搜索算法

1、有1亿个浮点数，请找出其中对大的10000个。提示：假设每个浮点数占4个字节，1亿个浮点数就要站到相当大的空间，因此不能一次将全部读入内存进行排序。

~~~~~~~~~~~~~

其实占用内存不算大, 可以接受. 呵呵.

既然不可以一次读入内存, 那可以这么试试:

方法1: 读出100w个数据, 找出最大的1w个, 如果这100w数据选择够理想, 那么最小的这1w个数据里面最小的为基准, 可以过滤掉1亿数据里面99%的数据, 最后就再一次在剩下的100w(1%)里面找出最大的1w个咯~~

方法2: 分块, 比如100w一个块, 找出最大1w个, 一次下来就剩下100w数据需要找出1w个了.

对于上面提到的找出100w个数据里面最大的1w个, 说起来比较罗嗦, 还是说说找到第1w个大的数字的方法:

用快速排序的方法（只进行一次划分，而不是对所有的排序), 分2堆, 如果大的那堆个数N大于1w个, 继续对大堆快速排序一次分成2堆, 如果大堆个数N小于1w, 就在小的那堆里面快速排序一次, 找第10000-N大的数字; 递归以上过程, 就可以找到第1w大的数. 据说也是STL的search\_n()的方法;

参考上面的找出第1w大数字, 相信楼主就可以类似的方法找出前1w大数字了

2、有一篇英文文章(也就是说每个单词之间由空格分隔)，请找出“csdn”着个单词出现的次数，要求效率最高，并写出算法的时间级。

假设不区分大小写，由于英文字母有26个，因此，可以将单词映射为数字。csdn被映射成:

( 'c '- 'a ')\*32\*32\*32+( 's '- 'a ')\*32\*32+( 'd '- 'a ')\*32+( 'n '- 'a ')

即：( 'c '- 'a ')\*(1 < <15)+( 's '- 'a ')\*(1 < <10)+( 'd '- 'a ')\*(1 < <5)+( 'n '- 'a ')

在搜索过程中发现两个很好的程序设计类面试题库（英文），共享一下，如果你想面试Microsoft，Google或者Goldman Sachs，看看这两个网站上的题目就可以了。

* [techInterview Discussion](http://discuss.techinterview.org/default.asp?interview)
* [CareerCup](http://www.careercup.com/)：网站似乎被G-F-W了，幸好[RSS](http://www.careercup.com/rss.xml)还能用。

XOR 指针

### back and forth linked list traversal

Given a linked list and a start and end pointer pointing to the first and last node of the linked list respectively. Now modify the list such that it can traversed in the forward as well as backward direction starting from start and end pointers respectively. You cannot allocate any extra memory or modify the structure of node that is :

  struct node{

    int I;

    struct node \*next

  };

这个是传说中的“异或指针双向链表”，也就是说：一个结构体只要两个域，一个放数据，一个放前驱后继的异或值（注意：指针和指针异或后依然是指针）；这样，如果当前某个指针ptr指向第i个结构体node[i]：

1) node[i].xor= node[i-1].addr + node[i+1].addr;

2) node[i].addr = ptr;

这里假设xor为异或值，addr为节点实际地址。

3) 另外，node[1].addr, node[n].addr 为已知。

那么:

由于node[1].xor = node[n].addr + node[2].addr，则

    node[n].addr + node[1].xor = node[n].addr + (node[n].addr + node[2].addr) = node[2].addr;

提示一下，上述能够成立是因为我们有：节点n的地址（node[n].addr）；节点1的地址node[1].addr得到的node[1].xor；

再进一步，如果我们知道，node[i-2].addr, node[i-1].addr，则：

    node[i-2].addr + node[i-1].xor = node[i-2].addr + (node[i-2].addr + node[i].addr) = node[i].addr;

差不多就这样了，如果还不明白，那么放弃考研吧。

**微软帽子问题**

N people team up and decide on a strategy for playing a game. Then they walk into a room. On entry to the room, each person is given a hat on which one of the first N natural nos is written. For example, for N=3 the 3 team members may get 2,1,2. Each person can see the numebers written on the other hats, but do not know the number written on his own hat.

Every person then simultaneously guesses the number of his own hat. what strategy can the team follow to make sure that altleast one person on the team guesses his hat correctly.

Answer:

Let a(k) be the hat-number of the k-th person.

Let guess(k) be the guess of the k-th person.

guess(k) =  (-a(1) - a(2) - ... -a(k-1) - a(k+1)-...- a(n) - k) (mod n)

a(k) - guess(k)

= (a(1) + a(2) + ... + a(n) + k) (mod n)

We have n such different numbers

so one of them must be 0,  say m, then

a(m) - guess(m) = 0

### Finding nth number

There is a series of numbers in ascending order. All these numbers have the same number of binary 1s in them. Given the number of 1 bit set in each of the numbers, write an efficient algorithm to find the nth number in the series.

let the no. of zeroes be z and the no. of ones be o and the nth no. has to be found,

//pseudo code

string res;

for(i=0;i<total\_no\_of\_bits;i++)

{

if((z+o)C(z) >= n)  //排列组合中的C, 比如C(5, 3).

{

res+="0";

z--;

}

else

{

res+="1";

n-=(z+o)C(z);

o--;

}

}

这个答案中可能有个逻辑问题，但是用C(m,n)的做法没有错。

比如，如果3个0， 2个1， 那么如果两个0和两个1的组合个数小于要求的第n个，那么第一位应该是1， 同时剩下3个0和1个1， 其中求它的n-C(4,2)个大的元素。如此继续.

### [MS]Which DS is best and why?

Given a file with a lot of words (10 million) find out the top 10% most frequently occuring words.

Using Hashtable can be one of the solutions，建立索引。

另一种就是根据字符进行分组并进行分类，类似基数排序。

建立新文件，文件名从0-n       【

  如果词越多，n可以越大，在1000到10000都可以，具体多少，要在实践中摸索，

  自定义新文件的数据类型

          词       string

          频率     long

  1   每次从文件中读取一行（也就是一个词），

  2   打开指定的新文件是，新文件可以用词的第一个字的UNCODE编码   对N   取模得到     要保存的文件

  3   从新文件中找出这个词的记录

      如果有       词频+1

          循环（如果词频大于前一条   则和前一条交换位置）直到小于前一条为止   以后这就叫置换排序

    如果无     在文件尾部加入一条

  goto   1   直到处理完所有词

  4   打开文件0   读入10条记录保存在内存中

  5   打开文件1

    6     从文件1中读入一条记录，如果这一条记录的词频大于内存中第10条记录时，把内存中的第10条记录换   成这一条     进行   置换排序

    7   读入下条记录，直到读入记录的词频不大于内存中第10条记录为止

  8   处理完N个文件

  内存中的记录就是你要的了

There is a central server and some clients connected to it. All the changes made to data occur at the server, and all the clients have just read access. You have two options:-

1. Push :- The server keeps pushing data to the clients.

2. Pull :- The client keeps requesting the server to send data.

What are the advantages and disadvantages of each type.Design a system which uses both the push as well as pull strategy

It depends on how frequently clients will request the data from the server. In case when it happens too oftnen - the push model is prefferable.

In the opposite case the "pull" model is preferrable, for example for web browsers..

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

Given a Parent -Child binary tree ,build the child -sibling version of it? Minimize the space requirements wherever possible.

createNewTree(TreeNode\* ptr)

{

if(ptr)

{

createNewTree(ptr->left);

if(ptr->left)

ptr->left->right=ptr->right;

createNewTree(ptr->right);

}

}

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

Given two array of numbers find if each of the two arrays have the same set of integers. （没有排序, e.g. 1,2,3,4,5 vs 2, 3, 4, 1, 5)Suggest an also faster than nlogn without extra space. The nos can be repeated.

XOR may be a solution, yet not perfect, as 0, 3 and 1, 2, is the same.

排序也是一种方式，但是需要额外的一个变量。

----------------------------------------------------------------------

[Count Equal Elements in two array](http://www.cnblogs.com/graphics/archive/2009/06/07/1498099.html)

Sort the arrays, then use

 1 public int Count(int[] a, int[] b)

 2 {

 3     int i = 0;

 4     int j = 0;

 5     int c = 0;

 6     while (i < a.Length && j < b.Length)

 7     {

 8         if (a[i] < b[j])

 9             i++;

10         else if (a[i] == b[j])

11         {

12             c++;

13             //i++;

14             //j++;

15         }

16         else

17             j++;

18     }

19     return c;

20 }

------------------------------------------------------------------------------------------

Given two sorted array A1 and A2 of lengths L1 and L2,L1<L2 and the last L1 slots in A2 are vacant. Get me all the numbers in A2 sorted in O(n) without using extra space.

int i = L1 - 1, j = L2 -1;

while (i >= 0) {

  if (A2[j] > A1[i]) {

    A2[L2+i] = A2[j];

    --j;

  }

  else {

    A2[L2+i] = A1[i];

    --i;

  }

}

Start merging into the end of A2. Whenever you merge an element from A2, the distance (number of empty cells) between the end of the original A2 and the beginning of the merge remains the same. Whenever you merge an element from A1, the distance decreases by one, but since there are L1 empty cells initially, the distance never drops below zero (which would imply

that something got overwritten).

------------------------------------------------------------------------------------------

Two persons are playing with dice. The faces of the dice are colored with white or black color.

Player1 wins if same color appears on both dice. Player 2 wins if different colors appear.

5 faces of dice1 is colored with black color and remaining 1 with white. Both guys have equal chances of winning. How dice 2 is colored.

then 5x + y = 18 (as totally there will be 6\*6 = 36 possibilities)

x+y = 6.

-----------------------------------------------------------------------------------------

Let us say we have input:

"swap three and then two words"

and need to swap words "three" and "two" to get output

"swap two and then three words".

Let us say that we have function reverseW(char\* s, int start, int end) that will reverse all the words in place. This can be implemented by first reversing the entire string in place by swapping chars from start and end and meeting in the middle and then repeating the same for each word in the reversed string.

Let us consider only the substring that contains the 2 words to swap as the rest can be ignored:

three and then two

owt neht dna eerht

two then and three

Now reverse the words in between the 2 words we wanted to swap.

then and

dna neht

and then

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

Imagin there is a square matrix with n x n cells. Each cell is either filled with a black pixel or a white pixel. Design an algorithm to find the maximum subsquare such that all four borders are filled with black pixels.

0 0 1 0 1

0 1 1 1 0

1 1 0 1 1

0 1 1 1 0

0 0 1 0 1

RowSum

0 0 1 1 2

0 1 2 3 3

1 2 2 3 4

0 1 2 3 3

0 0 1 1 2

ColSum

0 0 1 01

0 1 2 1 1

1 2 2 2 2

1 3 3 3 2

1 3 4 3 3

For i=1,j=1 and k=2

RowSum[i][j+k]=3 and RowSum[i][j]=1 diff=2==k

RowSum[i+k][j]=1 and RowSum[i][j]=1 diff=0

ColSum[i+k][j+k]=3 and ColSum[i][j+k]=1 diff=2==k

ColSum[i+k][j+k]=3 and ColSum[i+k][j]=3 diff=0

Then there is a square border from (i, j) to (i+k, j+k).

----------------------------------------------------------------------------------------

Given an array of size n. It contains numbers in the range 1 to n. Each number is present at least once except for 2 numbers. Find the missing numbers.

Assume 4, 5, 2,1, 2, 5

Set the value as -1 when for the number in the array[number]

First run,1, 5, 2, -1, 2, 5

second run, -1, 5, 2, -1, 2, 5

third run, -1, 2, 2, -1, -1, 5

......

Till all is done. When one's index is met as -1, then the number is duplicated; if one number's index is positive, then the number is missing.

---------------------------------------------------------------------------------------

You have a stack that is accessed by multiple threads simultaneously and you wish to

synchronize access. You do not want to use locking to implement synchronization. Implement

a thread-safe version of the stack

If you have garbage collection, just allocate a new node each time. Once all threads are done with a node, it is reclaimed.

If you don't, use a version number (e.g., with a 64-bit CAS, a 32-bit pointer and a 32-bit unique ID).  (ID for what, thread ID?) (Check the stack top before return, if the top is allocated by other threads already, if yes, retry another place).

---------------------------------------------------------------------------------------

对于lock-free data structure问题的解决，一般是使用流行的CAS操作。来防止需要读写的区域的数值和预期的数值不一样的情况。

Shared,non-blocking的数据结构，和shared blocking的数据结构相比，采用了比较原始的同步方法。咱讨论最多的一个同步方法，就是CAS（compare and swap）。

       关于CAS操作的伪代码如下：

CAS(A,B,C)

BEGIN ATOMIC

if (A==C) {A=B; return TRUE; }

else { return FALSE; }

END ATOMIC

在使用CAS操作的时候，在计算之前，首先记录下一个变量的值。经常是记录一个shared data structure 实现的pointer。然后把这个变量的值存起来。这个时候，当得到了这个变量的新的值，需要update这个共享变量的数值的时候，就需要采用CAS操作来自动update这个共享变量的值。

CAS操作首先检查shared var的值和已经保存起来的值是不是一样的，如果是一样的，就执行update操作。如果不是一样的，就报错。

如果操作失败了的话，就使用这个shared var的新的值来重新做一次计算操作(通常在一个无限循环中）

另一个变种是DCAS，引入了引用计数。

-----------------------------------------------------------------------------------------

在单项链表或者链式栈中，最后一个元素的空指针是可以利用的。

－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－

iven an array of positive integers( including 0 ).

Each value in the array indicates how many jumps you can take to the right. So, e.g. if array[3] = 2, it means that position 4 & 5 are reachable from position 3.

Find out if the last element is reachable from the front

traverse the array from first element...

at each element calculate 'max' index that can be reached from that node...

Now traverse upto max...At each intermediate node update max if new max is greater than previous

See if this leads to end of array..

time o(n)

space o(1)

Another approach is DFS.

----------------------------------------------------------------------------------------

I recently encountered this problem.

There are two binary trees T1 and T2 which store character data, duplicates allowed.

How can I find whether T2 is a subtree of T1 ? .

T1 has millions of nodes and T2 has hundreds of nodes.

?????

Try building strings based on the relationships and that should work. Using the example above, create a taxonomy of every relationship parent to child and identify it with a string or alphanumerical sequence.

So a parent B with child A and C, then C having a child D could be recreated as:

b with child a = "BA"

b with child c = "BC"

c with child d = "BCD"

a(b(c,d), e(f, nil))， 然后一层一层剥离？

如果是子树的话，简单，采用中序和左序周游，然后比较顺串的持续就可以了。

You might have to get creative how you code that in the both trees using text delimiters for "and", "or", and node "level" of the sequence.

------------------------------------------------------------------------------------------------

100 scientists are standing in a row 100th scientist can see de head of first 99 scientists.The 99th can see de head of 1st 98 and so on..A man puts a hat on de head of each scientist which is black or white.The problem is each scientist should say de color of de hat he s wearing.If he is wrong he will be killed.He can only say "Black" or "White".Whatever he says can be heard by all de scientists infront of him.No one knows de color of de hat he is wearing at the start.The start is made from de 100th scientist.how many can be saved?

第100位老兄坚持猜偶数顶黑帽子。他可能死，但其他人就知道猜什么了

------------------------------------------------------------------------------------------------

### One way flight trip with billions of layovers

The question was the following. I'm rephrasing the question to make it clear for everyone to understand:

- You are going on a one-way flight trip that includes billions of layovers.

- You have 1 ticket for each part of your trip (i.e: if your trip is from city A to city C with a layover in city B, then you will have 1 flight ticket from city A to city B, and 1 flight ticket from city B to city C.

- Each layover is unique. You are not stopping twice in the same city.

- You forgot the original departure city.

- You forgot the final destination city.

- All the tickets you have are randomly sorted.

Question are:

- Design an algorithm to reconstruct your trip with minimum complexity.

- How would you improve your algorithm.

Note: O(n\*n) is trivial solution

Example:

- randomly sorted:

New York->London

San Francisco-> Hong Kong

Paris->New York

London->San Francisco

- sorted:

Paris->New York

New York->London

London->San Francisco

San Francisco-> Hong Kong

just sort the ticket(strings) by starting starting stations .. O(nlgn)

Now look at each ticket from the beginning of  the sorted list. Search for the destination of the ticket under consideration, as the list is sorted u can do it in O(lgn) time.

if destination is not found then this is last station, otherwise proceed.

So in O(nlgn ) time destination is found.  Now just trace back to the starting station.(for this u again sort the tickets by destinations )

So overall O(nlgn)

Hashtable is another choice.

Count the appearance of the cities, if one city only appear once, this is the start or stop. then check the ticket whether this is the start or stop.

-------------------------------------------------------------------------------------------------------------------

there are 3 sorted lists. We have to choose an element from each of the 3 lists such that, distance is minimum where distance is: |a-b| + |b-c| + |a-c|. 'a' being the element from the 1st list, 'b' from the second list and 'c' from the 3rd list.

Need to do it in O(n) time, 'n' being sum-total of length of the 3 lists

there are 3 sorted lists. We have to choose an element from each of the 3 lists such that, distance is minimum where distance is: |a-b| + |b-c| + |a-c|. 'a' being the element from the 1st list, 'b' from the second list and 'c' from the 3rd list.

Need to do it in O(n) time, 'n' being sum-total of length of the 3 lists

So the distance function also can be written as 2 (max {a, b, c} - min {a, b, c}). For any given maximum of a, b, c, then, the other choices are clear...

This can be done as follows.

First take the minimum numbers from the three arrays. Find the distance and keep it as min\_seen so far.

Now take the next element from the array which has the least of the three numbers and again find the distance and compare it the min\_seen so far.

Keep doing this.

The logic behind this is that, if a triplet <x,y,z> optimizes the distance then they are the closest to each other than any other triplet. So if we find the distance of <x,y,z> and assuming 'x' is the least of these, by moving y or z we will get greater distance. Only by moving x, we "may" get a smaller distance.

Let me know if this works.

-------------------------------------------------------------------------------------------------------------------

There are 12 coins with 6 heads and 6 tails up. It is desired to divide the coins in such a way in two groups  such that each group contains the same number of heads (and therefore, the same number of tails) Following are the restrictions :

One can't see the coins.  Assume that it has to be performed while one a  is blind-folded

Flipping of coin operations can be used.

Devise a strategy to achive this .

pick six coins at random. This is group 1.  Flip all coins in group 1 once.  Now group 1 and the the remaining group will both have the same number of heads.    Classic problem...

----------------------------------------------------------------------------------------------------------------------