Message 64 (1594 chars)
Return-Path: <hector@DB.Stanford.EDU>
Received: from Coke.Stanford.EDU by EE.Stanford.EDU with SMTP (5.67b8/25-eef) id AA04126; Fri, 27 Jan 19
Received: by Coke.Stanford.EDU (5.57/25-DB-eef) id AA27352; Fri, 27 Jan 95 12:06:21 -0800
Message-Id: <9501272006.AA27352@Coke.Stanford.EDU>
To: shankle@ee.Stanford.EDU
Subject: qual question
Date: Fri, 27 Jan 95 12:06:20 -0800
From: hector@DB.Stanford.EDU
X-Mts: smtp

EE Qual Question
Hector Garcia-Molina, 1995

You wish to compare two files stored on separate computers, to see if they are identical.

PART I: How could you compare the files?

POSSIBLE ANSWER:
(1) Read file on one machine, send all bytes across network, do a byte by byte compare
(2) Compute a "check sum" or signature at one machine, send it across network, and compare to signature of other file.
(3) Assuming files were identical initially, keep a log of changes. Then compare logs.
(4) Compress the file before sending across network.

PART II: For strategy (2) above, compute the probability that the test "fails". Assume each file has p bits, signature is s bits.

POSSIBLE ANSWER:
Consider a given file X with signature Sx being compared against a file Y. There are 2**p possible Y files. Of these, 2**p/2**s have the same signature as Sx. Of these, (2**p/2**s) - 1 will cause our test to fail. (The remaining one with Sx signature is identical to X so the test does not fail.) Thus, the probability that our test gives an incorrect answer is [(2**p/2**s) - 1]/(2**p). If p is substantially larger than s, this simplifies to 1/(2**s).

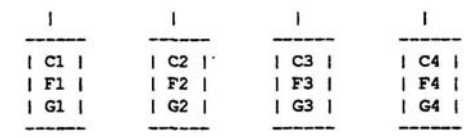
From: Hector Garcia-Molina <hector@DB.Stanford.EDU>
X-Authentication-Warning: Coke.Stanford.EDU: Host localhost [127.0.0.1] didn't use HELO protocol
To: shankle@ee.stanford.edu (Diane Shankle)
Subject: Re: Quals Questions 1996
Date: Sat, 27 Jan 96 09:29:24 -0800
X-Mts: smtp

> Send a copy of your Quals Question & Answer to
> Diane Shankle
Here is my question...
hector

EE QUALIFYING EXAM 1996
Hector Garcia-Molina

You have four computers, C1, C2, C3, C4, connected by a network.
Each computer Ci has a file Fi containing 1000 records.
Each record is 1000 bytes long, including an 8 byte key.
(Example: 1000 byte employee record, with 8 byte employee number as key.)
The keys range from 1 through 1,000,000.
Each computer has an unlimited amount of main memory.

We want to sort the records, so that we end up with four files,
G1, G2, G3, G4. File G1 is located at C1 and will contain the 1000
records with smallest keys. File G2 will be at C2 and will contain the
1000 records with the next 1000 smallest keys, and so on.



(1) Suggest one or more efficient algorithms for sorting the files.

(a) A simple solution is to send all records to one computer, say C1,
and sort them all there. Then the records are distributed to the
appropriate computers.

(b) Another solution is to only send the 8 byte keys to C1.
C1 sorts the keys and determines the three breakpoints, b1, b2, b3.
(That is, the three records with keys less than b1 must go to C1;
the ones with keys between b1 and b2 go to C2, and so on.)
The breakpoints are sent to all computers, which then distribute the records
accordingly.

(c) A third solution is to determine the breakpoints without transmitting
all the keys. For instance, each computer can sort its records locally,
and determine three breakpoints that separate its own records into
four groups. All computers send their local breakpoints to say C1,
and C1 can determine "roughly" where the global b1, b2, b3 breakpoints are.
At this point C1 can either perform more iterations with the other computers
to refine the breakpoints, or it can go ahead and initiate the record
transfers. (If the breakpoints are not known exactly, some records may go to
the wrong computer, but when this is discovered, they can be routed
to the correct place.) Another variation is for computers to initially
send more than three breakpoints, e.g., they send a "histogram"
that more accurately reflects the key ranges it has. With these
histograms, C1 can more accurately determine the true global breakpoints.

(2) What are the important metrics for evaluating the algorithms?
Can you roughly evaluate the performance of the algorithms?

Important metrics are total running time of the algorithm,
number of bytes transmitted, number of messages, processor utilization.
The evaluation of the algorithms depends on the exact variant that was
proposed. During the exam, students we asked to focus only on
the number of bytes transmitted.

X-Sender: hector@db.stanford.edu
Date: Tue, 27 Jul 1999 10:49:49 -0700
To: shankle@ee.Stanford.EDU
From: Hector Garcia-Molina <hector@CS.Stanford.EDU>
Subject: my qual question
Cc: siroker@DB.Stanford.EDU
Mime-Version: 1.0

CS

EE QUALIFYING EXAM 1999
Hector Garcia-Molina

Write (in any programming language, or in pseudo-code) a procedure for binary search. Assume you have an array if integers X[i] where "i" can range between 1 and N. The input to the procedure is an integer value "v" that you are searching for. The output is (1) a flag "found" which is true if "v" was one of the values in X, and (2) if flag is true, the value "k" such that X[k]=v.

Sort the table first

```
typedef _out
{
   int x
   int index;
} out,

int SortSeq ( int * pdwIn , Out * pO );
```

X-Sender: hector@db.stanford.edu (Unverified)
Date: Thu, 10 Feb 2000 14:09:54 -0800
To: Diane Shankle <shankle@ee.stanford.edu>
From: Hector Garcia-Molina <hector@cs.stanford.edu>
Subject: Re: Quals Question

here is my quals question... sorry for the delay...
hector

EE Qual exam Question January 2000
Hector Garcia-Molina

Consider a FIFO queue implemented with
a doubly linked list.

(a) Describe the data structure needed
to implement this queue.

(b) Using simple pseudo-code, write an Enqueue
procedure that adds an element to the queue.

(c) Write a Dequeue operation that returns an
element from the queue.

(d) Discuss what would happen in multiple processes
concurrently used your enqueue and dequeue procedures.
What can be done to prevent such conflicts?
(Just discuss briefly; do not write code.)

X-Sender: hector@db.stanford.edu
Date: Thu, 15 Feb 2001 13:25:44 -0800
To: Diane Shankle <shankle@ee.stanford.edu>
From: Hector Garcia-Molina <hector@cs.stanford.edu>
Subject: Re: Quals Question 2001

At 10:43 AM 2/15/01 -0800, you wrote:
> I am still waiting for you to submit your Quals Question either by hard copy
> or email.
> Please try to submit by 2/23/01.

Here is my question.
hector

Quals 2001 Question
Hector Garcia-Molina

(a) What is a hash table? What is it used for?
How can collisions be handled?

(b) Write pseudo-code to insert a new key into
a hash table. Assume that open addressing is used to
resolve collisions.

The hash table is implemented with an array X
ranging from 0 to N. You are given a hash function h
that you can call; the function return an integer
between 0 and N. Your insert procedure takes
as input a key. It returns a flag that is either:
OK: the value was successfully inserted
DUP: the value was not inserted because it already
exists in the table
FULL: the table was full, no value was inserted.

X-Sender: hector@db.stanford.edu
Date: Fri, 15 Feb 2002 16:44:16 -0800
To: Diane Shankle <shankle@ee.Stanford.EDU>
From: Hector Garcia-Molina <hector@cs.stanford.edu>
Subject: Re: Quals Question 2002

At 04:26 PM 2/15/2002 -0800, you wrote:
> Please turn in your Quals Question 2002. You can send email or a hard copy

Here is my question...
hector

Hector Garcia-Molina
EE Quals Question 2002

The Fibonacci sequence is 1, 1, 2, 3, 5, 8, 13, ...
Each term is computed as the sum of the previous two terms.

(1) Write a function F(I) that computes the Ith number in the
sequence. For example, F(6) should return 8.
First write the function using recursion, and
then using iteration.

(2) Estimate the amount of main memory used by each of the two
function implementations. For your estimate, just consider
the number of variables allocated (1 unit for each), either
on the stack, or in global space.

(3) Estimate the computation costs of each implementation.
Count only the number of additions performed by each.

X-Sender: hector@db.stanford.edu
Date: Tue, 01 Apr 2003 11:28:52 -0800
To: Diane Shankle <shankle@ee.Stanford.EDU>
From: Hector Garcia-Molina <hector@cs.stanford.edu>
Subject: Re: Quals Question 2003

At 09:15 AM 4/1/2003 -0800, you wrote:
> Still waiting for all the Quals Questions to be turned in.!

I may have already sent my question in, but just in case,
here it is again...

hector

Hector Garcia-Molina
EE Quals Question 2003
---

(1) Define briefly what a binary search tree is.

(2) Write a procedure for searching a binary tree.
There is a global variable T that points to the root of the tree.
(If the tree is empty, T is null.)
The procedure takes as a parameter the value to search for,
and returns a pointer. If the value was found in the tree, the
returned pointer identifies the node holding the value.
If not, the returned pointer is null.

(3) Write a procedure for inserting a new value into the same tree.
The input parameter is the value to be inserted.
No value is returned.
Duplicates are allowed in the tree, so a new node is created
even if the value is already in the tree.

(4) If there are N nodes in the tree,
What is the worst-case number of nodes that must be inspected
in a search? What is the average case?

X-Sender: hector@db.stanford.edu
Date: Mon, 26 Apr 2004 10:39:40 -0700
To: Diane Shankle <shankle@ee.Stanford.EDU>
From: Hector Garcia-Molina <hector@cs.stanford.edu>
Subject: Re: Quals Questions 2004 Overdue

At 10:17 AM 4/26/2004, you wrote:

> The Spring Quarter is coming to a close in less than seven weeks!
>
> Send in your Quals Question so I can mark you off my list!

Hector Garcia-Molina
EE Quals Question 2004

Consider two vectors A[1]...A[N] and B[1]...B[N]
stored in two arrays. We want to compute the dot
product defined as
d = A[1]*B[1] + A[2]*B[2] + ... + A[N]*B[N]

(1) Write a statement (pseudo-code) to compute the dot product.

(2) A sparse vector is one which contains very few non-zero values. If N is large, it is not effective to store a sparse vector in array, since a lot of space is wasted storing zeroes. Suggest an alterate representation for a sparse vector, which uses space proportional to the number of non-zero entries (not space proportional to N).

(3) Write pseudo-code to compute the dot product when two vectors are represented using the data structure of Part (2).

Hector Garcia-Molina
EE Quals Question 2005

Consider a STACK data structure. Each record on the stack has 3 fields:

* next: pointer to the next record, further down the stack
* prev: pointer to the previous record, up the stack
* val: an integer value

A global variable TOP points to a dummy record which is always at the very top of the stack. An empty stack has the dummy record only, with both pointers null.
If there is one element, top.next points to it.

(a) Write (pseudo-code) a procedure POP which returns the top value in the stack, and removes the record for that value.

(b) Write a procedure PUSH that adds a value X at the top of the stack. The procedure creates a new record.

(c) Write a procedure that deletes the last record from the stack (the one furthest away from TOP).

Hector Garcia-Molina
EE Quals Question 2007

The buffer manager for a file system keeps track of what
disk blocks have been read into memory.
When a new block is to be read, the manager must
select a buffer to hold the incoming data.
If no free space is available, the manage must first
select a buffer to flush.

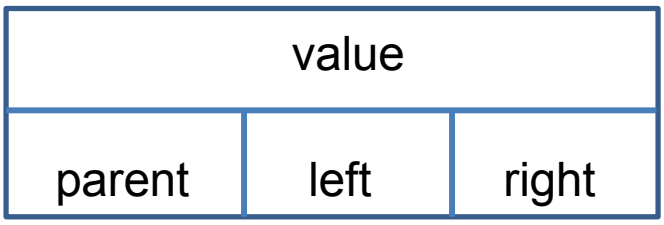
(a) What strategies can the buffer manager
use to select a buffer to flush when space is needed?

(b) Consider a buffer manager that keeps track
of its buffers using a doubly linked list.
Each record in the list represents a buffer
and indicates what disk block is stored in that buffer.
The records are ordered by access time, where
global pointer NEW points to the record that represents
the most recently accessed buffer, and OLD points
to the record that represents the least recently accessed buffer.

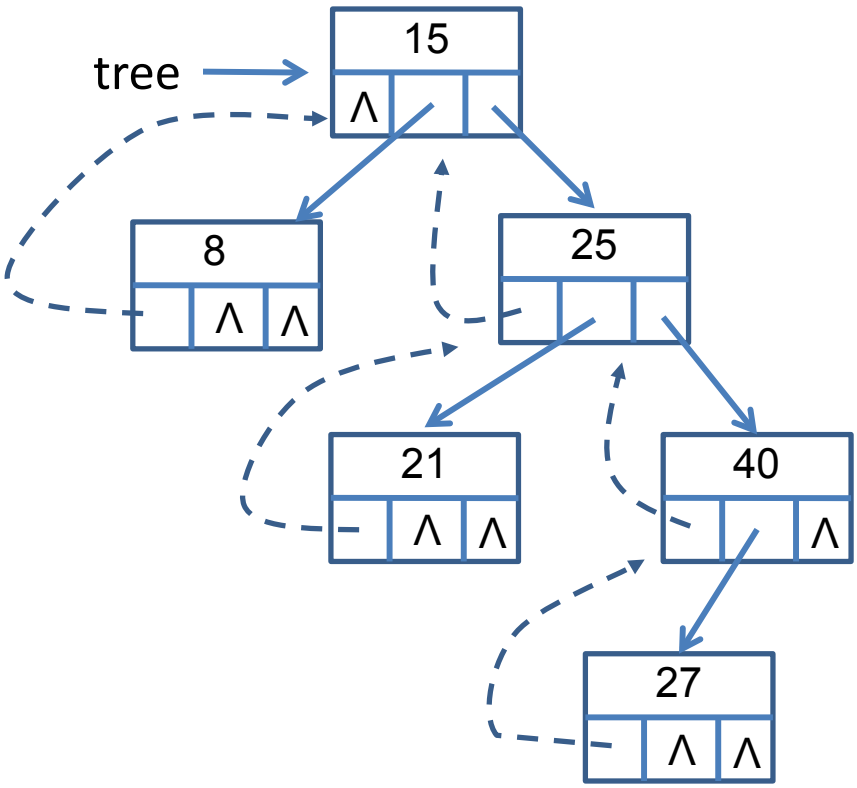
Write pseudo-code for a procedure REGISTER(P) that
registers the fact that the buffer represented by record P
is now being accessed. Thus, record P which is already in the list
need to become the new NEW record at the top of the
doubly-linked list.

Binary Tree

- Node x in tree:

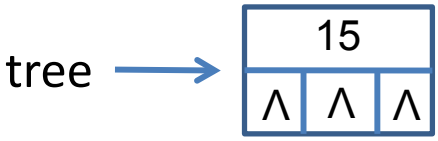


- x.value = search value
  x.parent = pointer to parent node
  x.left = pointer to left node
  x.right = pointer to right node
- Example:



- Empty tree:

- Tree with a single value:



- Question 1
  Write <u>recursive</u> pseudo code for lookup procedure
  lookup(tree, value): returns pointer to node
  null if value not found
- Question 2
  Write pseudo code for delete procedure
  delete(tree, value): removes node with value;
  does nothing if value not in

Binary Search

Prof. Hector Garcia-Molina

- We are given an array A of sorted integers
  Example:

  - N is number of elements (N > 0)
  - A[1] is first element, A[N] is last
  - Variables A, N are global
  - There are no duplicates in array
- Your task: code following function
  Procedure FIND(V): returns FOUND, INDEX
  - V is integer we are looking for
  - If V is in array, FOUND is set to TRUE, else FALSE
  - If FOUND=TRUE, INDEX is location of V
  - Example:
    FIND(13) returns FOUND=TRUE, INDEX=4
    FIND(14) returns FOUND=FALSE
    FIND(87) returns FOUND=FALSE
- Use binary search for your procedure
- Extra credit: handling duplicates in array