CIS 452 - Operating Systems Concepts

Nathan Bowman

Images taken from Silberschatz book

Page Replacement

Virtual address space can be larger than physical address space

Entire process must exist somewhere (disk), but only actively-used parts must reside in memory

Memory is similar to a "cache" for disk

Problem with any "cache" is that it will not hold as much as backing store

When only loading part of processes into memory, we do not just leave extra memory idle

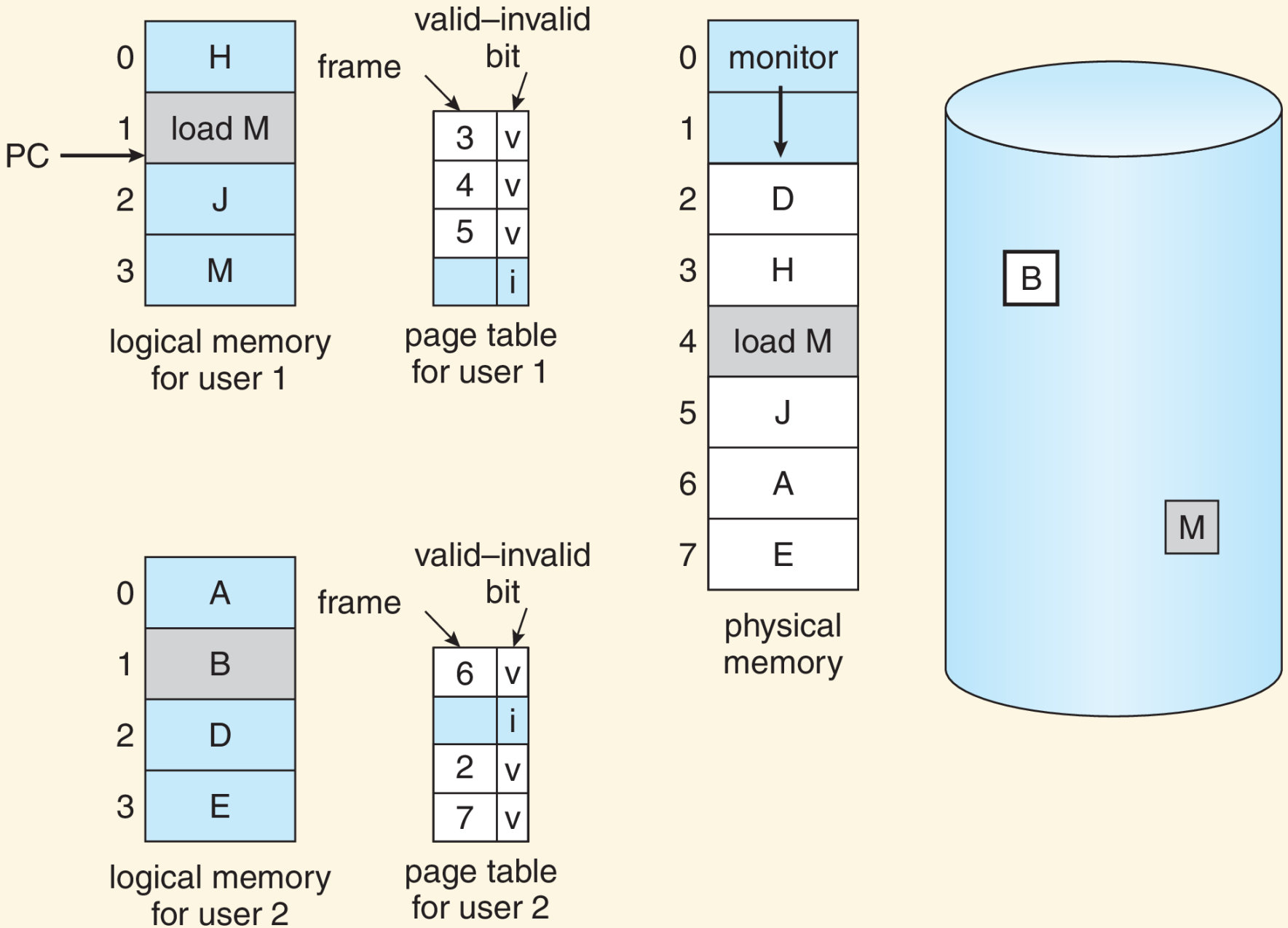
Degree of multiprogramming increased to allow better resource utilization

Also, I/O buffers take up a lot of memory

At some point, process memory requirement increases, such as during error handling

Page fault issued when process tries to read page not in memory

When OS handles page fault, free frame list is empty


0
H
1
load M
2
J
3
M
PC
logical memory
for user 1
frame
valid–invalid
bit
3 v
4 v
5 v
i
page table
for user 1
0
monitor
1
2
D
3
H
4
load M
5
J
6
A
7
E
physical
memory
B
M
0
A
1
B
2
D
3
E
logical memory
for user 2
frame
valid–invalid
bit
6 v
i
2 v
7 v
page table
for user 2

Need to remove page from memory to make room for new page

Page replacement -- page currently in memory moved out to backing store to make room for new page

Page replacement

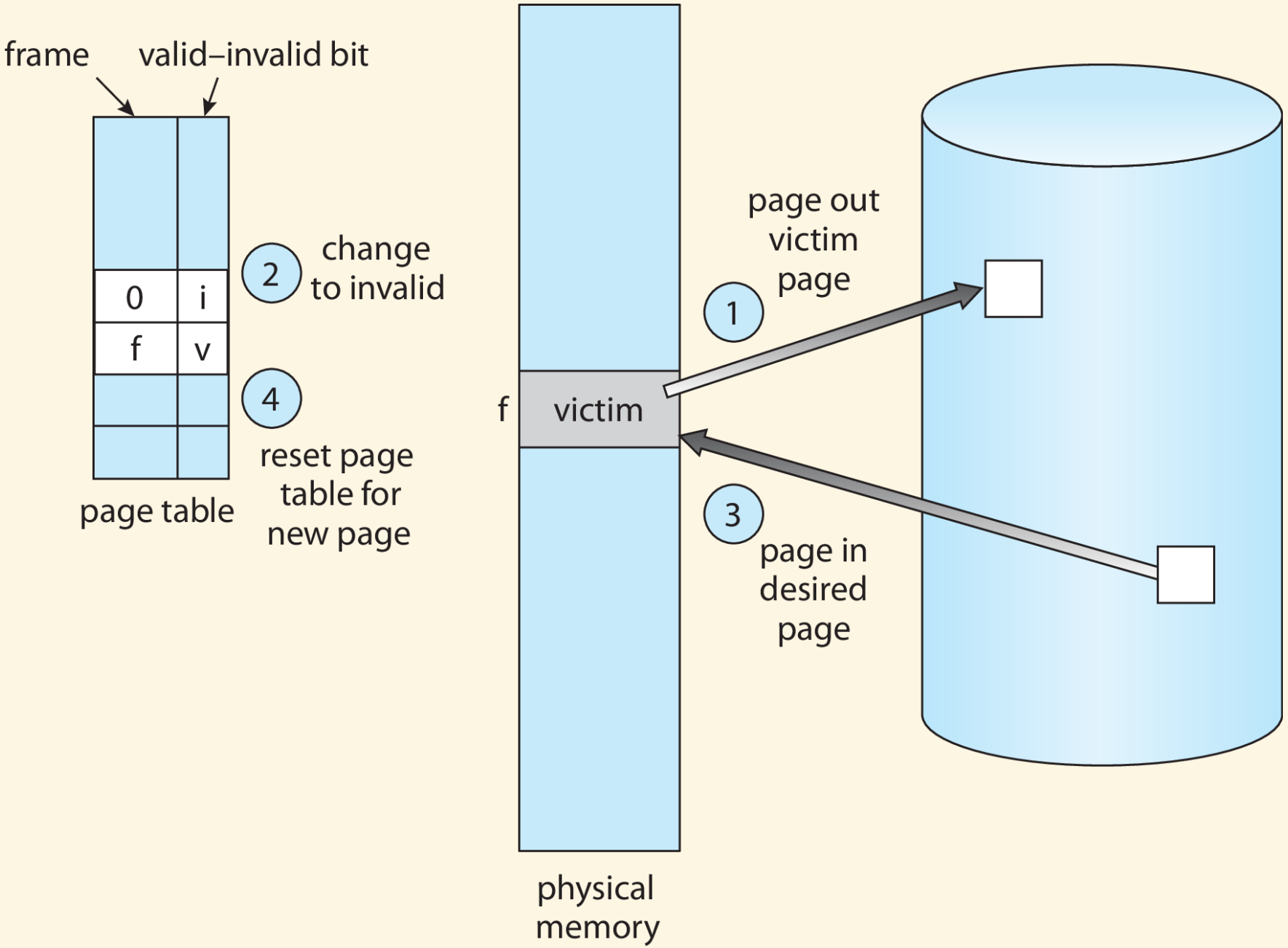
First, get rid of old page

- Find page in memory but not currently in use
- Write page to disk
- Update page table

Page replacement

Then, read new page

- Read new page into now-free frame
- Update page table


frame
valid–invalid bit
0
i
f
v
page table
2
change
to invalid
4
reset page
table for
new page
f
victim
physical
memory
1
page out
victim
page
3
page in
desired
page

Note that page requirement requires reading a page in *and* writing one out, doubling the memory movement required

Page fault time will therefore be roughly doubled, and effective access time will increase

We can keep time penalty somewhat lower by only writing out page when needed

If page has not been written to since it was read in from memory, no need to write it back to memory when evicted

Keep track by adding **modified bit** (or **dirty bit**) to each entry of page table

Whenever page is written to, modified bit is set

If modified bit is set for evicted page, page must be written back to disk

If modified bit is not set, system can save time by not writing page back to disk

Moving all of this memory around can be time-consuming

Always try to optimize slowest parts of system first

OS and architecture designers can get a lot of "bang for their buck" by improving page replacement

Key question we have not addressed: how to choose page to evict?

We will investigate in future lectures