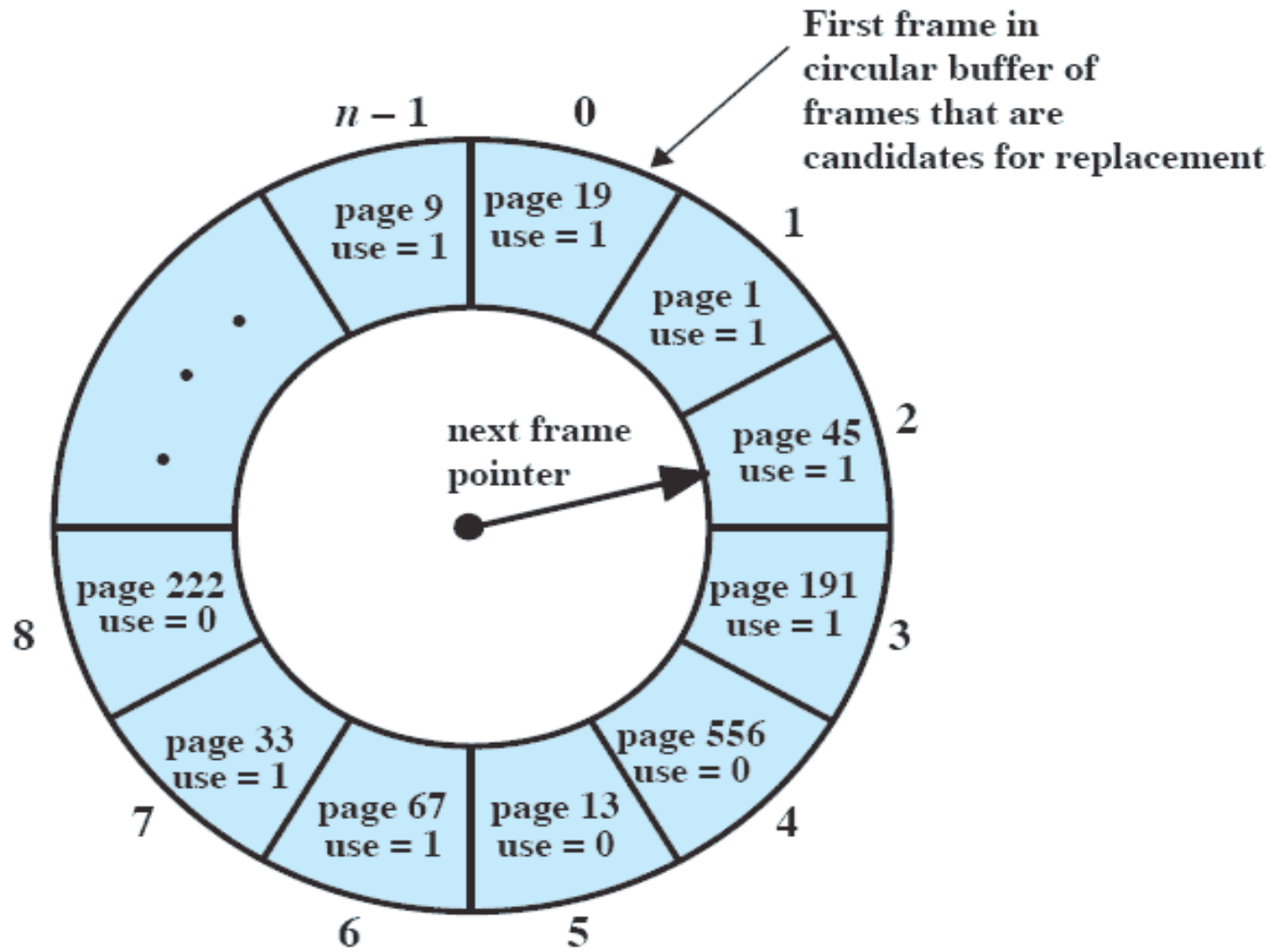
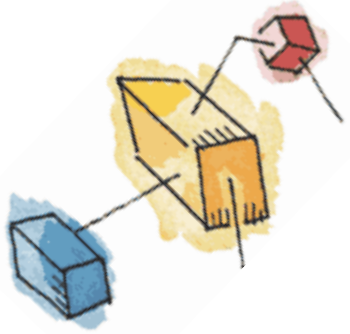


Basic Replacement Algorithms

- Clock Policy
 - Additional bit called a use bit
 - When a page is first loaded in memory, the use bit is set to 1
 - When the page is referenced, the use bit is set to 1
 - When it is time to replace a page, the first frame encountered with the use bit set to 0 is replaced.
 - During the search for replacement, each use bit set to 1 is changed to 0

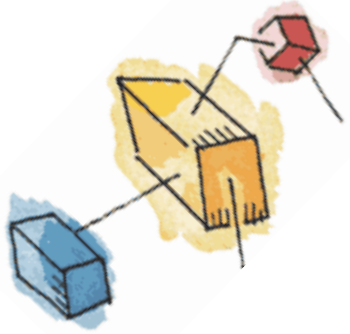


Clock Policy

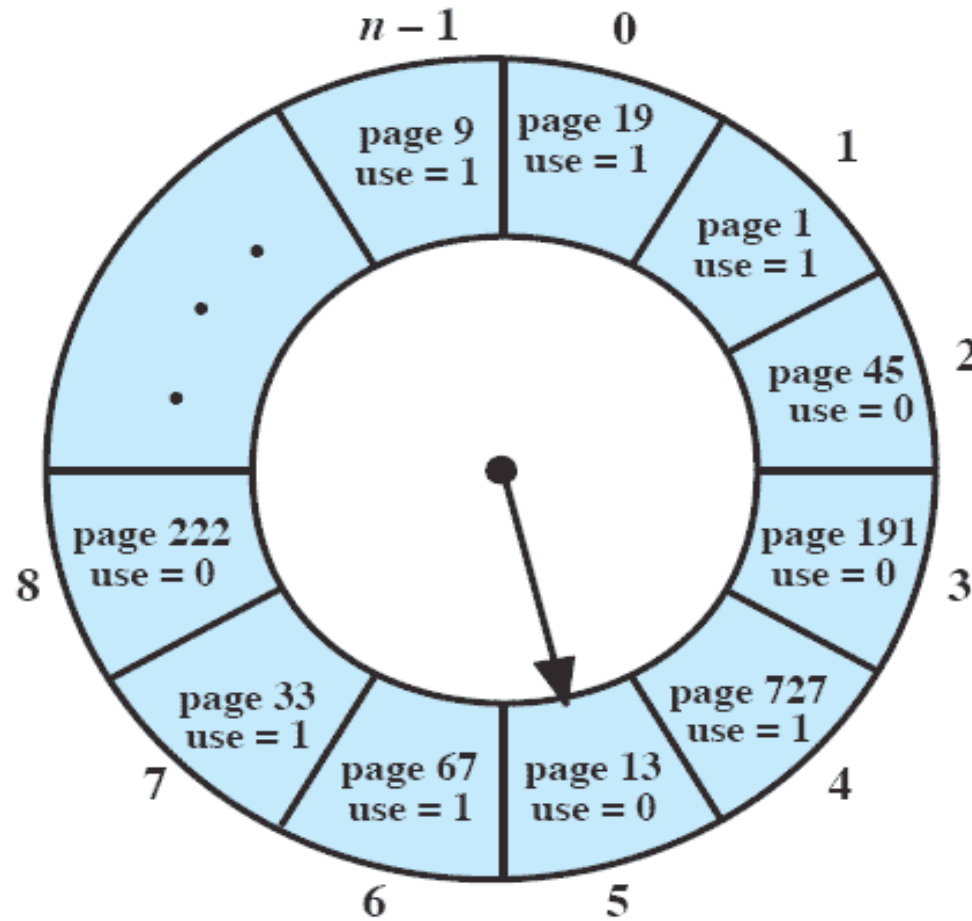


(a) State of buffer just prior to a page replacement





Clock Policy



(b) State of buffer just after the next page replacement



Figure 8.16 Example of Clock Policy Operation



Clock Policy

First frame in circular buffer for this process

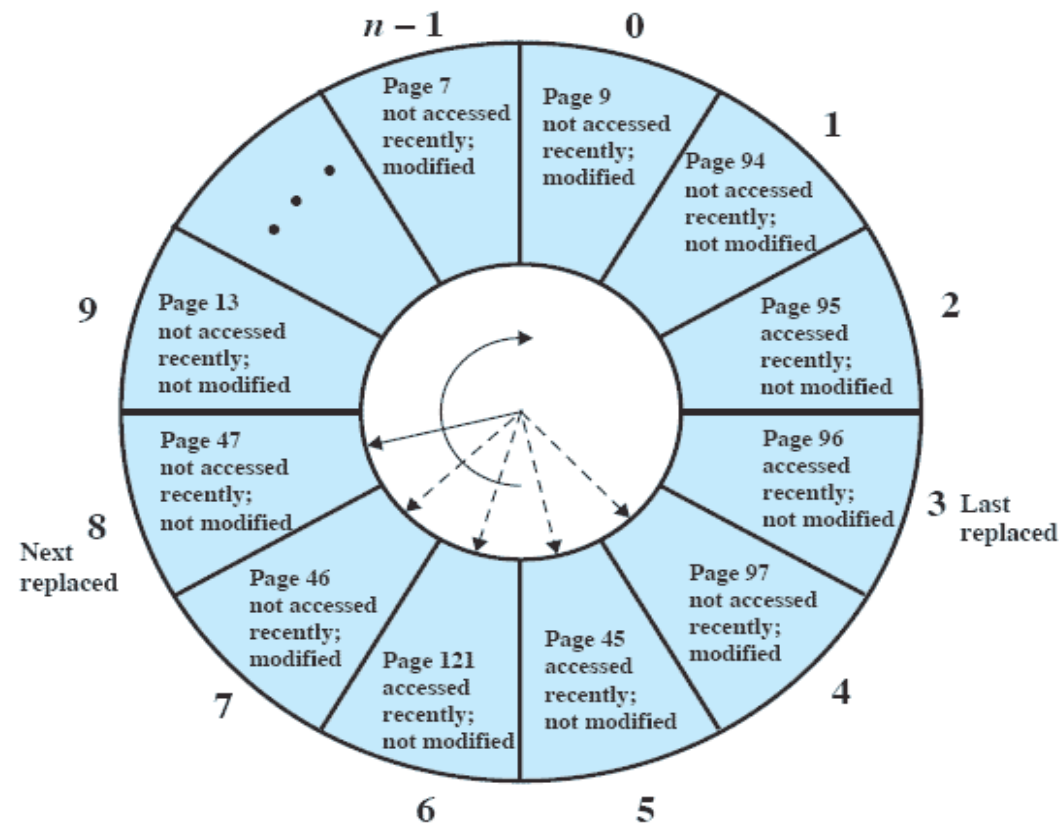
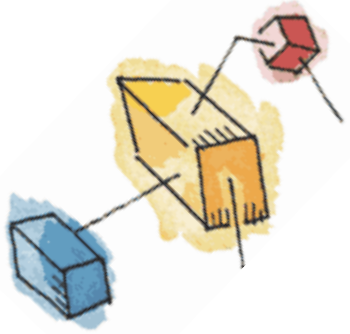
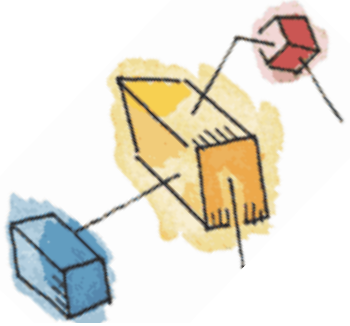


Figure 8.18 The Clock Page-Replacement Algorithm [GOLD89]





Comparison

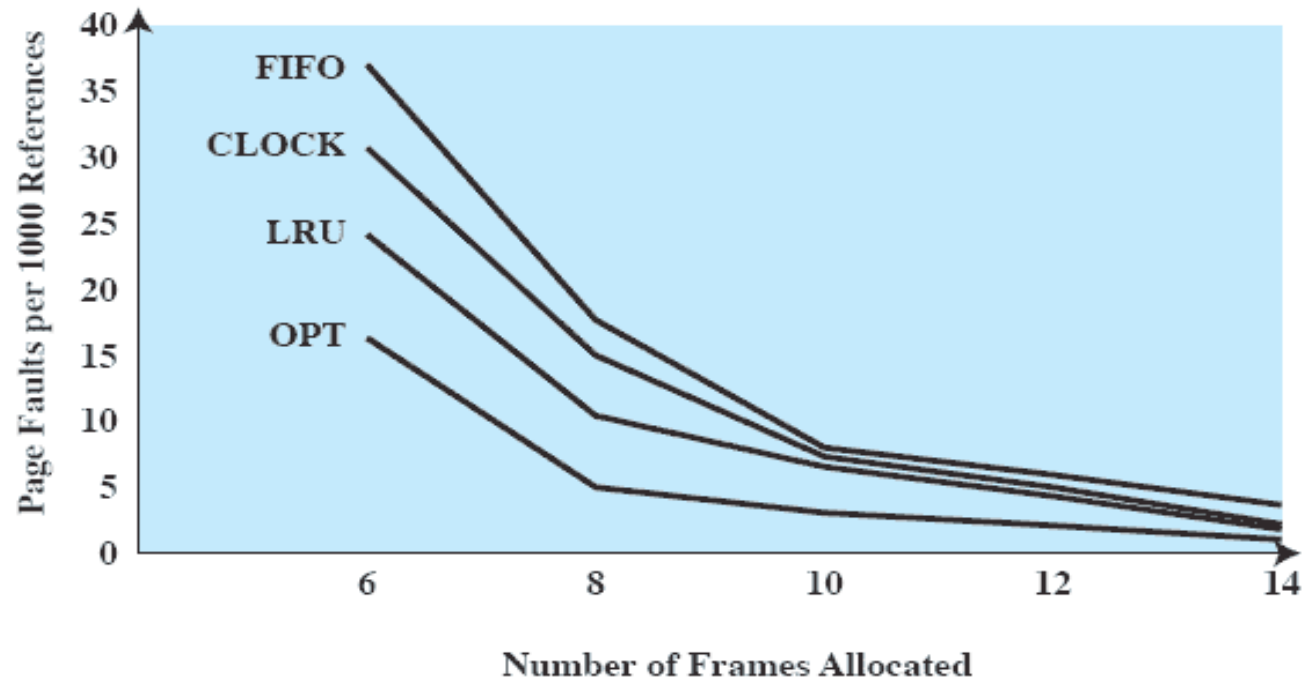
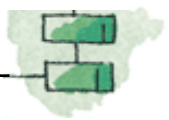


Figure 8.17 Comparison of Fixed-Allocation, Local Page Replacement Algorithms





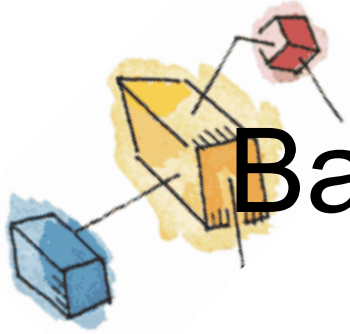
Behavior of Page Replacement Algorithms

Page address stream	2	3	2	1	5	2	4	5	3	2	5	2																																				
OPT	<table border="1"><tr><td>2</td></tr><tr><td></td></tr><tr><td></td></tr></table>	2			<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>1</td></tr></table>	2	3	1	<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table> F	2	3	5	<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table>	2	3	5	<table border="1"><tr><td>4</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table> F	4	3	5	<table border="1"><tr><td>4</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table>	4	3	5	<table border="1"><tr><td>4</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table>	4	3	5	<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table> F	2	3	5	<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table>	2	3	5	<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>5</td></tr></table>	2	3	5
2																																																
2																																																
3																																																
2																																																
3																																																
2																																																
3																																																
1																																																
2																																																
3																																																
5																																																
2																																																
3																																																
5																																																
4																																																
3																																																
5																																																
4																																																
3																																																
5																																																
4																																																
3																																																
5																																																
2																																																
3																																																
5																																																
2																																																
3																																																
5																																																
2																																																
3																																																
5																																																
LRU	<table border="1"><tr><td>2</td></tr><tr><td></td></tr><tr><td></td></tr></table>	2			<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>1</td></tr></table>	2	3	1	<table border="1"><tr><td>2</td></tr><tr><td>5</td></tr><tr><td>1</td></tr></table> F	2	5	1	<table border="1"><tr><td>2</td></tr><tr><td>5</td></tr><tr><td>1</td></tr></table>	2	5	1	<table border="1"><tr><td>2</td></tr><tr><td>5</td></tr><tr><td>4</td></tr></table> F	2	5	4	<table border="1"><tr><td>2</td></tr><tr><td>5</td></tr><tr><td>4</td></tr></table>	2	5	4	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>4</td></tr></table> F	3	5	4	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>2</td></tr></table> F	3	5	2	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>2</td></tr></table>	3	5	2	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>2</td></tr></table>	3	5	2
2																																																
2																																																
3																																																
2																																																
3																																																
2																																																
3																																																
1																																																
2																																																
5																																																
1																																																
2																																																
5																																																
1																																																
2																																																
5																																																
4																																																
2																																																
5																																																
4																																																
3																																																
5																																																
4																																																
3																																																
5																																																
2																																																
3																																																
5																																																
2																																																
3																																																
5																																																
2																																																
FIFO	<table border="1"><tr><td>2</td></tr><tr><td></td></tr><tr><td></td></tr></table>	2			<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td></td></tr></table>	2	3		<table border="1"><tr><td>2</td></tr><tr><td>3</td></tr><tr><td>1</td></tr></table>	2	3	1	<table border="1"><tr><td>5</td></tr><tr><td>3</td></tr><tr><td>1</td></tr></table> F	5	3	1	<table border="1"><tr><td>5</td></tr><tr><td>2</td></tr><tr><td>1</td></tr></table> F	5	2	1	<table border="1"><tr><td>5</td></tr><tr><td>2</td></tr><tr><td>4</td></tr></table> F	5	2	4	<table border="1"><tr><td>5</td></tr><tr><td>2</td></tr><tr><td>4</td></tr></table>	5	2	4	<table border="1"><tr><td>3</td></tr><tr><td>2</td></tr><tr><td>4</td></tr></table> F	3	2	4	<table border="1"><tr><td>3</td></tr><tr><td>2</td></tr><tr><td>4</td></tr></table>	3	2	4	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>4</td></tr></table> F	3	5	4	<table border="1"><tr><td>3</td></tr><tr><td>5</td></tr><tr><td>2</td></tr></table> F	3	5	2
2																																																
2																																																
3																																																
2																																																
3																																																
2																																																
3																																																
1																																																
5																																																
3																																																
1																																																
5																																																
2																																																
1																																																
5																																																
2																																																
4																																																
5																																																
2																																																
4																																																
3																																																
2																																																
4																																																
3																																																
2																																																
4																																																
3																																																
5																																																
4																																																
3																																																
5																																																
2																																																
CLOCK	<table border="1"><tr><td>2*</td></tr><tr><td></td></tr><tr><td></td></tr></table>	2*			<table border="1"><tr><td>2*</td></tr><tr><td>3*</td></tr><tr><td></td></tr></table>	2*	3*		<table border="1"><tr><td>2*</td></tr><tr><td>3*</td></tr><tr><td></td></tr></table>	2*	3*		<table border="1"><tr><td>2*</td></tr><tr><td>3*</td></tr><tr><td>1*</td></tr></table>	2*	3*	1*	<table border="1"><tr><td>5*</td></tr><tr><td>3</td></tr><tr><td>1</td></tr></table> F	5*	3	1	<table border="1"><tr><td>5*</td></tr><tr><td>2*</td></tr><tr><td>1</td></tr></table> F	5*	2*	1	<table border="1"><tr><td>5*</td></tr><tr><td>2*</td></tr><tr><td>4*</td></tr></table> F	5*	2*	4*	<table border="1"><tr><td>5*</td></tr><tr><td>2*</td></tr><tr><td>4*</td></tr></table>	5*	2*	4*	<table border="1"><tr><td>3*</td></tr><tr><td>2</td></tr><tr><td>4</td></tr></table> F	3*	2	4	<table border="1"><tr><td>3*</td></tr><tr><td>2*</td></tr><tr><td>4</td></tr></table>	3*	2*	4	<table border="1"><tr><td>3*</td></tr><tr><td>2</td></tr><tr><td>5*</td></tr></table> F	3*	2	5*	<table border="1"><tr><td>3*</td></tr><tr><td>2*</td></tr><tr><td>5*</td></tr></table>	3*	2*	5*
2*																																																
2*																																																
3*																																																
2*																																																
3*																																																
2*																																																
3*																																																
1*																																																
5*																																																
3																																																
1																																																
5*																																																
2*																																																
1																																																
5*																																																
2*																																																
4*																																																
5*																																																
2*																																																
4*																																																
3*																																																
2																																																
4																																																
3*																																																
2*																																																
4																																																
3*																																																
2																																																
5*																																																
3*																																																
2*																																																
5*																																																

F = page fault occurring after the frame allocation is initially filled

Figure 8.15 Behavior of Four Page-Replacement Algorithms

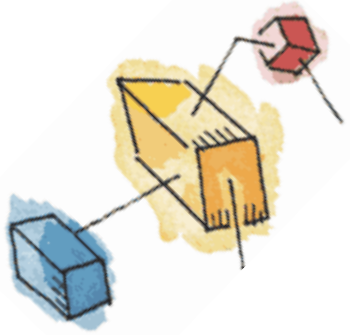




Basic Replacement Algorithms

- Page Buffering
 - Replaced page is added to one of two lists
 - Free page list if page has not been modified
 - Modified page list

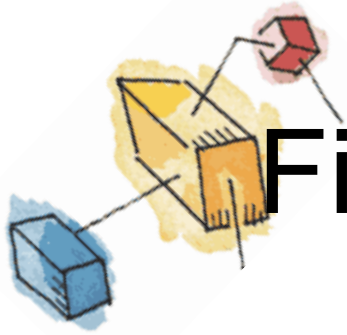




Resident Set Size

- Fixed-allocation
 - Gives a process a fixed number of pages within which to execute
 - When a page fault occurs, one of the pages of that process must be replaced
- Variable-allocation
 - Number of pages allocated to a process varies over the lifetime of the process

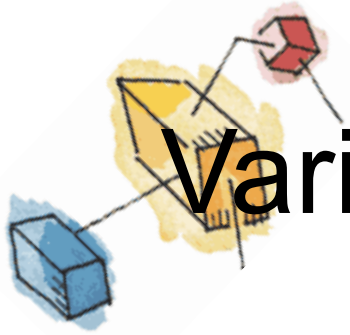




Fixed Allocation, Local Scope

- Decide ahead of time the amount of allocation to give a process
- If allocation is too small, there will be a high page fault rate
- If allocation is too large there will be too few programs in main memory
 - Processor idle time
 - Swapping

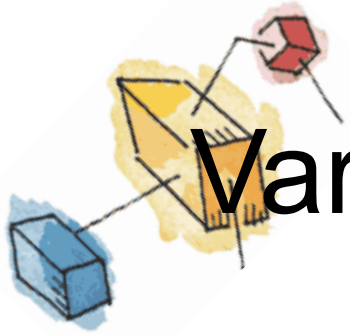




Variable Allocation, Global Scope

- Easiest to implement
- Adopted by many operating systems
- Operating system keeps list of free frames
- Free frame is added to resident set of process when a page fault occurs
- If no free frame, replaces one from another process

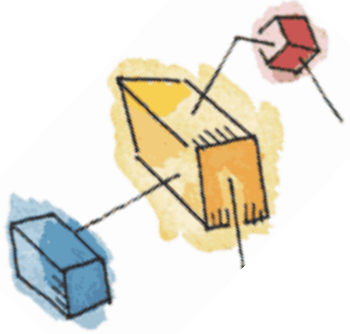




Variable Allocation, Local Scope

- When new process added, allocate number of page frames based on application type, program request, or other criteria
- When page fault occurs, select page from among the resident set of the process that suffers the fault
- Reevaluate allocation from time to time

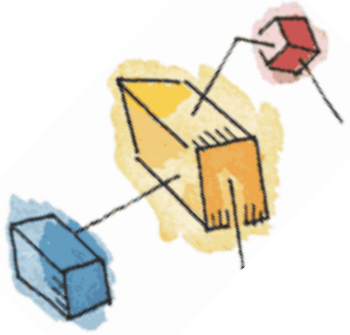




Cleaning Policy

- Demand cleaning
 - A page is written out only when it has been selected for replacement
- Precleaning
 - Pages are written out in batches

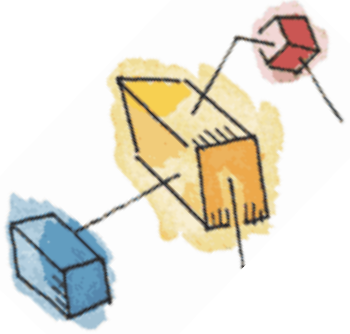




Cleaning Policy

- Best approach uses page buffering
 - Replaced pages are placed in two lists
 - Modified and unmodified
 - Pages in the modified list are periodically written out in batches
 - Pages in the unmodified list are either reclaimed if referenced again or lost when its frame is assigned to another page





Load Control

- Determines the number of processes that will be resident in main memory
- Too few processes, many occasions when all processes will be blocked and much time will be spent in swapping
- Too many processes will lead to thrashing



Multiprogramming

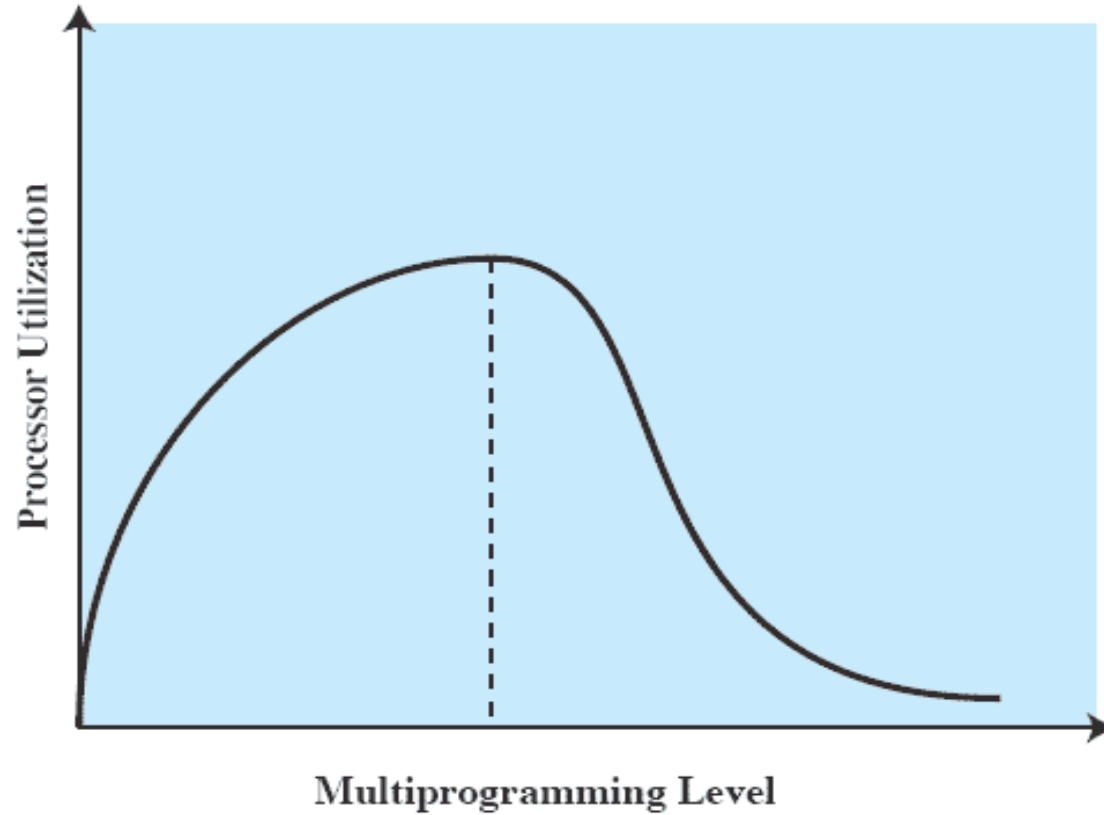
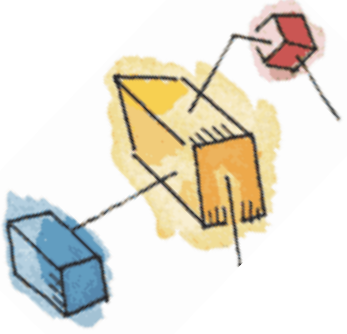
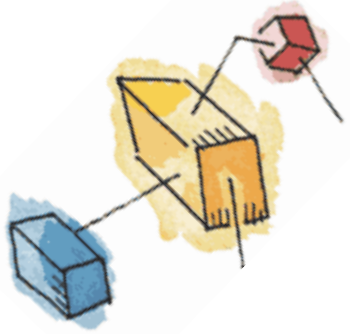


Figure 8.21 Multiprogramming Effects

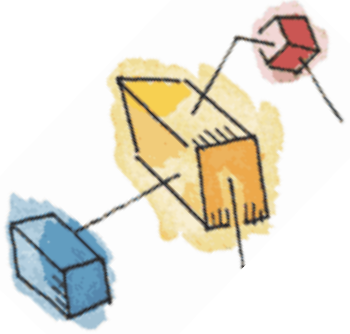




Process Suspension

- Lowest priority process
- Faulting process
 - This process does not have its working set in main memory so it will be blocked anyway
- Last process activated
 - This process is least likely to have its working set resident





Process Suspension

- Process with smallest resident set
 - This process requires the least future effort to reload
- Largest process
 - Obtains the most free frames
- Process with the largest remaining execution window

