Virtual memory policies

1. <u>The bring in policies</u>
   1. On demand → bring the page to the main memory from the disk only when it is needed. E.g., demand paging
   2. Anticipatory. E.g. pre-paging
2. <u>The replacement policies</u>
   - FIFO → First in first out
   - OPTIMAL → what a clairvoyant multi-level memory manager would do. Alternatively, construct the string of references and use it for a second execution of the program (with the same data as input).
   - LRU – Least Recently Used → replace the page that has not been referenced for the longest time.
   - MSU – Most Recently Used → replace the page that was referenced most recently
3. <u>How to evalute a policy</u> → use a string of references show what page is needed at each moment.
4. <u>The capacity of the main memory</u> is expressed as the number of frames.

Page replacement policies; Belady’s anomaly

- In the following examples we use a given string of references to illustrate several page replacement policies. We have <u>five pages, 0, 1, 2, 3, and 4</u>.
- The main memory has a capacity of
  - 3 frames, labeled 0,1,2
  - 4 frames, labeled 0, 1, 2, 3
- Once a frame has the “dirty bit” on it means that the page residing in that frame was modifies and must be written back to the secondary device, the disk before being replaced.
- The capacity of the primary device is important. One expects that increasing the capacity, in our case the number of frames in RAM leads to a higher hit ratio. That is not always the case as our examples will show. This is the Belady’s anomaly.
- Note: different results are obtained with a different string of references!!

FIFO Page replacement algorithm

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	3	3	3	4	4	4	4	4	
Frame 2	-	-	1	1	1	0	0	0	0	0	2	2	
Frame 3	-	-	-	2	2	2	1	1	1	1	1	3	
Page OUT	-	-	-	0	1	2	3	-	-	0	1	-	
Page IN	0	1	2	3	0	1	4	-	-	2	3	-	9

Frame 1	-	0	0	0	0	0	0	4	4	4	4	3	
Frame 2	-	-	1	1	1	1	1	1	0	0	0	0	
Frame 3	-	-	-	2	2	2	2	2	2	1	1	1	
Frame 4	-	-	-	-	3	3	3	3	3	3	2	2	
Page OUT	-	-	-	-	-	-	0	1	2	3	4	0	
Page IN	0	1	2	3	-	-	4	0	1	2	3	4	10

OPTIMAL page replacement algorithm

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	0	0	0	0	0	0	2	3	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	3	3	3	4	4	4	4	4	
Page OUT	-	-	-	2	-	-	3	-	-	0	2	-	
Page IN	0	1	2	3	-	-	4	-	-	2	3	-	7

Frame 1	-	0	0	0	0	0	0	0	0	0	0	3	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	2	2	2	2	2	2	2	2	
Frame 4	-	-	-	-	3	3	3	4	4	4	4	4	
Page OUT	-	-	-	-	-	-	3	-	-	-	0	-	
Page IN	0	1	2	3	-	-	4	-	-	-	3	-	6

LRU page replacement algorithm

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	0	0	0	0	0	0	0	3	
Frame 2	-	-	1	1	2	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	3	3	3	4	4	4	2	2	
Page OUT	-	-	-	1	-	2	3	-	-	4	0	1	
Page IN	0	1	2	3	-	1	4	-	-	2	3	4	9

Frame 1	-	0	0	0	0	0	0	0	0	0	0	0	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	2	2	2	4	2	2	2	2	
Frame 4	-	-	-	-	3	3	3	3	4	4	4	3	
Page OUT	-	-	-	-	-	-	2	-	-	-	4	0	
Page IN	0	1	2	3	-	-	4	-	-	-	3	4	7

LRU, OPTIMAL, MRU

- LRU looks only at history
- OPTIMAL “knows” not only the history but also the future.
- In some particular cases Most Recently Used Algorithm performs better than LRU.
- Example: primary device with 4 cells.

Reference string	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4
LRU	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
MRU	F	F	F	F	-	-	-	-	F	-	-	-	F	-	-

The OPTIMAL replacement policy keeps in the 3 frames the same pages as it does in case of the 4 frame primary memory. There are 5 pages, 0, 1, 2, 3, 4

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	0	0	0	0	0	0	2	3	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	3	3	3	4	4	4	4	4	
Page OUT	-	-	-	2	-	-	3	-	-	0	2	-	
Page IN	0	1	2	3	-	-	4	-	-	2	3	-	7

Frame 1	-	0	0	0	0	0	0	0	0	0	0	3	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	2	2	2	2	2	2	2	2	
Frame 4	-	-	-	-	3	3	3	4	4	4	4	4	
Page OUT	-	-	-	-	-	-	3	-	-	-	0	-	
Page IN	0	1	2	3	-	-	4	-	-	-	3	-	6

The LRU replacement policy keeps in the 3-frame primary memory the same pages as it does in case of the 4-frame primary memory.

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	0	0	0	0	0	0	0	3	
Frame 2	-	-	1	1	2	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	3	3	3	4	4	4	2	2	
Page OUT	-	-	-	2	-	-	3	-	-	0	2	-	
Page IN	0	1	2	3	-	1	4	-	-	2	3	4	9

Frame 1	-	0	0	0	0	0	0	0	0	0	0	0	
Frame 2	-	-	1	1	1	1	1	1	1	1	1	1	
Frame 3	-	-	-	2	2	2	2	4	2	2	2	2	
Frame 4	-	-	-	-	3	3	3	3	4	4	4	3	
Page OUT	-	-	-	-	-	-	2	-	-	-	4	0	
Page IN	0	1	2	3	-	-	4	-	-	-	3	4	7

The FIFO replacement policy does not keep in the 3-frame primary memory the same pages as it does in case of the 4-block primary memory

Time intervals	1	2	3	4	5	6	7	8	9	10	11	12	Total number of page faults
Reference string	0	1	2	3	0	1	4	0	1	2	3	4	
Frame 1	-	0	0	0	3	3	3	4	4	4	4	4	
Frame 2	-	-	1	1	1	0	0	0	0	0	2	2	
Frame 3	-	-	-	2	2	2	1	1	1	1	1	3	
Page OUT	-	-	-	0	1	2	3	-	-	0	1	-	
Page IN	0	1	2	3	0	1	4	-	-	2	3	-	9

Frame 1	-	0	0	0	0	0	0	4	4	4	4	3	
Frame 2	-	-	1	1	1	1	1	1	0	0	0	0	
Frame 3	-	-	-	2	2	2	2	2	2	1	1	1	
Frame 4	-	-	-	-	3	3	3	3	3	3	2	2	
Page OUT	-	-	-	-	-	-	0	1	2	3	4	0	
Page IN	0	1	2	3	-	-	4	0	1	2	3	4	10

How to avoid Belady’s anomaly

- The OPTIMAL and the LRU algorithms have the subset property, a primary device with a smaller capacity hold a subset of the pages a primary device with a larger capacity could hold.
- The subset property creates a total ordering. If the primary system has 1 blocks and contains page A a system with two block add page B, and a system with three blocks will add page C. Thus we have a total ordering

  A→B → C or (A,B,C)
- Replacement algorithms that have the subset property are called “stack” algorithms.
- If we use stack replacement algorithms a device with a larger capacity can never have more page faults than the one with a smaller capacity.

  m→ the pages held by a primary device with smaller capacity

  n → the pages held by a primary device with larger capacity

  m is a subset of n

Simulation analysis of page replacement algorithms

- Given a reference string *we can carry out the simulation for all possible cases when the capacity of the primary storage device varies from 1 to n with a single pass.*
- At each new reference to some page move to the top of the ordering and the pages that were above it either move down or stay in the same place as dictated by the replacement policy. We record whether this movement correspond to paging out, movement to the secondary storage.

Simulation of LRU page replacement algorithm

Time	1	2	3	4	5	6	7	8	9	10	11	12	
Reference string	**0**	**1**	**2**	**3**	**0**	**1**	**4**	**0**	**1**	**2**	**3**	**4**	

Stack contents after refrence	0	1	2	3	0	1	4	0	1	2	3	4	Total number of page faults
	-	0	1	2	3	0	1	4	0	1	2	3	
	-	-	0	2	2	3	0	1	4	0	1	2	
	-	-	-	0	1	2	3	3	3	4	0	1	
	-	-	-	-	-	-	2	2	2	5	4	0	

Size 1 in/out	**0/-**	**1/0**	**2/1**	**3/2**	**0/3**	**1/0**	**4/1**	**0/4**	**1/0**	**2/1**	**3/2**	**4/3**	**12**
Size 2 in/out	**0/-**	**1/-**	**2/0**	**3/1**	**0/2**	**1/3**	**4/0**	**0/1**	**1/4**	**2/0**	**3/1**	**4/2**	**12**
Size 3 in/out	**0/-**	**1/-**	**2/-**	**3/0**	**0/1**	**1/2**	**4/3**	**-/-**	**-/-**	**2/4**	**3/0**	**4/1**	**10**
Size 4 in/out	**0/-**	**1/-**	**2/-**	**3/-**	**-/-**	**-/-**	**4/2**	**-/-**	**-/-**	**2/3**	**3/4**	**4/0**	**8**
Size 5 in/out	**0/-**	**1/-**	**2/-**	**3/-**	**-/-**	**-/-**	**4/-**	**-/-**	**-/-**	**-/-**	**-/-**	**-/-**	**5**

Simulation of OPTIMUM

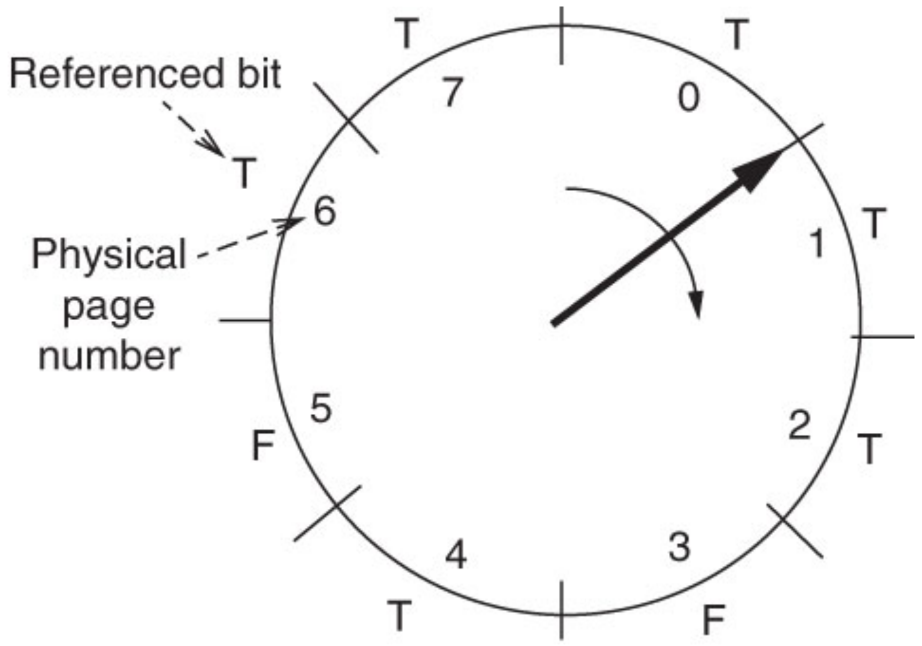
Time	1	2	3	4	5	6	7	8	9	10	11	12	
Reference string	**0**	**1**	**2**	**3**	**0**	**1**	**4**	**0**	**1**	**2**	**3**	**4**	

Stack contents after reference	0	1	2	3	0	1	4	0	1	2	3	4	Total number of page faults
	-	0	0	0	3	0	0	4	0	0	0	0	
	-	-	1	1	1	3	1	1	4	4	4	3	
	-	-	-	2	2	2	3	3	3	3	2	2	
	-	-	-	-	-	-	2	2	2	1	1	1	

Size 1 victim	**-**	**0**	**1**	**2**	**3**	**0**	**1**	**4**	**0**	**1**	**2**	**3**	**11**
Size 2 victim	**-**	**-**	**1**	**2**	**-**	**3**	**1**	**-**	**1**	**2**	**3**	**4**	**10**
Size 3 victim	**-**	**-**	**-**	**2**	**-**	**-**	**4**	**-**	**-**	**2**	**3**	**-**	**7**
Size 4 victim	**-**	**-**	**-**	**-**	**-**	**-**	**4**	**-**	**-**	**2**	**-**	**-**	**6**
Size 5 victim	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**5**

Clock replacement algorithm

- Approximates LRU with a minimum
  - Additional hardware: one reference bit for each page
  - Overhead
- Algorithm activated :
  - when a new page must be brought in move the pointer of a virtual clock in clockwise direction
  - if the arm points to a block with reference bit TRUE
    - Set it FALSE
    - Move to the next block
  - if the arm points to a block with reference bit FALSE
    - The page in that block could be removed (has not been referenced for a while)
    - Write it back to the secondary storage if the “dirty” bit is on (if the page has been modified.


Referenced bit
Physical page number
T
7
T
0
T
6
1
T
5
F
2
T
4
T
3
F