OpSys: Oblig #2

120912 – Victor Rudolfsson

Contents

Chapter 7.5: Theory

7.5.1. Hva brukes en bitmap til i forbindelse med minnehåndtering i et page-basert minne?

A bit map is a map of bits that indicate what pages are used/unused. One bit represents one page, of for example 4Kb size.

7.5.2. Hvilket felter finnes i en pagetable entry?

Bit indicating absence / presence, modified-bit, referenced-bit, caching disabled bit and protection bits and the page frame number.

7.5.3. Hvilke prinsipper kan vi benytte for å redusere størrelsen på en pagetable i internminnet?

Multi-level Page Tables, where one page table links to other page tables and thus not all of them need to be in memory at the same time.

7.5.4. Affinity scheduling ("CPU pinning") reduserer antall cache misser. Reduseres også antall TLB misser? Reduseres også antall page faults? Begrunn svaret.

7.5.5. Hvor stor blir en bitmap i et page-inndelt minnesystem med pagestørrelse 4KB og internminne på 512MB?

Because one bit represents a 4Kb page, dividing the memory into how many 4Kb pieces are needed should result in how many bits are needed to represent the whole memory.

2^32 / 2^15 = 2^17 = 16Kb

7.5.6. Tanenbaum oppgave 3.5

For each of the following decimal virtual addresses, compute the virtual page number and offset for a 4K page and for an 8K page: 20000, 32768, 60000.

7.5.7. Tanenbaum oppgave 3.10

Suppose that a machine has 48-bit virtual addresses and 32-bit physical address.
(a) If pages are 4K, how many entries are in the page table if it has only a single level? Explain.

(b) TLB with 32 entries. Suppose that a program contains instructions that fit into one page and it sequentially reads long integer elements from an array that spans thousands of pages. How effective will the TLB be for this case?

Chapter 7: Labs

7.6.1

```
#!/bin/bash
echo "1 - Hvem er jeg og hva er navnet på dette scriptet"
me=`uname -a`
me=$me" "$0

echo "2 - Hvor lenge er det siden siste boot?"
started=`uptime`

echo "3 - Hvor mange prosesser og tråder finnes?"
procthreads=`cat /proc/stat | grep "processes" | awk '{print $2}'`

echo "4 - Hvor mange context switch'er fant sted siste sekund?"
switch1=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
switch2=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
count=$(($switch2-$switch1))

echo "5 - Hvor stor andel av CPU-tiden ble benyttet i kernel mode og i usermode i siste sekund?"
USER1=`cat /proc/stat | grep "cpu " |  awk '{print $2}'`
SYSTEM1=`cat /proc/stat | grep "cpu " |  awk '{print $4}'`
sleep 1
USER2=`cat /proc/stat | grep "cpu " |  awk '{print $2}'`
SYSTEM2=`cat /proc/stat | grep "cpu " |  awk '{print $4}'`
usertotal=$((USER2-USER1))
systotal=$((SYSTEM2+SYSTEM1))

echo "6 - Hvor mange interrupts fant sted siste sekund?"
intr1=`cat /proc/stat | grep "intr" | awk '{print $2}'`
sleep 1
intr2=`cat /proc/stat | grep "intr" | awk '{print $2}'`
interrupts=$((intr2-intr1))

echo "9 - Avslut dette scriptet"

switch1=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
sleep 1
switch2=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
```

```
count=$(($switch2-$switch1))

echo "Velg funksjon: "
while true; do
       read number
       case $number in
              1) echo $me; break;;
              2) echo $started; break;;
              3) echo $procthreads; break;;
              4) echo $count; break;;
              5) echo "System: "$systotal"User:"$usertotal" i siste sekund"; break;;
              6) echo $interrupts; break;;
              9) exit; break;;
       esac
done
```

7.6.2

```
#!/bin/bash

me=`uname -a`
me=$me" "$0
started=`uptime`
procthreads=`cat /proc/stat | grep "processes" | awk '{print $2}'`

switch1=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
switch2=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
count=$(($switch2-$switch1))

USER1=`cat /proc/stat | grep "cpu " |  awk '{print $2}'`
SYSTEM1=`cat /proc/stat | grep "cpu " |  awk '{print $4}'`
sleep 1
USER2=`cat /proc/stat | grep "cpu " |  awk '{print $2}'`
SYSTEM2=`cat /proc/stat | grep "cpu " |  awk '{print $4}'`
usertotal=$((USER2-USER1))
systotal=$((SYSTEM2+SYSTEM1))

intr1=`cat /proc/stat | grep "intr" | awk '{print $2}'`
sleep 1
intr2=`cat /proc/stat | grep "intr" | awk '{print $2}'`
interrupts=$((intr2-intr1))

switch1=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`
sleep 1
switch2=`cat /proc/stat | grep "ctxt" | awk '{print $2}'`

count=$(($switch2-$switch1))

menuoption=$(whiptail --title "Velg funksjon:" --menu "Velg funksjon" 20 88 16 \
"1" "Hvem er jeg og hva er navnet på dette scriptet" \
"2" "Hvor lenge er det siden siste boot?" \
"3" "Hvor mange prosesser og tråder finnes?" \
"4" "Hvor mange context switch'er fant sted siste sekund?" \
"5" "Hvor stor andel av CPU-tiden ble benyttet i kernel mode og i usermode i siste
sekund?" \
"6" "Hvor mange interrupts fant sted siste sekund?" \
"9" "Avslut dette scriptet" 3>&1 1>&2 2>&3)
```

```
case $menuoption in
	1) whiptail --title "Hvem er jeg?" --msgbox "Jeg er $me" 8 78;;
	2) whiptail --title "Siste boot" --msgbox "$started" 8 78;;
	3) whiptail --title "Antall prosesser og tråder" --msgbox "$procthreads" 8 78;;
	4) whiptail --title "Antall context-switch'er siste sekund" --msgbox "$count" 8 78;;
	5) whiptail --title "Usermode & System siste sekund" --msgbox "System: $systotal
User:$usertotal" 8 78;;
	6) echo $interrupts;;
esac
```

Chapter 8.5: Theory

8.5.1. Hva menes med page replacement? Beskriv hva som skjer ved denne aktiviteten.

8.5.2. Hva menes med working set og thrashing i forbindelse med minnehåndtering?

8.5.3. Hva er hensikten med et swapområde?

To allow data that isn't being actively used in memory to be swapped from memory to disk, freeing up memory until a program tries to access it again, at which point it can be swapped back into memory.

8.5.4. Tanenbaum oppgave 3.28

A computer has 4 page frames. The time of loading, time of last access, and the R and M bits for each page are (the times are in clock ticks):

(a) Which page will NRU replace?

(b) Which page will FIFO replace?

(c) Which page will LRU replace?

(d) Which page will second chance replace?

8.5.5. Tanenbaum oppgave 3.9

32-bit address space and 8K page size, the page table is entirely in hardware, with one 32-bit word pr entry. When a process starts, the page table is copied from memory to hardware, at one word every 100 nsec. Each process runs for 100 msec (incl the time to load the page table), what fraction of CPU time is devoted to loading the page table?

Chapter 8: Labs

8.6.1

```
#!/bin/bash
read number
if [ $number -lt $((2**10)) ]
    then
        echo "$number B"
    elif [ $number -lt $((2**20)) ]
    then
        echo "$(($number/2**10)) KB"
    elif [ $number -lt $((2**30)) ]
    then
        echo "$(($number/2**20)) MB"
    elif [ $number -lt $((2**40)) ]
    then
        echo "$(($number/2**30)) GB"
fi
```

8.6.2

```
#!/bin/bash
read number
if [ $number -lt $((10**3)) ]
    then
        echo "$number ns"
    elif [ $number -lt $((10**6)) ]
    then
        echo "$(($number/10**3)) µs"
    elif [ $number -lt $((10**9)) ]
    then
        echo "$(($number/10**6)) ms"
    else
        echo "$(($number/10**9)) seconds"
fi
```

8.6.4

```
#!/bin/bash

for pid in "$@"
do
    VmSize=$(cat /proc/$pid/status | grep "VmSize" | awk '{print $2}')
    VmData=$(cat /proc/$pid/status | grep "VmData" | awk '{print $2}')
    VmStk=$(cat /proc/$pid/status | grep "VmStk" | awk '{print $2}')
    VmExe=$(cat /proc/$pid/status | grep "VmExe" | awk '{print $2}')
    VmLib=$(cat /proc/$pid/status | grep "VmLib" | awk '{print $2}')
    VmRSS=$(cat /proc/$pid/status | grep "VmRSS" | awk '{print $2}')
    VmPTE=$(cat /proc/$pid/status | grep "VmPTE" | awk '{print $2}')
    TotalVirtual=$(echo $VmData'+'$VmStk'+'$VmExe | bc)

    FileName=$pid-`date -u +%Y%m%d-%H%M%S`.meminfo
    echo "*********** Minne info om prosess med PID $pid" > $FileName
    echo "Total bruk av virtuelt minne (VmSize): $VmSize KB" >> $FileName
    echo "Mengde privat virtuelt minne (VmData+VmStk+VmExe): $TotalVirtual KB" >> $FileName
    echo "Mengde shared virtuelt minne (VmLib): $VmLib KB" >> $FileName
    echo "Total bruk av fysisk minne (VmRSS): $VmRSS KB" >> $FileName
    echo "Mengde fysisk minne som benyttes till page table (VmPTE): $VmPTE KB" >> $FileName
done
```

Chapter 9: Theory

1. Nevn minst tre eksempler på filattributter/filmetadata.

- Last Accessed
- Last Modified
- Created

2. Hva menes med ekstern og intern fragmentering i forbindelse med en fil?

3. Tanenbaum oppgave 4.11

One way to use contiguous allocation of the disk and not to suffer from holes is to compact the disk every time a file is removed. Since all files are contiguous, copying a file requires a seek and rotational delay to read the file, followed by the transfer at full speed. Writing the file back requires the same work. Assuming a seek time of 5 msec, a rotational delay of 4 msec, a transfer rate of 8 MB/sec, and an average file size of 8 KB, how long does it take to read a file into main memory and then write it back to the disk at a new location? Using these numbers, how long would it take to compact half of a 16-GB disk?

4. Tanenbaum oppgave 4.12

In light of the answer to the previous question, does compacting the disk ever make sense?

5. Tanenbaum oppgave 4.33

How many disk operations are needed to fetch the i-node for the file /usr/ast/courses/os/handout.t? Assume that the i-node for the root directory is in memory, but nothing else along the path is in memory. Also assume that all directories fit in one disk block.

6. Et prinsipp som nyttes i forbindelse med free-space management i filsystemer er basert på bitmap prinsippet. Forklar kort hvordan dette fungerer. Anta videre et 32-bits system og en bitmap som vist under. Finn nummeret på første ledige diskblokk når følgende bitmap er gitt.

Chapter 9: Lab

9.4.1

```
#!/bin/bash
full=$(df $1 | grep / | awk '{print $5}')
filecount=$(find $1/ -type f -print | wc -l)
filesystem=$(df $1 | grep / | awk '{print $1}')
mountpoint=$(mount | grep $filesystem | awk '{print $3}')
largestfile=$(find $1/ -type f -printf '%s %p\n' | sort -nr | head -n 1)
largestfilesize=$(echo $largestfile | awk '{print $1}')
largestfilename=$(echo $largestfile | awk '{f = $1; $1 = ""; print $0}')
avgfilesize=$(find $1/ -type f -printf '%s %p\n' | gawk '{sum += $1; n++;} END {print
sum/n;}')
hardlinks=$(find $1/ -type f -printf '%n %p\n' | awk 'BEGIN {file="";max=0} {if($1>max)
file=$2;max=$1} END {print file " " max}')

echo "Partisjonen $1 befinner seg på er $full full"
echo "Det finnes $filecount filer."
echo "Den største er $largestfilename som er $largestfilesize ("`echo $largestfilesize |
./human-readable-byte.bash`") stor."
echo "Gjennomsnittlig filstørrelse er $avgfilesize ("`echo $avgfilesize | ./human-readable-
byte.bash`") stor"
echo "Filen "`echo $hardlinks | awk '{print $1}'`" har flest hardlinks, den har "`echo
$hardlinks | awk '{print $2}'`
```

9.4.2

- | grep ^q
- | grep q$
- | grep .*q.*q.*
- | grep '^[^aw].*o.*o.*'
- | grep '^[a-zA-Z0-9]\{7,8\}$'

9.4.3

```
#!/bin/bash
find $1/ -type f -printf '%p\n' | grep "[ÅåÆæØø\s*]"
```

Chapter 10.5: Theory

10.5.1. Hvor store diskblokkadresser har vi i windows filsystemet NTFS?

48 bit

10.5.2. Hva forståes med MFT i NTFS filsystemet?

MFT, the Master File Table, contains metadata about all files such as name, creation date & time, access permissions, and size.

10.5.3. Tanenbaum oppgave 4.3