Name: (some people got this wrong) **User Id: (many people got this wrong)**

CMPSCI 377: Operating Systems

Exam 2: Synchronization, Deadlock, Interprocess Communication, File Systems, and Memory Management

November 13, 2002

Answer pieces in *italics* were not required to receive full points.

---

1. **Networking/Interprocess Communication** (25 pts)

   (a) (5 pts) True or False (and explain): In a packet switching network, resources must be be reserved for the duration of the session.

   False. In the packet switching network, resources are not reserved, but are instead used on demand. *(This allows the available network resources to be shared on a very fine time scale).*

   (b) (5 pts) True or False (and explain): The maximum number of hops that a message can incur in a bidirectional ring network with n nodes is n.

   False. The maximum number of hops is $\lfloor n/2 \rfloor$.

   (c) (5 pts) True or False (and explain): Ethernet is commonly used to connect machines on a wide area network. (Hint: you can answer this by only considering the properties of each).

   False. Ethernet is used to connect machines on a Local Area Network. *(A Wide Area Network would perform poorly if the nodes had to share the available network bandwidth.)*

(d) (10 pts) Give one example application for each of the following in which you would prefer to use:

1) data migration: Editing files located on a remote server. Also SETI is an example of data migration.

2) computation migration: Database query. *(especially important if the database is extremely large)*

3) job migration: A large numerical computation that must be performed on a special-purpose machine.

2. **Memory Management** (20 pts)

(a) (10 pts) Explain the difference between logical and physical addresses. Why are logical addresses important?

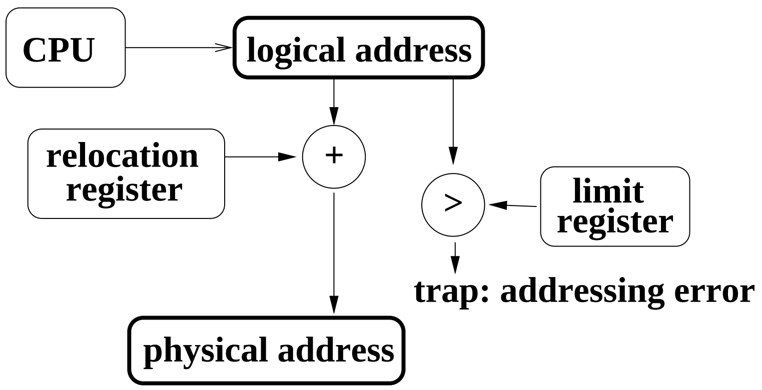
Logical/virtual addresses are with respect to the process, and range from "0" to the size of the memory that is used by the process.

Physical addresses are used to address bytes/words of the memory unit.

Logical addresses are important because they enable us to easily bring multiple processes into memory at the same time without the need for recompilation/relinking. *(But they require some form of translation to physical addresses).*

Also reasonable for the 2nd part: logical addresses allow us to locate a process in different parts of physical memory.

(b) (10 pts) Show a hardware design for a dynamic memory relocation scheme. What is the responsibility of the OS in managing this hardware?



Every time a context switch is made, the OS must reload the relocation and the limit registers with values appropriate for the new process.

3. **File Systems** (20 pts)

(a) (5 pts) Ignoring CPU overhead, what are the three primary components of disk access time (the time from the request of a sector to the time that it is available to the CPU). Given a transfer of a single sector, rank them according to their relative average cost.

Seek time $>$ rotational delay $>>$ transfer time of a single sector.

(b) (7 pts) Define external and internal fragmentation (with respect to file systems).

External fragmentation: occurs under file systems in which files are allocated as contiguous sets of blocks: small, unused holes accumulate within the file system that are not large enough to fit long files (even though collectively there may be enough free space).

Another "ok" answer: external fragmentation is the scattering of a file across very different portions of the disk. *Access time for large files will be larger than under a contiguous allocation method.*

Internal fragmentation: space within a block that is unused for the file data.

(c) (8 pts) List one advantage and one disadvantage of using a large block size.

Advantage: fewer blocks must be read for a given file, which, in cases of non-contiguous allocation of files, means a reduction of seek and rotational delays on average.

Advantage: page tables are smaller.

Disadvantage: leads to more space lost to internal fragmentation.

4. **Deadlock** (25 pts)

(a) (8 pts) What are the 4 conditions for deadlock (include a **short** definition of each)?

i. **Mutual Exclusion** - at least one thread must hold a resource in a non-sharable mode.

ii. **Hold and Wait** - at least one thread holds a resource and is waiting for other resource(s) to become available.

iii. **No Preemption** - A thread can only release a resource voluntarily; another thread or the OS cannot force the thread to release the resource.

iv. **Circular wait** - A set of waiting threads $\{t_1, \ldots, t_n\}$ where t_i is waiting on t_{i+1} $(i = 1\ to\ n)$ and t_n is waiting on t_1.

Consider the following system snapshot using the data structures in the Banker's algorithm, with resources A, B, C, and D, and processes P0 to P4:

	Max				Allocation				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	6	0	1	2	4	0	0	1								
P1	1	7	5	0	1	1	0	0								
P2	2	3	5	6	1	2	5	4								
P3	1	6	5	3	0	6	3	3								
P4	1	6	5	6	0	2	1	2								
													3	2	1	1

Using Banker's algorithm answer the following questions.

(b) (3 pts) How many resources of type A, B, C, and D are there?

{A, B, C, D} = 9, 13, 10, 11

(c) (3 pts) What are the contents of the Need matrix?

	Need			
	A	B	C	D
P0	2	0	1	1
P1	0	6	5	0
P2	1	1	0	2
P3	1	0	2	0
P4	1	4	4	4

(d) (3 pts) Is the system in a safe state? Why?

Yes, can satisfy system in this order: P0, P2, P3, P4, P1 (this is not the only order). The available resources as these processes complete are as follows:

	Available			
	A	B	C	D
before	3	2	1	1
after P0	7	2	1	2
after P2	8	4	6	6
after P3	8	10	9	9
after P4	8	12	10	11
after P1	9	13	10	11

Note that the last line matches the numbers of part b.

(e) (8 pts) If a request from process P4 arrives for additional resources of (1,2,0,0), can the Banker's algorithm grant the request immediately? Show the new system state and other criteria.

This state is not safe. P0 can be satisfied, but the available resources at that point (6 0 1 2) cannot satisfy the needs of the remaining processes.

Here is the full state description:

	Max				Allocation				Need				Available			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P0	6	0	1	2	4	0	0	1	2	0	1	1				
P1	1	7	5	0	1	1	0	0	0	6	5	0				
P2	2	3	5	6	1	2	5	4	1	1	0	2				
P3	1	6	5	3	0	6	3	3	1	0	2	0				
P4	1	6	5	6	1	4	1	2	0	2	4	4	2	0	1	1

5. **Synchronization** (10 pts)

What are the semantic differences (the differences in meaning) between the semaphore wait/signal operations and the monitor condition variables? List two.

When a process issues a monitor wait, the process releases the monitor lock, and other processes can enter the monitor. In code protected by a semaphore wait, only one process (or a counted number of processes) can enter the code; all the other processes must wait until the process with the semaphore releases it with a signal.

There is also no memory of the monitor notify if there are no waiting processes, whereas there is a memory for semaphore signals. Another way to say this is that the monitor signal/wait operations are not commutative, whereas the semaphore operations are.

Note 1: Several people talked about monitors enforcing the lock/unlock sequence whereas semaphores do not. This is correct and this is a difference between semaphores and monitors, but it is not technically a difference in the semantics between the semaphore and monitor signal/wait operations.

Note 2: Several people talked about "memory of" which process was waiting on a semaphore vs which process was waiting on a condition variable (and knowing which process is "next in line"). Depending upon the specific implementation, there may be some differences here, but these would be implementation differences, not semantic differences.