

CIS 505 - Spring 2006
Midterm

Mar 15, 2006

There are ten problems and each problem is worth 10 points. The maximum score you can get for this exam is **80**, roughly one point for one minute. Please pick any **eight** of them. If you answer more than eight problems, we will grade any eight of them for grading unless you state explicitly which ones should be graded.

Please write your answers **legibly** so that we can understand your answers. If a problem seems ambiguous or impossible, please feel free to state your assumptions explicitly and solve the problem. Obviously, your assumptions should be reasonable and should not trivialize the problem.

Please sign the **pledge** at the back of the answerbook. **Good luck!**

1. (10 points)
 - (a) (4 points)

Which of the following two is a system call and explain why?

 - `sbrk(size)`: Increase the size of heap space of the calling process by 'size'
 - `malloc(size)`: Allocate 'size' bytes of memory space from the heap
 - (b) (6 points)

Suppose a process P is blocked waiting for a signal inside monitor M . Let a process Q send a signal to unblock the waiting process P . Identify two possible semantics to decide when Q should be resumed. For each of them, also describe its advantages over the other one.
2. (10 points)
 - (a) (4 points)

What is the priority inversion problem? Also, describe using an example.
 - (b) (6 points)

Propose a way (i.e., an algorithm) to prevent the priority inversion problem. Show how it works using the example you provided for the last problem.
3. (10 points)
 - (a) (4 points)

Describe how RPC (remote procedure call) and conventional procedure call differ in each of the following two aspects: compilation and call-semantics.
 - (b) (6 points)

Describe the sequence of steps taken for a remote process call at run-time.
4. (10 points)

A global state, GS, of a system is a collection of the local states of its sites. That is, $GS = \{LS_1, LS_2, \dots, LS_n\}$ where n is the number of sites in the system and LS_i is a local state of site i . Define what it means to say that a global state is inconsistent. Give an example of such a state of a system consisting of three processes. Can this happen in a real distributed system? Justify your answer.
5. (10 points)

This problem is to compare reading a file using a single-threaded file server and a multi-threaded file server. Suppose it takes 25 msec to get a request for work, dispatch it, and do the rest of the necessary processing, when the required data is in a cache in main memory. If a disk operation is needed, as is the case one-third of the time, an additional 75 msec is required, during which time the thread sleeps. How many requests/sec can the server handle if it is single threaded? If it is multi-threaded? Consider two cases for the multi-threaded server: one using user-level threads and another using kernel-level threads. (Note that you are to provide three answers: single-threaded server, multi-threaded server with user-level threads, and multi-threaded server with kernel-level threads.)

6. (10 points)

(a) (5 points)

Consider the following url: `www.cis.upenn.edu/cis505/`. Suppose DNS is implemented using *iterative name resolution*. Show the sequence of messages that are exchanged between various name servers to resolve the above url, starting from the client's name resolver.

(b) (5 points)

Describe two important advantages of recursive name resolution over iterative name resolution.

7. (10 points)

(a) (4 points)

Describe Lamport's logical clock that preserves *happen-before* relations. Explain why it is not a total ordering. Explain how it can be extended to a total ordering.

(b) (6 points)

Show with an example that, with Lamport's logical clock, if $LC(e_1) < LC(e_2)$, it is not necessarily true that e_1 happened before e_2 . Vector clocks are developed to fix this problem. Explain how vector clocks work.

8. (10 points)

(a) (3 points)

Prove or disprove that

$$P \parallel RUN_{\alpha P} = P$$

(b) (7 points)

One solution to enforce mutual exclusion in a distributed system is to use a centralized coordinator. Using CSP, describe the coordinator and client processes. You may assume that there are only five processes, which are known to the coordinator. Also, state the correctness property and show that your solution satisfies it.

9. (10 points)

(a) (2 points)

Prove or disprove that

$$STOP_{\emptyset} = RUN_{\emptyset}$$

(b) (4 points)

$$\alpha P = \{a, b\}$$

$$\alpha Q = \{b, c\}$$

$$\alpha R = \{a, c\}$$

$$P = a \rightarrow b \rightarrow P \mid b \rightarrow a \rightarrow P$$

$$Q = b \rightarrow c \rightarrow Q \mid c \rightarrow b \rightarrow Q$$

$$R = a \rightarrow c \rightarrow R \mid c \rightarrow a \rightarrow R$$

Define the processes $((P \parallel Q) \parallel R)$, $(P \parallel (Q \parallel R))$ recursively using processes P, Q and R .

(c) (4 points)

Prove or disprove the following statement:

$((P \parallel Q) \parallel R) = (P \parallel (Q \parallel R))$ for every P, Q, R .

10. (10 points)

Design a distributed termination algorithm for a ring of N processes and show that the algorithm is correct.

(a) There are N processes, $P[i]$, for $1 \leq i \leq N$.

(b) A process $P[i]$ can send messages only to $P[(i \bmod N) + 1]$.

(c) All messages sent are delivered in FIFO order.

(d) At any point in time, each process $P[i]$ is **active** or **idle**.

(e) Initially, every process is **active**.

(f) A process becomes **idle** if it has completed (i.e., finished its execution) or is delayed at a **receive** statement.

(g) After receiving a message, an **idle** process becomes active.

We say that a distributed computation has terminated if every process is idle and no messages are in transit.

Show that when your algorithm reports termination, every process is idle and no messages are in transit.