# 609. Find Duplicate File in System

1. Imagine you are given a real file system; how will you search files? DFS or BFS?

Answer: BFS优先访问同目录下文件，比较快，但是对内存要求比较高；而DFS会频繁在目录之间切换，速度可能会慢，但是对内存要求比较低

DFS for a local file system and BFS for distributed file system (like crawler),

Question-2: If the file content is very large (GB level), how will you modify your solution?

Answer:

Solution 1: Core idea: make use of meta data, like file size before really reading large content:

* DFS to map each size to a set of paths that have that size: Map<Integer, Set>
* For each size, if there are more than 2 files there, compute hashCode of every file by MD5, if any files with the same size have the same hash, then they are identical files: Map<String, Set>, mapping each hash to the Set of filepaths + filenames. This hash ids are very very big, so we use the Java library Big Integer.

To optimize Step-2, In GFS, it stores large file in multiple "chunks" (one chunk is 64KB). we have meta data, including the file size, file name and index of different chunks along with each chunk's checksum (the XOR for the content).

Optimization 1 based on Solution1: For the files with the same size:

Compare each file's checksum (don’t need to do compare)

vs

Split file into chunks, save chunks’ checksum and compare all chunks’ checksum.

Disadvantage: There might be False Positive duplicates, because different content might share the same checksum.

Optimization 2 base on Optimization 1: Compare files byte by byte.

3. If you can only read the file by 1kb each time, how will you modify your solution?

Answer: Just need to modify the MD5 function. We may read all the files and calculate the MD5 first. Now we have to read 1k a time and calculate the MD5;

Question 4: What is the time complexity of your modified solution? What is the most time-consuming part and memory consuming part of it? How to optimize?

Answer: Hashing part is the most time-consuming and memory consuming. Optimize as above mentioned, but also introduce false positive issue.

6. How to make sure the duplicated files you find are not false positive?

Answer: Refer to the optimization of question 2.

找一个图片的从左到右的最佳水平线： 给你一个float[][]， 让你找从左到右的 path 中， 每一步选取最大值走，找出这个最大path中的最小值返回

假设

1 0 2 3

2 1 5 6

3 1 2 0

我觉得这上面是 best path 是 3 1 5 6 返回 1

Given a 2-d array of "sharpness" values. Find a path from the leftmost column to the rightmost column which has the highest minimum sharpness.

Output the highest minimum sharpness. Each move can only move to the top right, right or bottom right grid

Example: 3\*3 matrix

5 7 2

7 5 8

9 1 5

The path with highest minimum sharpness is 7-->7-->8, because 7 is the highest minimum value in all the paths.

Idea: Use DP dp[r][c] = min(max(dp[r-1][c-1], dp[r][c-1], dp[r+1][c-1]), grid[r][c])

followup和之前的面经一样，问的是如果是100million \* 100 million怎么办。因为看过面经，我先回答的是答案是把这个matrix翻转90度，然后一行行处理，但翻转的时候，读行输出列会有硬盘写文件耗时，读列输出行会有硬盘读文件耗时。

然后又回答说可以读一个正方形，一个正方形一个正方形处理。小哥让把code写一下，我就写了一段pseudocode。

然后小哥给分析了下发现这样有问题。如果处理matrix中间5x5矩阵，已知第一列中的五个值X，第二列只能算出来中间三个值，第三列只能算出来中间一个值。最后说还是只能翻转90度再一行行的做。

X O O O O X O O O O

X O O O O X X O O O

X O O O O --------> X X X O O

X O O O O X X O O O

X O O O O X O O O O

follow up:

如果图片是1million\*1million的怎么办，整张图片读不进内存，我答说dp结构可以改成一维的，然后每次只读一列。小哥说每次读一列的第一个字符非常耗时，因为要不断的跳读指针，然后我说可以对这个矩阵转置写到另外一个文件里，然后每次想做这个函数就对这个新文件操作就好了，这样就能按行读。小哥就问我怎么转置再存到另外一个文件里，我说根据内存大小可以多读几列。

然后小哥就说还可以再优化，他说这有一个balance，读行输出列，写文件就很耗时，读列输出行，读文件就很耗时（主要问题是 写指针或读指针跳转到下一行 所带来的时间消耗），其实听到这里我就应该有反应了，但当时还是傻傻的想。最后结果是每次根据内存大小读一个接近正方形的矩形，将他们写到新文件，再读下一块矩形。这样的话，读写指针跳转次数就最小了。

**写一下翻转代码**

刚面完，就问了一道题，设计两个function，分别是log\_hit()和get\_hit\_number\_in\_last\_5\_mins(). 之前在面经上也见过这题，就是要记录一个web被visited的次数以及返回过去5 min内被visited的次数。 聊了一下简历，问了一下问题。

**看一遍代码**

之前Dropbox店面了这道题，虽然已经做了，也准备了infinite board的情况，但还是挂了。诚求成功通过面试的python implementation！面试官没让楼主用O（1） space做。

依稀记得follow-up是，

1. What is the bottle-neck of your code？ 答曰：space complexity, it will run out of memory。 面试官不是很满意..

2. How do you make it more performant? 答曰：三行三行的读，然后更新center的count。 面试官质疑，你这样真的能正确更新吗。

然后开始写code，这是我的implementation，二日后卒

**看代码，同时写三行读取代码的**

1.16号海投夏天的实习，1.17号收到OA邀请。题目与之前面筋一致，给你一个用邻接矩阵表示的图，表示由若干个设备连接起来的网络；以及一个中毒设备的array，问哪一个中毒设备的网络面积最大。

与lc里面一道求岛屿的最大面积有相似之处，可以用并查集或者深搜解决。

第一轮 Find duplicated files，follow up加上了rolling hash的实现

这轮体验不是特别好，面试官迟到了10分钟，然后中间又出去了五分钟，最后也没有延长面试的时间，说他现在的room后面有人reserve了。不过后来recruiter说当时楼里有fire drill，表示了抱歉，继续约了下一轮

Duplicate file 那道题， 楼主虽然之前写过，full content comparison还是写错了，面试小哥指出来之后慌得一笔，后来还是改过来了。

基本就是size -> partial hash -> complete hash -> full content comparison，还问了symlink怎么办。

写了巨长的代码，小哥最后说虽然messy不过interview也就这样

1. file很大怎么办？会占用太多内存，或者内存不够，这个时候就要用hash，楼主当时提到了MD5 hash。MD5还是有极小的collision概率（不同的file可能会被hash到同一个Key），所以需要把MD5 hash匹配的文件再一个byte一个byte地匹配一遍，看是否是真的一样。

2. 如果有link path怎么办？本来只需要考虑file path和directory path，现在又加入了link path，也就是要考虑cycle的情况，保证不会无线循环。同时要考虑这个link path是指向给定root下的某个节点，还是不在给定root范围内的节点。

3. 最后还有10分钟，又出了一个followup：提供nextByte(file)，写一个compareFile(file1, file2)返回两个file是否一模一样

Input是一个文件路径和一个buffer

Output是判断这个文件是否包含这个buffer

(文件可能会非常大 超出内存)

按照块读取文件：

第二轮 File Access

/A <-- explicit access

|\_\_\_ /B

| |\_\_\_ /C <-- explicit access

| |\_\_\_ /D -- /k access

| ---

|\_\_\_ /E <-- access

|\_\_\_ /F

1) check if the user has the access to folder

2) 给某个folder加一个属性，表示这个folder的access不能被subfolder继承，修改代码

面到这里面试官说他可以给我通过了，接下来的时间我可以问他问题，也可以继续做bonus题目，我只能说继续bonus题目了。。。

3) bonus: remove redundant access

高频的duplicate files, game of life, design hit counter什么的都准备好了，结果今天碰见个新题。。。. check 1point3acres for more.

大意就是给很多对(child, parent)这样的文件关系，再给定一个set，user对这个set中的文件夹有direct access，对这些文件夹的子文件夹也有access。

现在给一个文件夹作为has\_access的参数，让你实现这个函数，查询user对它有没有访问权限。

其实有看到地里几天前有面经说这个题的，但是没太仔细准备。。参见http://www.1point3acres.com/bbs/thread-333274-1-1.htm

面试官一开始就说程序不用编译运行，main函数都不要，似乎input也不用做处理，我一开始就有点懵，有点纠结这程序结构到底写成啥样，input定义在哪里，事实证明面试官完全没care。。。

后面跟面试官确认思路和细节，然后开始写，写完面试官说Great. That should work. 最奇怪的是我也没被问follow up，问了我有什么问题之后就这么结束了。。

全程四十多分钟，感觉面试官有点心不在焉，经常我说完话过数秒才回应，有时候还让我再说一遍，不是很好的面试体验。。。。。

虽然不是很顺利，但是还是很想去onsite见识见识湾区第一食堂的QAQ 攒人品求昂赛

coderpad还没关，直接把面试官给的描述和example po上来，希望有帮助的话给加点米哈~~

/A

|\_\_\_ /B

|\_\_\_ /C <-- access

|\_\_\_ /D

|\_\_\_ /E <-- access

|\_\_\_ /F

|\_\_\_ /G

. 1point3acres

folders = [

('A', None),

('B', 'A'),

('C', 'B'),

('D', 'B'),

('E', 'A'),

('F', 'E'),

]

access = set(['C', 'E'])

has\_access(String folder\_name) -> boolean

has\_access("B") -> false

has\_access("C") -> true

has\_access("F") -> true

has\_access("G") -> true

/A <-- explicit access

|\_\_\_ /B

| |\_\_\_ /C <-- explicit access

| |\_\_\_ /D -- /k access

| ---

|\_\_\_ /E <-- access

|\_\_\_ /F

// corresponds to folders

比如，A为系统的root, B,E are A's subfolder, C and D are B's sub folder, if the has the access for B, then the one has B' access and all its subfolder's Access.

1) given all known access right of the file system, implement a method to check if the user has the right to certain folders

2) follow up, we want to remove redundant access information in our system, how to do that ? The redundant access information, could be, the user already has access to A, however, in some access list

it explicitly authorize the user access to folder C. This is redundant as, C is a subfoler of B, and B is sub-folder of A.

\*/

Set<String> access = new HashSet<String>();

access.add("C");

access.add("E");

// Child, Parent

List<String[]> folders = Arrays.asList(new String[][]{

{"A", null},

{"B", "A"},

{"C", "B"},

{"D", "B"},

{"E", "A"},

{"F", "E"},

});

Give a directory name (like /user/foo), return a list of files, in full path, under it.

For example, we have. From 1point 3acres bbs

/user/foo/fil1

/bar

/file11

/file12

/xyz

/file111

/file112

/file113

For the retured list, identfiy which two files are identical.

NASA存图片那题，具体请看http://www.1point3acres.com/bbs/thread-327178-1-1.html

做了一个2D vector记图片的disk directory

follow up 是问怎么找oldest image，答了linked list，写了基本的框架时间就到了

这道题主要考思路，他们的api给什么全靠问

Game of life

注意几个点：

1.follow up的时候会问时间空间复杂度 会问的很细到 比如 1m\*1m 大小的grid 空间复杂度会占用多少个bytes 楼主好久不做运算突然间算一下感觉还是有点生疏的

2.一开始没有直接给inline的转化方法用的是最土的存到另一个数组里 后来居然也没有让优化

3.follow up优化3行3行的读会要求写代码。提前练习一下准备好就行了 不用跑

题目疑似新题 Space Panoram

大意就是实现一个class用来存放image，同时支持update image和fetch image。最开始并没有给File的class，都是后来问了才说我们有这些api。期间会要你计算自己方法占用的内存是多少。

followup要求支持获得oldest的image，也就是最长时间没有更新的image，给了Sector的class，返回image对应的Sector，也就是坐标了。

面完第二天收到消息，说我们再来一轮电面吧。问hr为什么还要电面，是不是表现不够好，hr大概是为了安慰我说 Our interview process generally has 1-2 phone interviews, then a round of 3-4 onsite interviews. In this case, you were asked an experimental question so we'd love to do another round for more signal. 一脸懵逼。。。

废话不多说，下面是面试官给的api和要实现的class。

|  |
| --- |
| import java.io.\*;  import java.util.\*;  /\*\*  \* NASA selects Dropbox as its official partner, and we’re tasked with managing  \* a panorama for the universe. The Hubble telescope (or some other voyager we  \* have out there) will occasionally snap a photo of a sector of the universe,  \* and transmit it to us. You are to help write a data structure to manage this.  \* For the purpose of this problem, assume that the observable universe has been  \* divided into 2D sectors. Sectors are indexed by x- and y-coordinates.  \*/  public File {  public File(String path) {}  public Boolean exists() {}  public byte[] read() {}  public void write(bytes[] bytes) {}  }  public Image {  public Image(byte[] bytes) {}  byte[] getBytes() {} // no more than 1MB in size  }  public Sector {  public Sector(int x, int y) {}  int getX() {}  int getY() {}  }  /\*\*  \* row-major indexing to be consistent.  \*/  public class SpacePanorama {  /\*\*  \* initializes the data structure. rows x cols is the sector layout.  \* width, height can be as large as 1K each.  \*/  public SpacePanorama(int rows, int cols) {}  /\*\*  \* The Hubble will occasionally call this (via some radio wave communication)  \* to report new imagery for the sector at (y, x)  \* Images can be up to 1MB in size.  \*/  public void update(int y, int x, Image image) {}. check 1point3acres for more.  /\*\*  \* NASA will occasionally call this to check the view of a particular sector.  \*/. From 1point 3acres bbs  public Image fetch(int y, int x) {}  /\*\*  \* return the 2D index of the sector that has the stalest data.  \* the idea is that this may help the telescope decide where to aim next.  \*/  public Sector getStalestSector() {}  } |