POLITECNICO DI TORINO

System and device programming

OS internals Exam - 20/06/2022 (Cabodi) - standard version - for ON-SITE (in CLASSROM) EXAM

	BEATRICE PIRAS 304812
Iniziato	lunedì, 20 giugno 2022, 11:00
Terminato	lunedì, 20 giugno 2022, 12:20
Tempo impiegato	1 ora 20 min.
Valutazione	**10,75** su un massimo di 15,00 (**72**%)

Domanda 1

Completo

Punteggio ottenuto 2,00 su 3,00

WHEN RESULTS ARE NUMBERS, RELEVANT INTERMEDIATE STEPS (OR FORMULAS) ARE NEEDED
ALL YES/NO ANSWERS MUST BE EXPLAINED/MOTIVATED

Consider a virtual memory system based on paging, with a Byte addressable RAM. The system has a TLB (Translation Look-aside Buffer). A two level page-table is used, based on splitting a 64-bit logical address (from MSB to LSB) into 3 parts: *p1*, *p2* and *d:* p2 uses14 bits, and d uses 13 bits. No other data structures (such as hash tables or inverted page tables) are used. Virtual memory is managed with demand paging.

Answer the following questions:

A) Assuming the RAM memory has an access time T_{RAM} = 75 ns, calculate the TLB miss ratio, assuming that EAT_{PT} >= 100 ns (so this is known), and that the TLB lookup time is negligible. Is the result a lower bound or an upper bound for the TLB miss ratio?

B) Now consider the page fault frequency p_{PF}. An experimental evaluation shows that a page fault is served in 2 ms on average, and that the performance decrease (increase in running time) due to page faults, is between 10% and 30%.
Compute the range of values for p_{PF} that are compatible with the experimental evaluation. (assume that $EAT_{PT} = 100\ ns$)

C) Another experimental evaluation/simulation has been done using two reference strings w_1, w_2, having length, respectively, len(w_1)=10^6, len(w_2)=$2*10^6$, and probability p_1 and p_2. The simulation generated 400 page faults overall (100 with w_1 and 300 with w_2) and estimated an empirical probability f = 0.00012 (expected frequency) for a page fault occurring in the real system. Compute the values of p_1 and p_2

A) TLB miss ratio = ...
upper or lower bound?

TLB miss ratio = (1-htlb)
with 2 level page table in case of page fault we have 3 times the access time, for 2 levels + access to physical addr
EATpt >=h*Tram+(1-h)*3*Tram -> htlb= 0.835
TLB miss ratio = 0.165 -> 16.5%

Upper or lower bound ?
Upper bound, if EATpt grows the TLB miss ratio grows

B) Range of values for p_{PF}

pmin < p_{PF} < pmax

0.1<Tpf*Ppf/EATpf < 0.3
=> 0.1*100ns/2ms < Ppf < 0.3*100ns/2ms
p_{min} = 0.05*10^(-2)
p_{Max} = 0.15*10^(-2)

C) Calculation of string probabilities p_1 and p_2

f= (pfA/len(wA)) * pA + (pfB/len(wB))*pB
12*10^-6 = (100/10^6)* pA + (300/2*10^6)*pB
pA= 1-pB

//calculations came out looking wrong, maybe i should have used 100/400 for pfA and 300/400 for pfB but i have no time to implement it
p_1 = 2.76

p_2 =

A) TLB miss ratio = ...
upper or lower bound?

T_{RAM} = 75 ns (RAM access time)
EAT_{PT} = 100 ns
x = $miss_{TLB} = 1 - h_{TLB}$
2 level (hierarchical) PT => 2 reads for PT lookup

$EAT_{PT} = h_{TLB}*T_{RAM} + (1-h_{TLB})*3T_{RAM} = (1 + 2*(1-h_{TLB}))T_{RAM} = (1 + 2x)\ T_{RAM}$
$2x = EAT_{PT}/T_{RAM} - 1 = 1/3$
x = 1/6 = 0.17
Lower bound as a lower miss ratio decreases EAT_{PT}

B) Range of values for p_{PF}

T_{PF} = 2 ms (PF service time)

$p_{PF}*T_{PF}$ = increase in time due to page faults
$0.1 < p_{PF}*T_P / EAT_{PT} < 0.3$
$0.1 < p_{PF}*(4/200)*10^6 < 0.3$
$0.1 < 2*10^4 p_{PF} < 0.3$
$5*10^{-6} < p_{PF} < 3/2*10^{-5}$
$p_{min} = 5*10^{-6}$
$p_{Max} = 1.5*10^{-5}$

C) Calculation of string probabilities p_1 and p_2

F_1=100, F2=300 ($F_1+F_2 = 400$)
p1 + p2 = 1 (we just have two strings)
$f = p_1*F_1/len(w_1) + p_2*F_2/len(w_2)$

$1.2*10^{-4} = p_1*100/10^6 + p_2*300/(2*10^6)$
$p_1 = 1 - p_2$

$(1-p_2)*10^{-4} + 1.5p_2*10^{-4} = 1.2*10^{-4}$
$1+0.5*p_2 = 1.2$
$p_2 = 0.4$
$p_1 = 0.6$

Commento:
A) ok but lower bound -> 0,75

B) 100ns/2ms = 10^-4 ! -> 0,75

C) equation is right. Calculations wrong -> 0,5

Domanda 2

Completo

Punteggio ottenuto 2,25 su 3,00

ALL YES/NO ANSWERS MUST BE EXPLAINED/MOTIVATED. WHEN RESULTS ARE NUMBERS, THE FINAL RESULT, AND RELEVANT INTERMEDIATE STEPS (OR FORMULAS) ARE NEEDED

Consider a Unix-like file system, based on inodes, with 13 pointers/indexes (10 direct, 1 single indirect, 1 double indirect and 1 triple indirect). Pointers/indexes have size 32 bits, and disk blocks have size 2KB. The file system resides on a disk partition of size 800GB, ***that includes both data blocks and index blocks***.

A) Assuming that all metadata (except index blocks) have negligible size, compute the maximum number of files that the file system can host, using double indirect indexing (N2), and using triple indirect indexing (N3).

B) Given a binary file of size 1024 0.5KB, compute exactly how many index blocks and data blocks the file is using.

C) Consider the same file of Q B, where the lseek(fd,offset,SEEK_END) operation is called to position the file offset for the subsequent read/write operation (SEEK_END refers to the end of the file). Suppose fd is the file descriptor associated to the file (already open), that offset = 0x00800000. Compute the logical block number (relative to the file blocks, numbered starting at 0) at which the position is moved. If the file uses a single (or double/triple) indirect indexing, compute which row of the outer index block contains the data block index (or the inner index block index).

A) Assuming that all metadata (except index blocks) have negligible size, compute the maximum number of files that the file system can host, using double indirect indexing (N2), and using triple indirect indexing (N3).

index size = 4B
n Indexes in one block = 2KB/4B =512
Tot blocks= 800GB/2KB = 400 M blocks
Nmin(N2)= (10+512+1) = 523 // min indexes needed for smallest file in double ind indexing
Nmin(N3) = (10+512+512^2+1) = 262667 // same as before but for triple

N2 = 400M/523 = 783 K files

N3 = 400M/262667 = 1.5 K files

B) Given a binary file of size 1024 0.5KB, compute exactly how many index blocks and data blocks the file is using.

Data blocks: Fsize/2KB = 5121 blocks

Int. frag: 0.75 block

Index blocks: It is in double indexing
10 direct access indexes, 512 from single indirect (1 index)
5121 - 513 = 4608 blocks indexed with double indirect => 4608/512 = 9 index blocks
=> total of 9+10+1= 20 index blocks

C) Consider the same file of Q B, where the lseek(fd,offset,SEEK_END) operation is called to position the file offset for the subsequent read/write operation (SEEK_END refers to the end of the file). Suppose fd is the file descriptor associated to the file (already open), that offset = 0x00800000. Compute the logical block number (relative to the file blocks, numbered starting at 0) at which the position is moved. If the file uses a single (or double/triple) indirect indexing, compute which row of the outer index block contains the data block index (or the inner index block index).

Data block index: 8M / 2KB = 4 K

Outer(single) index: it is in double index and at 6 th row => (4K-513) /512

A) Assuming that all metadata (except index blocks) have negligible size, compute the maximum number of files that the file system can host, using double indirect indexing (N2), and using triple indirect indexing (N3).

General observations

An index block contains 2KB/4B = 512 pointers/indexes.

The partition contains 800GB/2KB = 400 M blocks

Computing maximum numbers N2/N3 *(for files with double*/triple indirect indexing)

In order to compute the maximum number of files, we need to consider the minimum occupancy. We need to consider both data blocks and index blocks.

Let's use MIN_2 and MIN_3 for minimum occupation of the two kinds of file:

MIN_2 = (10 + 512 + 1) data blocks +
1 (sigle) + 2 (double) index blocks = 526

MIN_3 = $(10 + 512 + 512^2 + 1)$ data blocks +
1 (single) + 1 + 512 (double) + 1 + 1 + 1 (triple) index blocks
$= 16 + 1024 + 512^2 = 1040 + 512^2$

N2 = floor(400M / 526) = 797397 = 0,76 M

N3 = floor(400M / $(1040 + 512^2)$) = 1594

B) Given a binary file of size 10240 .5KB, compute exactly how many index blocks and data blocks the file is using.

Data blocks: ceil(10240.5KB/2KB) = 5121
Int. frag = 1-0.25 blocks = 0.75 blocks = (1024+512)B = 1536B
Index blocks
Single index: 1
Inner index blocks (double): ceil((5121-10-512)/512) = 9
Outer index block (double): 1 (double indirect is enough)
Total index blocks: 1+9+1 = 11

C) Consider the same file of Q B, where the lseek(fd,offset,SEEK_END) operation is called to position the file offset for the subsequent read/write operation (SEEK_END refers to the end of the file). Suppose fd is the file descriptor associated to the file (already open), that offset = 0x00800000. Compute the logical block number (relative to the file blocks, numbered starting at 0) at which the position is moved. If the file uses a single (or double/triple) indirect indexing, compute which row of the outer index block contains the data block index (or the inner index block index).

As the new offset is ADDED to the end-of-file, this could be correct or wrong (so producing error) dependng on the opening mode of the file. As the description provided was partial, I'm considering correct also the solution SUBTRACTING the offset

A) ADDING THE OFFSET (and no error)

The new offset is 10240.5K+0x00800000 = 0x00A00200+0x00800000 = 0x01200200
Data block index: 0x01200200/2K = (18M+512)/2K = 9K = 0x2400
10 data blocks are direct
512 data blocks are at single indirect
The block is at the double indirect level
Outer index = (9K-512-10)/512 = 16

B) SUBTRACTING THE OFFSET (this is non the right behavior, but it is considered correct here)

The new offset is 10240.5K-0x00800000 = 0x00A00200-0x00800000 = 0x00200200

Data block index: 0x00200200/2K = (2M+512)/2K = 1K = 0x400
10 data blocks are direct
512 data blocks are at single indirect
The block is at the double indirect level
Outer index = (1K-512-10)/512 = 0

Commento:
A) ok -> 1

B) ok but wrongly considers 10 direct indexes as index blocks -> 0,75

C) start from begin of file instead of end -> 0,5

Domanda 3

Completo

Punteggio ottenuto 1,25 su 3,00

ALL YES/NO ANSWERS MUST BE EXPLAINED/MOTIVATED. WHEN RESULTS ARE NUMBERS, THE FINAL RESULT, AND RELEVANT INTERMEDIATE STEPS (OR FORMULAS) ARE NEEDED

Answer the following questions on memory management:

A) Consider dynamic loading and dynamic linking. Is it possible to dynamically load a program

without requiring dynamic linking? Does dynamic linking require a program to also be dynamically loaded (dynamic loading)?

B) Briefly explain why an Inverted Page Table needs a HASH table. Why is the solution of IPT + HASH table different from a solution with PT based only on Hash table?

C) Consider a CPU equipped with a TLB, can the TLB contain entries of multiple processes or is it constrained to contain entries for just one process?

A) Is it possible to dynamically load a program without requiring dynamic linking?

Yes it is possible , dynamical linking would be useful for dynamic loading but not necessary

B) Briefly explain why an Inverted Page Table needs a HASH table. Why is the solution of IPT + HASH table different from a solution with PT based only on Hash table?

IPT directly links to physical memory addresses and it needs an hash table to map the enormous number of addresses into a smaller and easily storable table, it would be too memory consuming to directly map all addresses into an address table.

C) Consider a CPU equipped with a TLB, can the TLB contain entries of multiple processes or is it constrained to contain entries for just one process?

It is not constrained to only contain one process, it depends on how the system is set it can also contain multiple processes,for example using threads it can be set that when the process is split in parent and child, and the TLB is still the same for both processes until one of them writes

A) Is it possible to dynamically load a program without requiring dynamic linking?

Yes. Although dynamic linking can be combined with dynamic loading, it is not mandatory: a program can be statically linked and dynamically / incrementally loaded, eg. program-based dynamic loading.

Note: dynamic load means that pieces of a program are loaded into memory only if / when needed, dynamic link means that references between modules are resolved at runtime (especially useful for shared libraries).

B) Briefly explain why an Inverted Page Table needs a HASH table. Why is the solution of IPT + HASH table different from a solution with PT based only on Hash table?

Because if you didn't use a HASH table you would need a linear search of the logical page number (p) in the IPT.
The two solutions differ because, despite being very similar in terms of performance of the HASH table, in one case (with the IPT, the IPT entries are inserted directly into the chaining lists of the HASH table, while in the case of simple HASH the classic allocations and deallocations of elements are required at each insertion / cancellation from the HASH. So IPT+HASH solution is more efficient in the use of memory.

C) Consider a CPU equipped with a TLB, can the TLB contain entries of multiple processes or is it constrained to contain entries for just one process?

Both types of TLBs exist: the ones containing entries for multiple processes and the one containing just entries for the currently active process. The latter ones need a TLB cleanup/reset at each context switch

Commento:

A) Explanation/motivation is to concise. Some more details needed -> 0,75

B) It seems this is just a problem of mamory, whereas HAS is essentially trying to speed-up time -> 0,25

C) Mixing/confusing different problems. Parent/child has nothing to do with this question. TLB is a Hardwer component of the CPU. No mention to ASID/PID -> 0,25

Domanda 4

Completo

Punteggio ottenuto 2,25 su 3,00

ALL YES/NO ANSWERS MUST BE EXPLAINED/MOTIVATED. WHEN RESULTS ARE NUMBERS, THE FINAL RESULT, AND RELEVANT INTERMEDIATE STEPS (OR FORMULAS) ARE NEEDED

Consider three OS161 kernel threads implementing a data transfer task based on a producer/consumer model (2 producers, multiple consumers). The threads share a C structure of type struct prodCons defined as follows:

```
#define NumP 2

struct prodCons {
  void *data[NumP];
  int size[NumP];
  int busy[NumP];
  int ready[NumP];
  struct lock *pc_lk;
  struct cv *pc_cv;
  … /* other stuff – omitted */
};
```

Producer i (with i = 0 or 1) updates the i-th entry in the data and size arrays (in practice, this is a buffer). Consumers can read either of the two entries (of data/size) indifferently. The lock is used for mutual exclusion.
the ready and busy arrays of flags indicate, if they are 1:
- ready: corresponding data (and size) ready to be read
- busy: data currently being read or written

Before working on the data, a consumer must therefore wait/verify that a proper condition (for reading) on the flags is true (you need to determine which condition): this happens by calling the function:

```
int consumerWait(struct prodCons *pc);
```

Answer the following questions:

A) Can the shared structure be located into the thread stack, or should it be a global variable or other?

B) As the pc_lk and pc_cv fields are pointers, where should the lock_create() and cv_create() be called? In the producer threads, in the consumer thread? Elsewhere?

C) Provide an implementation of the consumerWait function. No producer and/or consumer code is required, only the function, bearing in mind that the function has the purpose of returning the index (0 or 1) of the buffer (data/size) from which to read (having set the corresponding ready flag to 1). The data must not be read (whoever calls the consumerWait function will do it)

A) Can the shared structure be located into the thread stack, or should it be a global variable or other?

Since it should be accesses by both consumer and producer processes, it should be a global variable to be visible by both.

B) As the pc_lk and pc_cv fields are pointers, where should the lock_create() and cv_create() be called? In the producer threads, in the consumer thread? Elsewhere?

With proper synchronization, they could be called in one of the two types of processes (es in producer process and then be accessible from the consumer one), but more easily we could call them in a parent procerss before the cons and producer processes are created, in this way we are sure to have tehem available for both.

C) Provide an implementation of function consumerWait. No producer and/or consumer code is needed, just the function.

```
void consumerWait(struct prodCons *pc, int minCnt, int minSize) {

  lock_acquire( pc->pc_lk );

  while (!pc->Ready[0] && !pc->Ready[1] ) {
    cv_wait(pc->pc_cv, pc->pc_lk );
  }

  lock_release( pc->pc_lk );


  return              ;
}
```

A) Can the shared structure be located into the thread stack, or should it be a global variable or other?

Each thread has its own thread stack, so a shared data cannot reside there. A global variable is the usual option. Other options include dynamic allocation (by kmalloc), if properly handled.

B) As the pc_lk and pc_cv fields are pointers, where should the lock_create() and cv_create() be called? In the producer threads, in the consumer thread? Elsewhere?

It could be done in each of them, provided that they properly synchronize: e.g. one thread does initializations, the other threads wait.
But a more common solution could be that initializations are done by another thread, the parent/master thread, who is creating the producers and the consumer.

C) Provide an implementation of function consumerWait. No producer and/or consumer code is needed, just the function.

```
/*
It is necessary to wait for one of the two entries to be "available" for reading.
An entry is available if ready and not busy: both conditions are needed, one of the two is not enough, otherwise the protocol would not work in the other direction (also the producer must be able to know when an entry is available for writing).
The reading must NOT be carried out in the function as it costs too much to do so in mutual exclusion.
It would be advisable to put set the busy flag before releasing the mutual exclusion
*/
/* error checking/handling instructions are omitted for simplicity */

int consumerWait(struct prodCons *pc) {
  int ret;
  /* lock per  mutua esclusione (anche i produttori accedono usando il lock */
  lock_acquire(pc->pc_lk);
  /* un produttore puo' chiamare cv_signal o cv_broadcast ad ogni modifica
     in ogni caso occorre un ciclo, motivato dalla semantica Mesa */
  while ((pc->busy[0]||!pc->ready[0]) && (pc->busy[1] || (!pc->ready[1]) ) {
    cv_wait(pc->pc_cv, pc->pc_lk);
  }
  if (!pc->busy[0]&&pc->ready[0])
    ret = 0;
  else ret = 1;
  pc->busy[ret] = 1;
  lock_release(pc->pc_lk);
  return ret;
}
```

Commento:

A & B) ok -> 1,5

C) Just considering ready. The busy flags should be considered as well.
And return value is not handled -> 0,75

Domanda 5

Completo

Punteggio ottenuto 3,00 su 3,00

ALL YES/NO ANSWERS MUST BE EXPLAINED/MOTIVATED. WHEN RESULTS ARE NUMBERS, THE FINAL RESULT, AND RELEVANT INTERMEDIATE STEPS (OR FORMULAS) ARE NEEDED

Consider a possible implementation of the file system calls in OS161, based on per-process and system file tables defined as follows

```
/* system open file table */
struct openfile {
  struct vnode *vn;
  off_t offset;
  unsigned int countRef;
};
/* this is a global variable */
struct openfile systemFileTable[SYSTEM_OPEN_MAX];
...
/* user open file table: this a field of struct proc */
struct openfile *fileTable[OPEN_MAX];
```

Answer the following questions (all answers have to be motivated):

A) Why is systemFileTable a global variable, whereas fileTable is a field of struct proc ?

B) Suppose that, given two user processes P1 and P2 (with p1 and p2 pointers to the respective struct proc) the following conditions are true:

- (p1-> fileTable [5] == & systemFileTable [20])

- (p2-> fileTable [8] == & systemFileTable [20])

to which file descriptor (fd) does the displayed table parts correspond? 5, 8, 20? (motivate)

If P1 and P2 only make readings on the above file (s), are the readings independent or is there interaction between the processes?

C) Assume that the support for the SYS_lseek system call has been implemented. The lseek function is described as follows:

off_t lseek(int fd, off_t offset, int whence);

lseek() repositions the file offset of the open file description associated with the file descriptor fd to the argument offset according to the directive whence as follows:

SEEK_SET

The file offset is set to offset bytes.

SEEK_CUR

The file offset is set to its current location plus offset bytes.

SEEK_END

The file offset is set to the size of the file plus offset bytes.

Will a call to lseek made by P1 (see question B) affect the subsequent read made by P2?

1) always?

2) never?

2) only if the offset parameter is SEEK_CUR?

(multiple selections are possible, for each selected option explain/motivate)

A) Why is systemFileTable a global variable, whereas fileTable is a field of struct proc ?

systemFileTable will refer to all active system processes while fileTable witll refet only to the process referred to in the struct proc

B) to which file descriptor (fd) does the displayed table parts correspond? 5, 8, 20? (motivate)

If P1 and P2 only make readings on the above file (s), are the readings independent or is there interaction between the processes?

The displayed table parts belong to 5 and 8 in the specific file tables of the processes, but they both point to the same entry in the system file table, so they are 'saved' differently in the 'personal' file table of the processes but in the general system they both point to the same fd.
They are not completely independent, for which the table parts 5 and 8 belonging to the specific processes still point to the system file table in the same fd, so they can influence each other

C) Will a call to lseek made by P1 (see question B) affect the subsequent read made by P2?

1) always? no not always, it could be that the P2 process uses a parameter that will not be affected by the previous P1 lseek

2) never? No, for similar reason as above, the parameter could be set in P2 lseek so that it is influenced by the P1 previous lseek

2) only if the offset parameter is SEEK_CUR? Yes, If the parameter is Seek_CURR but in the second Process called, in that case the current offset will be the one where the previous process ended up so the P2 lseek will eb influenced by it

(multiple selections are possible, for each selected option explain/motivate)

A) Why is systemFileTable a global variable, whereas fileTable is a field of struct proc ?

systemFileTable a global variable because it is unique and shared by all processes, whereas we have one fileTable for each process, so we can directly allocate it within the struct proc.

B) to which file descriptor (fd) does the displayed table parts correspond? 5, 8, 20? (motivate)
If P1 and P2 only make readings on the above file (s), are the readings independent or is there interaction between the processes?

We must refer to the (per-process) fiuleTable. Therefore

5 for P1
8 for P2

The reads are NOT independent as the same entry is shared in the systemFileTable. This means that P1 and or P2 "continue" to read from the offset left by the previous reading by the other process left (if there was any). Synchronization might need to be managed, but that's another problem.

C) Will a call to lseek made by P1 (see question B) affect the subsequent read made by P2?

1) always? YES because if P1 repositions the offset, P2 shares it (regardless of which position is taken as a reference)

2) never? NO (see previous answer)

2) only if the offset parameter is SEEK_CUR? NO. Because even if the reference to the beginning / end of the file does not depend on what happened before (therefore on the current position of the offset), the result, the new offset, affects P2

(multiple selections are possible, for each selected option explain/motivate)

Commento:
ok

Domanda 6

Completo

Non valutata

Select "Withdraw from exam" if you wish to withdraw (your test won't be evaluated).

Don't answer or select "I want my exam evaluated" if you are completing your test for evaluation.

You'll be able to possibly withdraw later on, after the test is closed, once you see the proposed solution.

- (a) Withdraw from exam (my test won't be evaluated)
- (b) I want my exam evaluated

Risposta errata.

La risposta corretta è: Withdraw from exam (my test won't be evaluated)