Virtual Memory
Names, Virtual Addresses & Physical Addresses

Source Program

Name Space

Absolute Module

Virtual Address Space

Compile/Link tools
Names, Virtual Addresses & Physical Addresses

? t: Virtual Address Space ? Physical Address Space
Virtual Memory

• Uses dynamic address relocation/binding
  – Generalization of base-limit registers
  – Physical address corresponding to a compile-time address is not known until run time
• Idea is that only part of the address space is loaded as process executes
• This works because of program and data locality
Virtual Memory (cont)

- Use a dynamic virtual address map, $?_t$
Address Formation

• Translation system produces an address space, but addresses are *virtual* instead of physical

• A virtual address, \( x \):
  – Is mapped to \( y = t(x) \) if \( x \) is loaded at physical address \( y \)
  – Is mapped to ? if \( x \) is not loaded

• The map changes as the program executes

• \( t : \) Virtual Address \( \rightarrow \) Physical Address \( \{ ?, ? \} \)
Size of Blocks of Memory

• Fixed size: *Pages* are moved back and forth between primary and secondary memory

• Variable size: Programmer-defined *segments* are the unit of movement

• Paging is the commercially dominant form of virtual memory today
Paging

- A *page* is a fixed sized block of virtual addresses
- A *page frame* is a fixed size block of physical memory, the same size as a page
- When a virtual address in page i is referenced by the CPU
  - If page i is loaded at page frame j, the virtual address is relocated to page frame j
  - If page is not loaded, the OS interrupts the process and loads the page into a page frame
Addresses

• Suppose there are $G=2^{g+h}$ virtual addresses and $H=2^{j+h}$ physical addresses
  – Each page/page frame is $2^h$ addresses
  – There are $2^g$ pages in the virtual address space
  – $2^j$ page frames are allocated to the process
  – Rather than map individual addresses, $t$ maps the $2^g$ pages to the $2^j$ page frames
Address Translation

- Let $N = \{d_0, d_1, \ldots, d_{n-1}\}$ be the pages
- Let $M = \{b_0, b_1, \ldots, b_{m-1}\}$ be page frames
- Virtual address, $i$, satisfies $0 \leq i < G = 2^{g+h}$
- Physical address, $k = U 2^h + V$ ($0 \leq V < G = 2^h$)
  - $U$ is page frame number
  - $V$ is the line number within the page
  - $? : [0:G-1] \to <U, V> \to \{?\}$
  - Since every page is size $c = 2^h$
    - page number $= U = i/c$
    - line number $= V = i \mod c$
Address Translation (cont)

Virtual Address

Missing Page

Physical Address

CPU

Memory

MAR

MAR

g bits

h bits

Page #

Line #

j bits

Frame #

Line #

h bits

?_t

CPU Memory
Demand Paging

• Page fault occurs
• Process with missing page is interrupted
• Memory manager locates the missing page
• Page frame is unloaded (replacement policy)
• Page is loaded in the vacated page frame
• Page table is updated
• Process is restarted
Modeling Page Behavior

- Let \( r_1, r_2, r_3, \ldots, r_i, \ldots \) be a page reference stream
  - \( r_i \) is the \( i^{th} \) page # referenced by the process
  - The subscript is the virtual time for the process

- Given a page frame allocation of \( m \), the memory state at time \( t \), \( S_t(m) \), is set of pages loaded
  - \( S_t(m) = S_{t-1}(m) \oplus X_t - Y_t \)
    - \( X_t \) is the set of fetched pages at time \( t \)
    - \( Y_t \) is the set of replaced pages at time \( t \)
More on Demand Paging

- If $r_t$ was loaded at time $t-1$, $S_t(m) = S_{t-1}(m)$
- If $r_t$ was not loaded at time $t-1$ and there were empty page frames
  - $S_t(m) = S_{t-1}(m) \cap \{r_t\}$
- If $r_t$ was not loaded at time $t-1$ and there were no empty page frames
  - $S_t(m) = S_{t-1}(m) \cap \{r_t\} - \{y\}$
- The alternative is *prefetch* paging
Static Allocation, Demand Paging

- Number of page frames is static over the life of the process
- Fetch policy is demand
- Since $S_t(m) = S_{t-1}(m) \cap \{r_t\} - \{y\}$, the replacement policy must choose $y$ -- which uniquely identifies the paging policy
Random Replacement

• Replaced page, $y$, is chosen from the $m$ loaded page frames with probability $1/m$

Let page reference stream, $\mathcal{P} = 2031203120316457$

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0
1
2
Random Replacement

• Replaced page, $y$, is chosen from the $m$ loaded page frames with probability $1/m$

Let page reference stream, $? = 2031203120316457$

Frame  
0 2 2 2  
1 0 0  
2 3
Random Replacement

• Replaced page, y, is chosen from the m loaded page frames with probability 1/m

Let page reference stream, $\mathcal{R} = 2031203120316457$

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2
1 0 0 1
2 3 3
Random Replacement

- Replaced page, y, is chosen from the m loaded page frames with probability $1/m$

Let page reference stream, $\pi = 2031203120316457$

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2
1 0 0 1 1 1
2 3 3 3 0
Random Replacement

- Replaced page, \( y \), is chosen from the \( m \) loaded page frames with probability \( 1/m \)

Let page reference stream, \( \theta = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 2 2 2 2 3</td>
</tr>
<tr>
<td>1</td>
<td>0 0 1 1 1 1</td>
</tr>
<tr>
<td>2</td>
<td>3 3 3 0 0</td>
</tr>
</tbody>
</table>
Random Replacement

• Replaced page, $y$, is chosen from the $m$ loaded page frames with probability $1/m$

Let page reference stream, $? = 2031203120316457$

Frame | 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7 |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>_ 2 2 2 2 2 3 3 3</td>
</tr>
<tr>
<td>1</td>
<td>_ 0 0 1 1 1 1 1 1</td>
</tr>
<tr>
<td>2</td>
<td>_ 3 3 3 0 0 0 0 2</td>
</tr>
</tbody>
</table>
Random Replacement

• Replaced page, \( y \), is chosen from the \( m \) loaded page frames with probability \( 1/m \)

Let page reference stream, \( \hat{r} = 2031203120316457 \)

Frame: \[ 2 \ 0 \ 3 \ 1 \ 2 \ 0 \ 3 \ 1 \ 2 \ 0 \ 3 \ 1 \ 6 \ 4 \ 5 \ 7 \]

0 \_ 2 2 2 2 2 2 3 3 3 0 \_ 
1 \_ 0 0 1 1 1 1 1 1 1 \_ 
2 \_ 3 3 3 0 0 0 2 2 \_
Random Replacement

- Replaced page, \( y \), is chosen from the \( m \) loaded page frames with probability \( 1/m \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 2 2 2 3 3 3 0 0</td>
</tr>
<tr>
<td>1</td>
<td>0 0 1 1 1 1 1 1 1 1 3</td>
</tr>
<tr>
<td>2</td>
<td>3 3 3 0 0 0 2 2 2</td>
</tr>
</tbody>
</table>
Random Replacement

• Replaced page, y, is chosen from the m loaded page frames with probability $1/m$

Let page reference stream, $? = 2031203120316457$

<table>
<thead>
<tr>
<th>Frame</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Random Replacement

• Replaced page, y, is chosen from the m loaded page frames with probability 1/m

Let page reference stream, \( \pi = 2031203120316457 \)

Frame | 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7 |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 2 2 2 3 3 3 0 0 0 0 4</td>
</tr>
<tr>
<td>1</td>
<td>0 0 1 1 1 1 1 1 1 3 3 6 6</td>
</tr>
<tr>
<td>2</td>
<td>3 3 3 0 0 0 2 2 2 1 1 1</td>
</tr>
</tbody>
</table>
Random Replacement

- Replaced page, \( y \), is chosen from the \( m \) loaded page frames with probability \( \frac{1}{m} \)

Let page reference stream, \( \beta = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Random Replacement

• Replaced page, y, is chosen from the m loaded page frames with probability 1/m

Let page reference stream, \( \pi = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2 3 3 3 0 0 0 0 4 4 7
1 0 0 1 1 1 1 1 1 1 1 1 3 3 6 6 5 5
2 3 3 3 0 0 0 2 2 2 1 1 1 1 1 1
Random Replacement

• Replaced page, y, is chosen from the m loaded page frames with probability 1/m

Let page reference stream, \( ? = 2031203120316457 \)

Frame  2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2 3 3 3 0 0 0 4 4 7
1 0 0 1 1 1 1 1 1 1 3 3 6 6 5 5
2 3 3 3 0 0 0 2 2 2 1 1 1 1 1

13 page faults

• No knowledge of \( ? \) not perform well

• Easy to implement
Belady’s Optimal Algorithm

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{FWD}_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2</td>
</tr>
<tr>
<td>1</td>
<td>0 0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
Belady’s Optimal Algorithm

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{FWD}_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \text{FWD}_4(2) = 1 \]
\[ \text{FWD}_4(0) = 2 \]
\[ \text{FWD}_4(3) = 3 \]
Belady’s Optimal Algorithm

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{FWD}_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2
1 0 0 0 0
2 3 1

\[ \text{FWD}_4(2) = 1 \]
\[ \text{FWD}_4(0) = 2 \]
\[ \text{FWD}_4(3) = 3 \]
Belady’s Optimal Algorithm

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1(m)}} FWD_{t}(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 2 2 2 2</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>3 1 1 1 1</td>
</tr>
</tbody>
</table>
Belady’s Optimal Algorithm

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{FWD}_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3</td>
</tr>
<tr>
<td>2</td>
<td>3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>

\[
\text{FWD}_7(2) = 2 \\
\text{FWD}_7(0) = 3 \\
\text{FWD}_7(1) = 1
\]
Belady’s Optimal Algorithm

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} FWD_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2 2 2 2 2 0
1 0 0 0 0 0 3 3 3 3
2 3 1 1 1 1 1 1 1 1

\( FWD_{10}(2) = ? \)
\( FWD_{10}(3) = 2 \)
\( FWD_{10}(1) = 3 \)
Belady’s Optimal Algorithm

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1(m)}} FWD_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2 2 2 0 0 0
1 0 0 0 0 0 3 3 3 3 3 3
2 3 1 1 1 1 1 1 1 1 1

\[ FWD_{13}(0) = ? \]
\[ FWD_{13}(3) = ? \]
\[ FWD_{13}(1) = ? \]
Belady’s Optimal Algorithm

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1(m)}} FWD_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2 2 2 0 0 0 4 4 4
1 0 0 0 0 0 3 3 3 3 3 3 6 6 6 7
2 3 1 1 1 1 1 1 1 1 1 1 5 5

10 page faults

- Perfect knowledge of \( ? \) ? perfect performance
- Impossible to implement
Least Recently Used (LRU)

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} BKWD_t(x) \)

Let page reference stream, \( r = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 \underline{2} \underline{2} \underline{2}
1 \underline{0} \underline{0}
2 \underline{3}

\( BKWD_4(2) = 3 \)
\( BKWD_4(0) = 2 \)
\( BKWD_4(3) = 1 \)
Least Recently Used (LRU)

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{BKWD}_t(x) \)

Let page reference stream, \( \pi = 20312031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( \text{BKWD}_4(2) = 3 \)
\( \text{BKWD}_4(0) = 2 \)
\( \text{BKWD}_4(3) = 1 \)
Least Recently Used (LRU)

- Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} BKWD_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7

0 2 2 2 1 1

1 0 0 0 2

2 3 3 3

\( BKWD_5(1) = 1 \)
\( BKWD_5(0) = 3 \)
\( BKWD_5(3) = 2 \)
Least Recently Used (LRU)

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1(m)}} \text{BKWD}_t(x) \)

Let page reference stream, \( \mathbf{?} = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7

0 2 2 2 1 1 1

1 0 0 0 2 2

2 3 3 3 0

\( \text{BKWD}_6(1) = 2 \)
\( \text{BKWD}_6(2) = 1 \)
\( \text{BKWD}_6(3) = 3 \)
Least Recently Used (LRU)

• Replace page with maximal forward distance: \( y_t = \max_{x \in S \ t-1(m)} BKWD_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame

| 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7 |
| 0 2 2 2 1 1 1 3 3 3 0 0 0 6 6 6 7 |
| 1 0 0 0 2 2 2 1 1 1 3 3 3 4 4 4 |
| 2 3 3 3 0 0 0 2 2 2 1 1 1 5 5 |

Least Recently Used (LRU)

• Replace page with maximal forward distance: \( y_t = \max_{x \in S_{t-1}(m)} \text{BKWD}_t(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• Backward distance is a good predictor of forward distance -- locality
Least Frequently Used (LFU)

- Replace page with minimum use:
  \[ y_t = \min_{x \in S} \text{FREQ}(x) \]

Let page reference stream, \( ? = 2031203120316457 \)

Frame: 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7

| Frame | 2  | 0  | 3  | 1  | 2  | 0  | 3  | 1  | 2  | 0  | 3  | 1  | 6  | 4  | 5  | 7  |
|-------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
|       | 0  | 2  | 2  | 2  |    |    |    |    |    |    |    |    |    |    |    |
|       | 1  | 0  | 0  |    |    |    |    |    |    |    |    |    |    |    |    |
|       | 2  |    |    | 3  |    |    |    |    |    |    |    |    |    |    |    |

\[ \text{FREQ}_4(2) = 1 \]
\[ \text{FREQ}_4(0) = 1 \]
\[ \text{FREQ}_4(3) = 1 \]
Least Frequently Used (LFU)

- Replace page with minimum use:
  \[ y_t = \min_{x \in S} t-1(m)FREQ(x) \]

Let page reference stream, \( ? = 2031203120316457 \)

Frame:

<table>
<thead>
<tr>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>2</th>
<th>0</th>
<th>3</th>
<th>1</th>
<th>6</th>
<th>4</th>
<th>5</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ FREQ_4(2) = 1 \]
\[ FREQ_4(0) = 1 \]
\[ FREQ_4(3) = 1 \]
Least Frequently Used (LFU)

• Replace page with minimum use:
  \[ y_t = \min_{x \in S_{t-1}(m)} \text{FREQ}(x) \]

Let page reference stream, \( ? = 2031203120316457 \)

Frame  
| 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 2 2 2 2 2 |
| 1 | 0 0 1 1 1 |
| 2 | 3 3 3 0 |

\[
\text{FREQ}_6(2) = 2 \\
\text{FREQ}_6(1) = 1 \\
\text{FREQ}_6(3) = 1
\]
Least Frequently Used (LFU)

- Replace page with minimum use:
  \[ y_t = \min_{x \in S, t-1(m)} \text{FREQ}(x) \]

Let page reference stream, \( ? = 2031203120316457 \)

Frame 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7
0 2 2 2 2 2 2
1 0 0 1 1 1
2 3 3 3 0

FREQ_7(2) = ?
FREQ_7(1) = ?
FREQ_7(0) = ?
First In First Out (FIFO)

- Replace page that has been in memory the longest: \( y_t = \max_{x \in S_{t-1(m)}} \text{AGE}(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2</td>
</tr>
<tr>
<td>1</td>
<td>0 0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

\[
\text{AGE}_4(2) = 3 \\
\text{AGE}_4(0) = 2 \\
\text{AGE}_4(3) = 1
\]
First In First Out (FIFO)

• Replace page that has been in memory the longest: $y_t = \max_{x \in S_{t-1}(m)} \text{AGE}(x)$

Let page reference stream, $? = 2031203120316457$

Frame

<table>
<thead>
<tr>
<th>Frame</th>
<th>2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2 2 2 1</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>3 3</td>
</tr>
</tbody>
</table>

$\text{AGE}_4(2) = 3$
$\text{AGE}_4(0) = 2$
$\text{AGE}_4(3) = 1$
First In First Out (FIFO)

- Replace page that has been in memory the longest: \( y_t = \max_{xeS_{t-1}(m)} \text{AGE}(x) \)

Let page reference stream, \( ? = 2031203120316457 \)

Frame: 2 0 3 1 2 0 3 1 2 0 3 1 6 4 5 7

\[
\begin{array}{cccccccccccc}
0 & 2 & 2 & 2 & 1 & \\
1 & 0 & 0 & 0 & \\
2 & 3 & 3 & \\
\end{array}
\]

\[
\text{AGE}_5(1) = ? \\
\text{AGE}_5(0) = ? \\
\text{AGE}_5(3) = ?
\]
Belady’s Anomaly

Let page reference stream, \(? = 012301401234\)

<table>
<thead>
<tr>
<th>Frame</th>
<th>0 1 2 3 0 1 4 0 1 2 3 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0 3 3 3 4 4 4 4 4 4</td>
</tr>
<tr>
<td>1</td>
<td>1 1 1 0 0 0 0 0 2 2 2 2</td>
</tr>
<tr>
<td>2</td>
<td>2 2 2 1 1 1 1 1 3 3 3 3</td>
</tr>
</tbody>
</table>

Frame 0 1 2 3 0 1 4 0 1 2 3 4
0 0 0 0 0 0 0 4 4 4 4 3 3
1 1 1 1 1 1 1 0 0 0 0 4
2 2 2 2 2 2 1 1 1 1 1 1
3 3 3 3 3 3 3 2 2 2 2

- FIFO with m = 3 has 9 faults
- FIFO with m = 4 has 10 faults
Stack Algorithms

• Some algorithms are well-behaved
• Inclusion Property: Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$

LRU

Frame 0 1 2 3 0 1 4 0 1 2 3 4
0 0 0 0 3
1 1 1 1
2 2 2

Frame 0 1 2 3 0 1 4 0 1 2 3 4
0 0 0 0 0
1 1 1 1
2 2 2
3 3
## Stack Algorithms

- Some algorithms are well-behaved
- **Inclusion Property:** Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**LRU**

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Stack Algorithms

- Some algorithms are well-behaved
- Inclusion Property: Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$

<table>
<thead>
<tr>
<th>Frame</th>
<th>0 1 2 3 0 1 4 0 1 2 3 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 0 0 3 3 3</td>
</tr>
<tr>
<td>1</td>
<td>1 1 1 0 0</td>
</tr>
<tr>
<td>2</td>
<td>2 2 2 1</td>
</tr>
</tbody>
</table>

LRU

<table>
<thead>
<tr>
<th>Frame</th>
<th>0 1 2 3 0 1 4 0 1 2 3 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 0 0 0 0 0 0 0</td>
</tr>
<tr>
<td>1</td>
<td>1 1 1 1 1 1</td>
</tr>
<tr>
<td>2</td>
<td>2 2 2 2 2</td>
</tr>
<tr>
<td>3</td>
<td>3 3 3</td>
</tr>
</tbody>
</table>
Stack Algorithms

- Some algorithms are well-behaved
- Inclusion Property: Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$
Stack Algorithms

• Some algorithms are well-behaved
• Inclusion Property: Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

LRU

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Stack Algorithms

- Some algorithms are well-behaved
- Inclusion Property: Pages loaded at time $t$ with $m$ is also loaded at time $t$ with $m+1$

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>4</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementation

• LRU has become preferred algorithm

• Difficult to implement
  – Must record when each page was referenced
  – Difficult to do in hardware

• Approximate LRU with a reference bit
  – Periodically reset
  – Set for a page when it is referenced

• Dirty bit
Dynamic Paging Algorithms

• The amount of physical memory -- the number of page frames -- varies as the process executes

• How much memory should be allocated?
  – Fault rate must be “tolerable”
  – Will change according to the phase of process

• Need to define a placement & replacement policy

• Contemporary models based on working set
Working Set

• Intuitively, the working set is the set of pages in the process’s locality
  – Somewhat imprecise
  – Time varying
  – Given k processes in memory, let \( m_i(t) \) be # of pages frames allocated to \( p_i \) at time \( t \)
    • \( m_i(0) = 0 \)
    • \( \sum_{i=1}^{k} m_i(t) \leq |\text{primary memory}| \)
    • Also have \( S_t(m_i(t)) = S_t(m_i(t-1)) \Leftrightarrow X_t - Y_t \)
    • Or, more simply \( S(m_i(t)) = S(m_i(t-1)) \Leftrightarrow X_t - Y_t \)
Placed/Replaced Pages

- $S(m_i(t)) = S(m_i(t-1)) \ ? \ X_t - Y_t$
- For the missing page
  - Allocate a new page frame
  - $X_t = \{r_t\}$ in the new page frame
- How should $Y_t$ be defined?
- Consider a parameter, $\gamma$, called the window size
  - Determine $BKWD_t(y)$ for every $y$? $S(m_i(t-1))$
  - if $BKWD_t(y) \ ? \gamma$, unload $y$ and deallocate frame
  - if $BKWD_t(y) < \gamma$, do not disturb $y$
Working Set Principle

• Process $p_i$ should only be loaded and active if it can be allocated enough page frames to hold its entire working set

• The size of the working set is estimated using $\delta$
  – Unfortunately, a “good” value of $\delta$ depends on the size of the locality
  – Empirically this works with a fixed $\delta$
Example (? = 3)

<table>
<thead>
<tr>
<th>Frame</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

# 1
Example (? = 4)

Frame 0 1 2 3 0 1 2 3 0 1 2 3 4 5 6 7
0    0

#    1
Segmentation

- Unit of memory movement is:
  - Varibly sized
  - Defined by the programmer
- Two component addresses, <Seg#, offset>
- Address translation is more complex than paging

\[
\begin{align*}
?_t &: \text{segments x offsets} \rightarrow \text{Physical Address} \\
?_t(i, j) &= k
\end{align*}
\]
Segment Address Translation

• \( t \) : segments x offsets \( \to \) physical address \( \{ \} \)
• \( t(i, j) = k \)
• \( \text{: segments } \Rightarrow \text{ segment addresses} \)
• \( t(\text{segName}, j) = k \)
Segment Address Translation

• $t_i$: segments x offsets $\rightarrow$ physical address $\rightarrow\{?\}$
• $t(i, j) = k$
• $t$: segments $\rightarrow$ segment addresses
• $t(\text{segName}, j) = k$
• $t$: offset names $\rightarrow$ offset addresses
• $t(\text{segName}, \text{offsetName}) = k$
• Read implementation in Section 12.5.2
Address Translation

\(<\text{segmentName, offsetName}>\)

segment #

offset

Missing segment

Limit

Relocation

+?

Limit Base P

To Memory Address Register
Implementation

• Segmentation requires special hardware
  – Segment descriptor support
  – Segment base registers (segment, code, stack)
  – Translation hardware

• Some of translation can be static
  – No dynamic offset name binding
  – Limited protection
Multics

- Old, but still state-of-the-art segmentation
- Uses *linkage segments* to support sharing
- Uses dynamic offset name binding
- Requires sophisticated memory management unit
- See pp 368-371