

Operating Systems Exercise 2

2002-06-12

Dominique Emery, s97-602-056
Thomas Bocek, s99-706-319
Philip Iezzi, s99-714-354
Florian Caflisch, s96-920-525

Exercise 2.1

a) placement algorithms

Average length of search: (where n is the number of free blocks)

best fit:	n
worst fit:	n
first fit:	$(n+1)/2$
next fit:	$(n+1)/2$

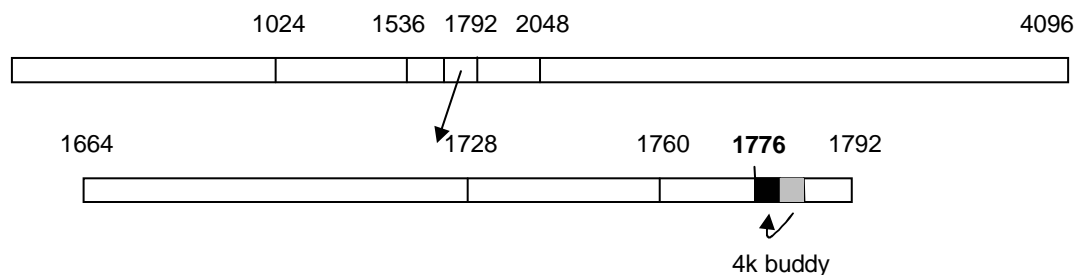
pros and cons of worst fit:

+	-
<ul style="list-style-type: none"> small internal fragmentation compared to best fit efficient implementation if we got an ordered list 	<ul style="list-style-type: none"> search time for holes is n not very powerful big processes may not have always free space (→ large external Fragmentation)

b) buddy system

binary address of the buddy:

011011110100



general expression for $buddy_k(x)$:

```
if ((x/2^k)%2 == 0)
    buddy_k(x) = x + 2^k
else
    buddy_k(x) = x - 2^k
```

Exercise 2.2

a) formal description for LIFO, LFU:

LIFO

$$|K| = m \quad \text{and} \quad Q = N^m \qquad q = (y_1, y_2, \dots, y_m)$$

$$g_{\text{LIFO}}(K, q, x) = \begin{cases} (K, q) & \text{if } x \in K \\ (K+x-y_1, q') & \text{if } x \notin K \quad \text{with } q'=(x, y_2, \dots, y_m) \end{cases}$$

LFU

$$|K| = m \quad \text{and} \quad Q = N^m \qquad q = (y_1, y_2, \dots, y_m)$$

$$g_{\text{LFU}}(K, q, x) = \begin{cases} (K, q') & \text{if } x \in K \\ & \text{with } q'=(y'_1, \dots, y'_m), \quad x=y_k, h'_k=h_k+1 \\ & h'_i=h_i \text{ for } i=1, \dots, m, \quad (i \neq k) \quad h'_i \geq h'_{i+1} \\ (K+x-y_m, q'') & \text{if } x \notin K \\ & \text{with } q''=(y_1, \dots, y_{m-1}, x) \end{cases}$$

b) formal definition of the dynamic page fault frequency algorithm

$K = W(t, \tau)$ with $|K| = \omega$ and $\gamma = r_1, \dots, r_t, r_{t+1}, \dots$
 $nf :=$ page faults
 $UL :=$ upper limit
 $LL :=$ lower limit

$$g_{\text{PFF}}(K, t, r_{t+1}) = \begin{cases} (K-Y, t+1) & \text{if } r_{t+1} \in W(T, \tau) \\ (K+r_{t+1}-Y, t+1) & \text{if } r_{t+1} \notin W(T, \tau) \end{cases}$$

and $Y = \{y \mid y \in W(t-1, \tau) \text{ and } y \notin W(t, \tau) \text{ d.h. interference-interval} > \tau$
and if $r_{t+1} \notin W(T, \tau) \Rightarrow nf+1$ and $(nf_{\text{periode}} > UL \Rightarrow \tau+1 \text{ or } nf_{\text{periode}} < LL \Rightarrow \tau-1)$

LRU can be used as technique to implement the page fault frequency algorithm. If lower limit and upper limit approaches, then PFF is a simple LRU.

c/d) LRU, LFU, working set algorithm

Table 1: LRU

Reference Sequence	8	1	4	7	5	1	2	3	4	3	2	8	6	7	4	1	8
Control State	8	1	4	7	5	1	2	3	4	3	2	8	6	7	4	1	8
	-	8	1	4	7	5	1	2	3	4	3	2	8	6	7	4	1
	-	-	8	1	4	7	5	1	2	2	4	3	2	8	6	7	4
	-	-	-	8	1	4	7	5	1	1	1	4	3	2	8	6	7
Nr. of Page Faults	1	2	3	4	5	5	6	7	8	8	8	9	10	11	12	13	14

Table 2: LFU

Reference Sequence	8	1	4	7	5	1	2	3	4	3	2	8	6	7	4	1	8
Control State	8	1	4	7	5	1	1	1	1	3	2	2	2	2	2	1	1
	-	8	1	4	7	5	2	3	4	1	3	3	3	3	3	2	2
	-	-	8	1	4	7	5	2	3	4	1	1	1	1	1	3	3
	-	-	-	8	1	4	7	5	2	2	4	8	6	7	4	4	8
Nr. of Page Faults	1	2	3	4	5	5	6	7	8	8	8	9	10	11	12	12	13
Frequency Counter for Pages	1	-	1	1	1	1	2	2	2	2	2	2	2	2	2	3	3
	2	-	-	-	-	-	1	1	1	1	2	2	2	2	2	2	2
	3	-	-	-	-	-	-	1	1	2	2	2	2	2	2	2	2
	4	-	-	1	1	1	1	-	-	1	1	1	-	-	-	1	1
	5	-	-	-	-	1	1	1	1	-	-	-	-	-	-	-	-
	6	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-
	7	-	-	-	1	1	1	1	-	-	-	-	-	-	1	-	-
	8	1	1	1	1	-	-	-	-	-	-	-	1	-	-	-	-

Table 3: Working Set Algorithm

Reference Sequence	8	1	4	7	5	1	2	3	4	3	2	8	6	7	4	1	8	
W(t,6)	1	-	6	5	4	3	6	5	4	3	2	1	-	-	-	-	6	5
	2	-	-	-	-	-	-	6	5	4	3	6	5	4	3	2	1	-
	3	-	-	-	-	-	-	-	6	5	6	5	4	3	2	1	-	-
	4	-	-	6	5	4	3	2	1	6	5	4	3	2	1	6	5	4
	5	-	-	-	-	6	5	4	3	2	1	-	-	-	-	-	-	-
	6	-	-	-	-	-	-	-	-	-	-	-	-	6	5	4	3	2
	7	-	-	-	6	5	4	3	2	1	-	-	-	-	6	5	4	3
	8	6	5	4	3	2	1	-	-	-	-	-	6	5	4	3	2	6
Nr. Page Faults	1	2	3	4	5	5	6	7	7	7	7	8	9	10	10	11	11	
$\omega(6)$	1	2	3	4	5	5	5	6	6	6	5	4	4	5	6	6	5	

e) comparison of algorithms with respect to performance and implementation

algorithm	performance	implementation
LIFO [last-in-first-out]	average	easy, stack
FIFO [first-in-first-out]	average the last page might be still in use → bad choice!	easy, queue
LFU [least-frequently-used]	pretty good	average, linked list, with frequency table, and sorting algorithm
MFU [most-frequently-used]	very bad	average, linked list, with frequency table, and sorting algorithm
LRU [least-recently-used]	good	average – heavy, circle linked list, or sorted linked list with timestamp (list must be updated on every memory reference)
MRU [most-recently-used]	very bad, old pages are kept available	average – heavy, circle linked list, or linked list with timestamp and sorted
working set	good, depending on the size of the working set	heavy, for each page a timestamp must be checked each time
page fault frequency	very good	very heavy, for each page a timestamp must be checked each time and the working set must be adjusted each time according to the working set limit.

Typical loads:

FIFO: batch system, best performance if pages are different, or if pages are the same, they must be close to each other.

LFU: best performance if there are stable peaks in frequency usage of pages, and if the memory size is exactly the width of this peak.

LRU: best performance if there are peaks in the usage of pages, and if the memory size is exactly the width of this peak.