Homework 6

Due: 11:59 PM, Wednesday, October 30, 2002

Your answers should be submitted as a postscript file `hw06.ps` or a PDF file `hw06.pdf`. Please use these filenames!

1. Tanenbaum, p. 266, Question 32. (1 point)

2. (4 points) Consider a demand paging system with the following time-measured utilizations:

CPU	20%
Paging disk	97.7%
Other I/O devices	5%

   For each of the following, say whether it will (or is likely to) improve CPU utilization:

   (a) install a faster CPU

   (b) install a bigger paging disk

   (c) increase the degree of multiprogramming

   (d) decrease the degree of multiprogramming

   (e) install more main memory

   (f) install a faster hard disk, or multiple controllers with multiple hard disks

   (g) add prepaging to the page-fetch algorithms

   (h) increase the page size

3. Read the handout containing the excerpt from McKusick, Bostic, Karels, and Quarterman. It describes the virtual memory management of the BSD 4.4 system, upon which FreeBSD is based. The code that implements virtual memory for FreeBSD is located in the `/sys/vm` directory of your favorite lab machine. Browse the source code to find the implementations of the facilities described in the handout. There is nothing to turn in for this question.

4. (10 points) Consider the following program segment, written in a C-like language:

```
const int n=10;
int i, j, A[n], B[n], C[n], temp;

for (i=1; i<=n; i++) {
  A[i]=i;
  B[i]=n-i+1;
}
for (i=1; i<=n; i++) {
```

```
    temp=0;
    for (j=i; j<=n; j++) {
      temp=temp+A[n+i-j]*B[j];
    }
    C[i]=temp;
  }
```

Using a machine with registers denoted by Ri and a fixed instruction size of 1 word per instruction, the machine language version of this program is loaded in virtual address space (with page size 4K, *i.e.*, 1024 words) as follows:

```
0x2FBC  (R1) <- ONE            Index i
0x2FC0  (R2) <- n              Loop bound
0x2FC4  compare R1,R2          Test i>n
0x2FC8  branch_greater * + 0x20
0x2FCC  A(R1) <- (R1)          Compute A[i]
0x2FD0  (R0) <- n              Compute B[i]
0x2FD4  (R0) <- (R0) - (R1)
0x2FD8  (R0) <- (R0) + ONE
0x2FDC  B(R1) <- (R0)
0x2FE0  (R1) <- (R1) + ONE     Increment i
0x2FE4  branch * - 0x20
0x2FE8  (R1) <- ONE            Index i
0x2FEC  (R2) <- n              Loop bound
0x2FF0  compare R1,R2          Test i>n
0x2FF4  branch_greater * + 0x50
0x2FF8  (R0) <- ZERO           temp <- 0
0x2FFC  temp <- (R0)
0x3000  (R3) <- (R1)           Index j
0x3004  (R4) <- n              Loop bound
0x3008  compare R3,R4          Test j>n
0x300C  branch_greater * + 0x20
0x3010  (R0) <- n              Compute A[n+i-j]
0x3014  (R0) <- (R0) + (R1)
0x3018  (R0) <- (R0) - (R3)
0x301C  (R5) <- A(R0)
0x3020  (R6) <- B(R3)          Compute B[j]
0x3024  (R5) <- (R5) * (R6)
0x3028  (R5) <- (R5) + temp
0x302C  temp <- (R5)
0x3030  (R3) <- (R3) + ONE     Increment j
0x3034  branch * - 0x20
0x3038  C(R1) <- (R5)          Compute C[i]
0x303C  (R1) <- (R1) + ONE     Increment i
0x3040  branch * - 0x50
  ...
0x6000  Storage for C
```

```
0x7000  Storage for ONE
0x7004  Storage for n
0x7008  Storage for temp
0x700C  Storage for ZERO
0x8000  Storage for A
0x9000  Storage for B
```

Upon execution of this program segment, the following reference string is generated:

$\omega = 272722(28272272927222)^n 272722(272733733(373338393373737333)^{n-i+1}3637322)^n$

In `/home/faculty/terescoj/shared/cs432/hw06` you will find a C++ program that simulates the run-time behavior of this program segment when a working set memory management policy is used. The program prints values:

$$\begin{aligned} \Delta &= \text{window size} \\ P(\Delta) &= \text{total number of page faults} \\ W(\Delta) &= \text{average working set size} \\ F(\Delta) = \frac{P(\Delta)}{|\omega|} &= \text{average page fault rate} \end{aligned}$$

The given main program takes the value of Δ as a command-line parameter. This allows you to write a script (in your favorite scripting language) that runs the program repeatedly for the values of Δ required. The value of Δ is specified with the `-d` flag. A debugging mode is turned on by `-D`. The program also takes a flag `-n` to specify n in the reference string used. The default is 10, and you may use that to generate your plots. You are encouraged to try other values of n, but you need only plot for $n = 10$.

Note that as each entry in the reference string (page) is processed, one of four things will happen to the working set. (*i*) the page is added to the set, and none is removed, (*ii*) the page is added to the set and one old page is removed, (*iii*) the page is already in the set and another page is removed, or (*iv*) the page is already in the set and no other page is removed.

(a) Use this program to plot the following curves: Δ vs. $P(\Delta)$, Δ vs. $W(\Delta)$, Δ vs. $1/F(\Delta)$, for Δ ranging from 1 to 200. From the plot of Δ vs. $1/F(\Delta)$, explain the cause of all knees in the graph in terms of program (or reference string) structure.

(b) Is the strategy used by this program one that could be used by a real system to keep track of a process' working set? Why or why not?