Page-Replacement Algorithms

A ***page replacement algorithm*** picks a page to paged out and free up a frame

- **FIFO —** first-in, first-out
- **Optimal —** the one that leads to the least faults
- **LRU —** least-recently used
- **LRU** approximations

FIFO

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	0	1	2	2	3	0	4	2	3	0	0	0	1	2	2	2	7	0	1
	7	0	1	1	2	3	0	4	2	3	3	3	0	1	1	1	2	7	0
		7	0	0	1	2	3	0	4	2	2	2	3	0	0	0	1	2	7

1	2	3	4	1	2	5	1	2	3	4	5
1	2	3	4	1	2	5	5	5	3	4	4
	1	2	3	4	1	2	2	2	5	3	3
		1	2	3	4	1	1	1	2	5	5

1	2	3	4	1	2	5	1	2	3	4	5
1	2	3	4	4	4	5	1	2	3	4	5
	1	2	3	3	3	4	5	1	2	3	4
		1	2	2	2	3	4	5	1	2	3
			1	1	1	2	3	4	5	1	2

Belady's anomaly: more frames can be worse

Optimal

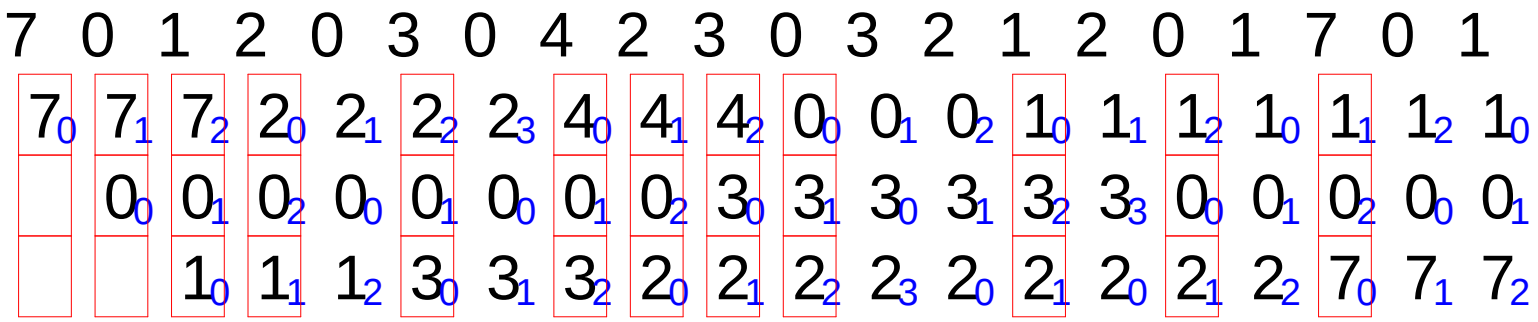
7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
	0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1

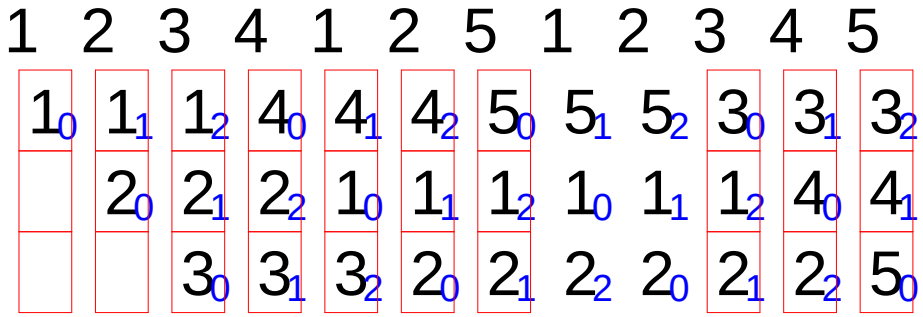
1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	1	1	1	1	1	1
	2	2	2	2	2	2	2	2	3	4	4
		3	4	4	4	5	5	5	5	5	5

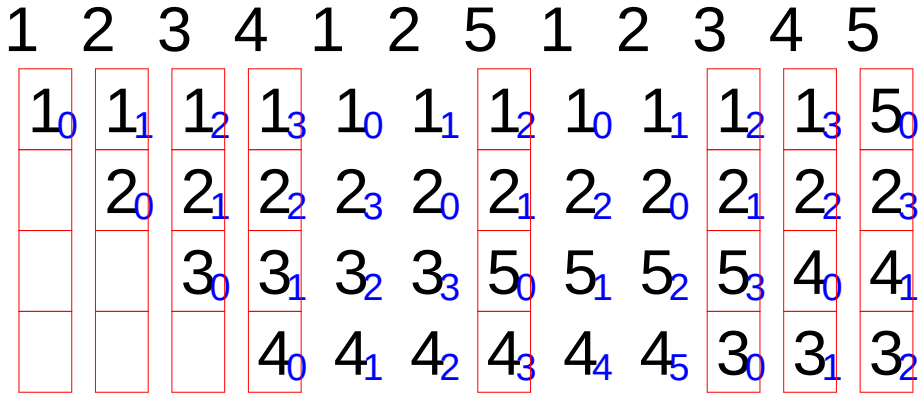
1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	1	1	1	1	1	1
	2	2	2	2	2	2	2	2	2	2	2
		3	3	3	3	3	3	3	3	4	4
			4	4	4	5	5	5	5	5	5

But how do you predict the future?

LRU

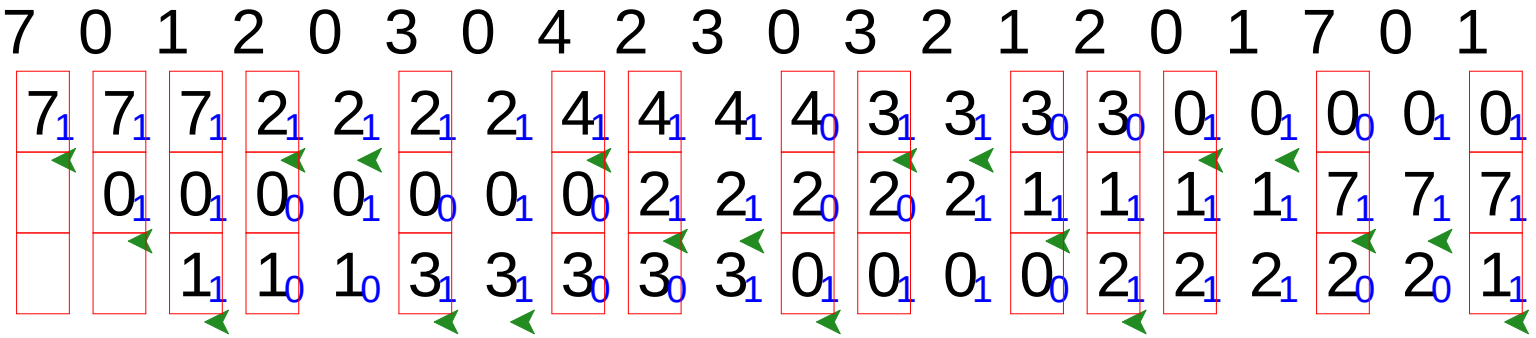


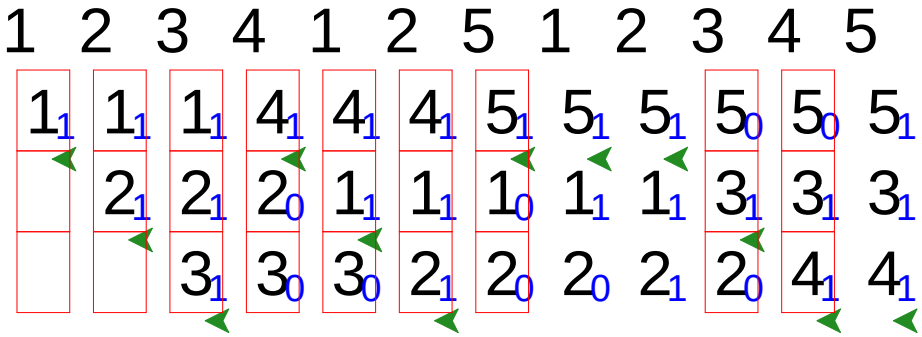


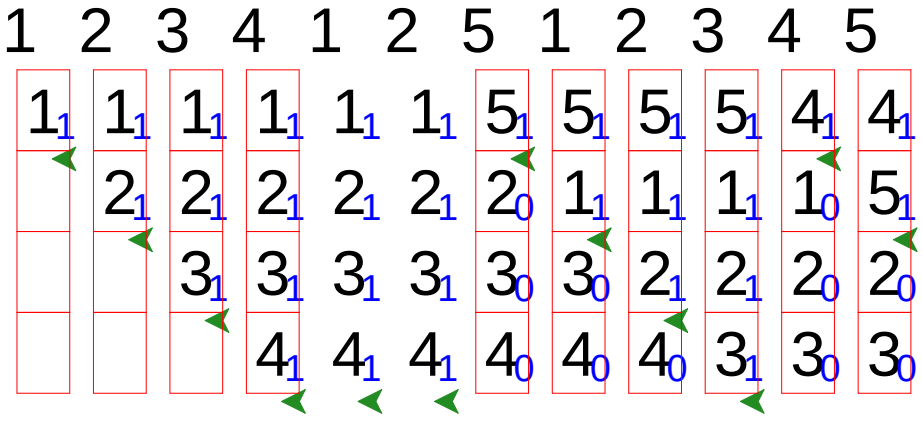


Searching all times is expensive on fault; keeping a sorted list is expensive on access

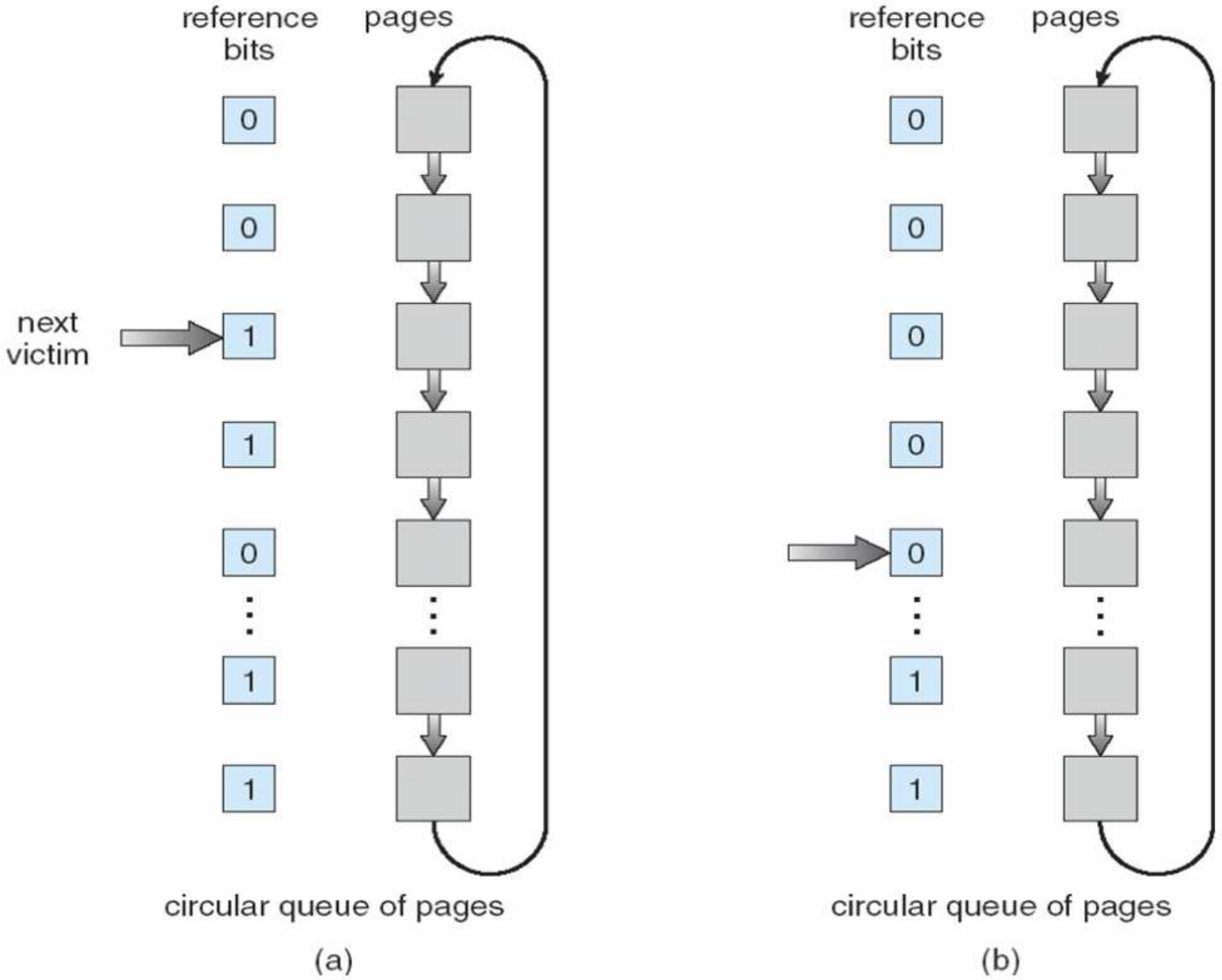
Second Chance







Second Chance



Enhanced Second Chance

Track both use and modifies (relative to copy on disk):

- Not used, Not modified — good to replace
- Not used, Modified — ok, but have to write out
- Used, Not modified — rather keep it
- Used, Modified — really rather keep it

Allocating Frames

Frame allocation can be defined orthogonal to page replacement:

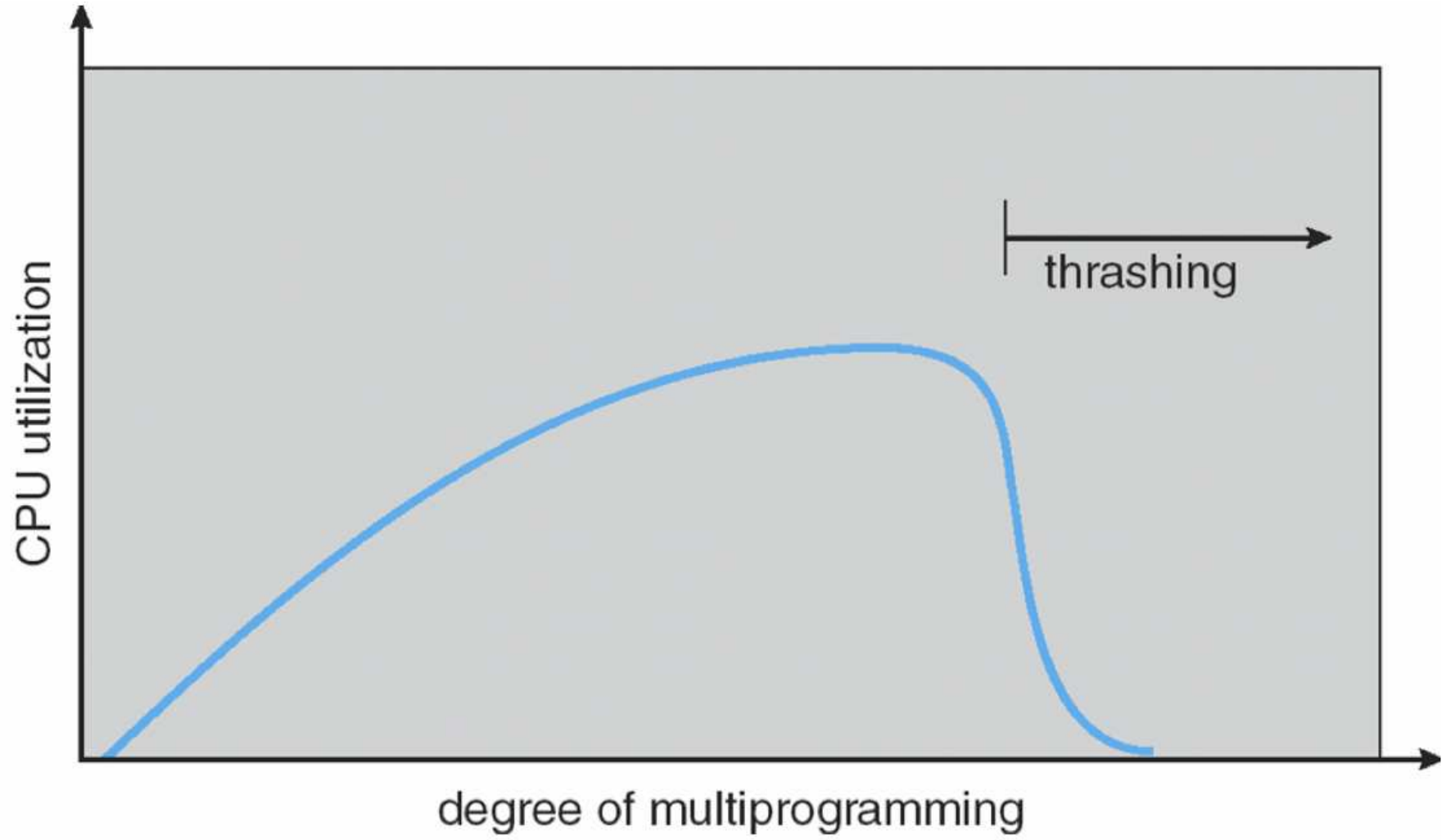
- Allocation algorithm determines candidate pages to evict
- Replacement algorithm picks a specific page to evict

- ***Local allocation***: each process has some frames
- ***Global allocation***: frames shared among all processes

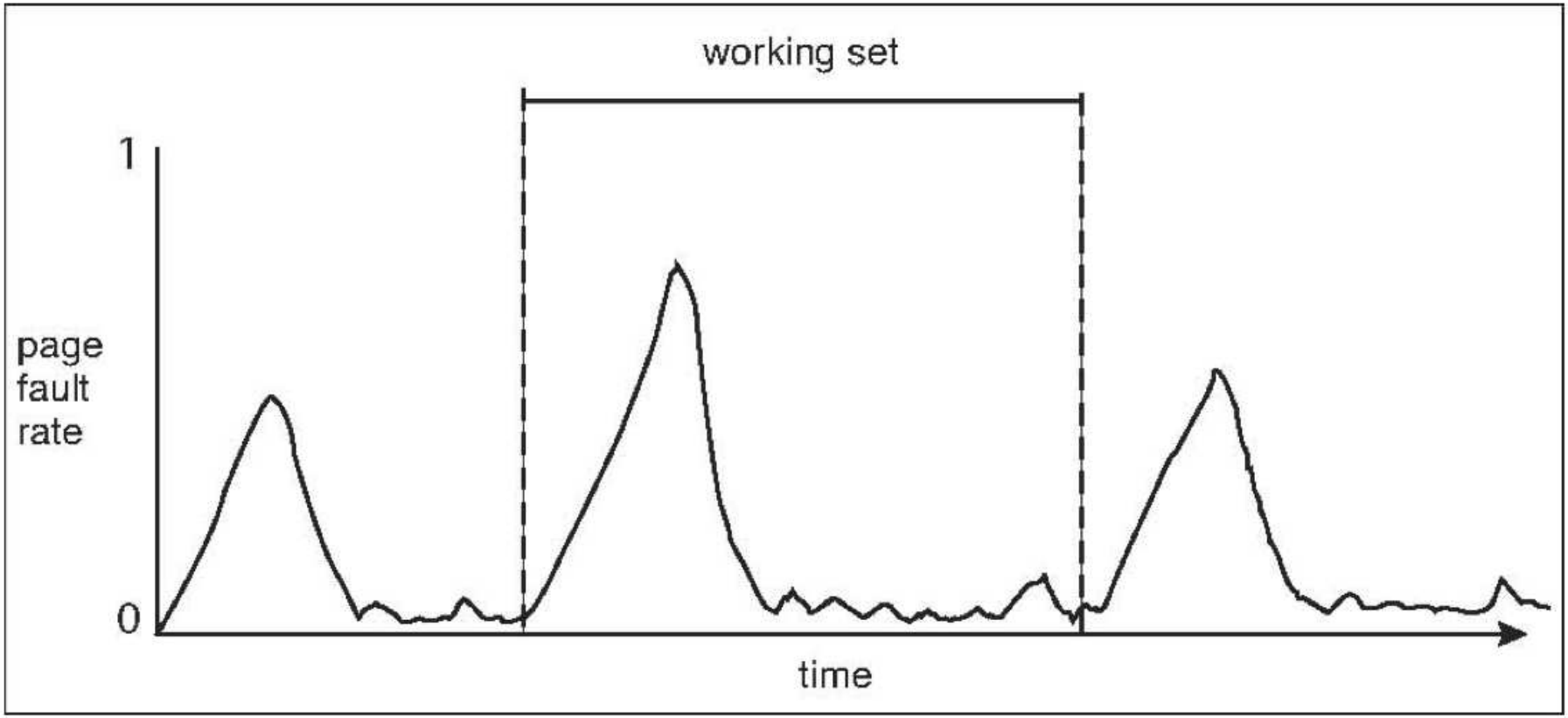
Thrashing

```
#define SIZE (1024*1024*32)

int main (void)
{
  while (1) {
    char *x;
    int i;

    x = (char *) malloc (SIZE);
    for (i=0; i<SIZE; i++) x[i] = 1;

    fork();
  }
}
```

Thrashing



Measuring Working Set



Interaction with I/O

- Some memory is used for a disk cache
- Communicating with an I/O device may require physical memory $\Rightarrow$ lock bits
- Memory-mapped files $\Rightarrow$ shared memory