ITP 30002 Operating System, Spring 2020

Final Exam

9 problems in 80 minutes

Problem 1 (10 points)

Describe one or more specific situations where an entry in translation lookaside buffer needs to be invalidated.

Problem 2 (12 points)

Discuss possible drawbacks of increasing concurrency in the performance of a computer system.

(i.e., possible issues that an operating system needs to manage at increasing concurrency)

Problem 3 (10 points)

Write two operations of a counting semaphore, `signal()` and `wait()` using an atomic variable (as the counter) with `compare_and_swap()`:

- `void signal(atomic_int * v) ;`
- `void wait(atomic_int *v) ;`

(Hint: see Section 6.4.3)

Problem 4 (10 points)

Machine learning programs that run neural network training have difference characteristics than traditional user-level programs. These programs tend to load large-size matrices at once from files at an early point in an execution, and then conduct long-running computation on the matrices, which consist of repeated column-wise or row-wise readings/writings of the matrices.

Suppose that you are asked to customize a conventional operating system such that the computer system performs betters with these neural network training programs.

How would you change the operating system?

Problem 5 (10 points)

Answer to each of the following two questions on the Amdahl's law.

a) Calculate the maximum speed up gain of an application that has 50 percents of parallel components for a 4-core processor.

b) Why a concurrent program is not perfectly parallelizable?

Problem 6 (9 points)

Suppose that your computer system has 3 frames each of which has one reference bit, and uses second-chance algorithm for page replacement.

Consider the following page reference string:

```
# 5 2 6 2 1 1 # 3 1 2 4 2 3 # 5 4 2 1
```

The sharp (#) symbols stands for a reference bit clearing event.

How many times page replacement would occur for the given page reference string?

Problem 7 (12 points)

Consider a computer system with a 32-bit logical address and 64-KB page size. The system supports up to 1 GB of physical memory.

Answer to the following two questions:

a) How many pages are there to store a single-level page table?
(explain the logic that you used to get this number)

b) How many entries exist in an inverted page table?
(explain the logic that you used to get this number)

Problem 8 (9 points)

Suppose that Eraser monitored an execution of the following multithreaded program.

```
int x, y, z ; // global variables
mutex a, b, c, d, e ;

main () {
 x = 0 ; y = 0 ;
 t1 = thread_start(fun1) ;
 z = 0 ;
 t2 = thread_start(fun2) ;
 thread_join(t1) ;
 thread_join(t2) ;
}
```

```
fun1 () {
  lock(a) ;
  lock(c) ;
  y = x ;
  unlock(c) ;
  x = 1 ;
  unlock(a) ;
}
```

```
fun2 () {
  lock(c) ;
  z = y ;
  unlock(c) ;
  lock(a) ;
  lock(b)
  x = z ;
  unlock(b) ;
  unlock(a) ;
}
```

Give (1) the lockset of each global variable at the end of an execution, and (2) data race detection results by Eraser.

Problem 9 (18 points)

```
#include <stdio.h>
#include <stdlib.h>

int main () {
  char * s = malloc(100) ;
  scanf(“%s”, s) ;
  printf(“%s”, s) ;
}
```

<hello.c>

```
$ gcc hello.c
$ ls
a.out   hello.c   input.txt
$ ./a.out < input.txt
```

<bash>

Suppose that you had built `hello.c` as `a.out`, and now you are commanding bash to execute `a.out < input.txt`.

Explain all possible system calls and traps that the operating system services for handling this command (i.e., `a.out < input.txt`).