把图片都写入到一个文件中，为此需要在本地构建一个文件索引表，（图片名、图片在文件中的偏移地址，图片大小），把图片作为大文件的记录插入，插入时需要同时维护这个文件索引表，读取图片时也是先查找这个索引表。

报告内容：请在报告中写明你的实现逻辑，和将每张图片单独作为小文件来存储的性能比较，并附上相应的代码段和截图。

**实现**

将本地的文件夹中的每个文件顺序存入hdfs，在本地创建local\_index文件，存储该文件夹中每个文件的偏移量。

查找某个文件时只需按照local\_index中的偏移量和该文件的长度来定位hdfs系统中的该文件位置，并将其下载。

String root\_path="/home/lwx/Documents/hw1/";

String hdfs\_fileName = "hw1/g2/Group\_file";

String local\_fileName = root\_path+"local\_index";

Configuration conf = new Configuration();

conf.set("fs.defaultFS", "hdfs://Master:9000");

conf.set("fs.hdfs.impl", "org.apache.hadoop.hdfs.DistributedFileSystem");

FileSystem fs = FileSystem.get(conf);

Path hdfs\_file\_path = new Path(hdfs\_fileName);

File local\_index\_file\_obj=new File(local\_fileName);

if(!local\_index\_file\_obj.exists())

{

//this hdfs file path not exist

//create index local file

System.out.println("create local index file");

create\_local\_index\_file(local\_fileName);

}

//readFileIndex(local);

if(args[0].equalsIgnoreCase("build"))

{

System.out.println("build");

read\_dir(args[1]);

// for(int j =0; j<file\_num\_now;j++)

// System.out.println(f\_name[j]);

save\_File(fs,hdfs\_file\_path,local\_fileName);

}

else if(args[0].equalsIgnoreCase("query"))

{

readFileIndex(local\_fileName);

queryFile(fs,hdfs\_file\_path,args[1]);

}

fs.close();

**savefile函数：将文件夹上传到hdfs中并在本地创建local\_index索引**

public static void save\_File(FileSystem fs, Path hdfs\_file\_path,String local\_index) throws IOException{

FSDataOutputStream os =fs.create(hdfs\_file\_path);

File local\_index\_obj =new File(local\_index);

FileWriter fileWriter = new FileWriter(local\_index\_obj.getName(),true);

System.out.println("will write index into index file : "+local\_index);

System.out.println("file num now: "+file\_num\_now);

fileWriter.write(file\_num\_now + "\n");

fileWriter.flush();

/\*

byte[] Total\_byte = intToByteArray(200);

os.write(Total\_byte,0,Total\_byte.length);

byte[] now\_byte = intToByteArray(0);

os.write(now\_byte,4,now\_byte.length); \*/

//int file\_off\_counter = 0;

for(int i = 0; i<file\_num\_now; i++)

{

fileWriter.write(file\_off[i] + "\n");

fileWriter.write(file\_len[i] + "\n");

fileWriter.write(f\_name[i] + "\n");

}

fileWriter.close();

byte[] buffer = new byte[4096];

for(int i = 0; i<file\_num\_now; i++)

{

//byte[] buffer = new byte[4096];

File f = new File(directory\_path + f\_name[i]);

int t;

//System.out.println(i+" len="+file\_len[i]);

InputStream in = new FileInputStream(f);

for(int j = 0; j < file\_len[i]; j+=4096)

{

if(j + 4096 <= file\_len[i])

t = 4096;

else

t = file\_len[i] -j;

//System.out.println(j+"j t"+t);

in.read( buffer, 0, t);

os.write(buffer, 0, t);

//file\_off\_counter += t;

}

in.close();

}

System.out.println("Save file in hdfs:"+hdfs\_file\_path.getName());

os.close();

}

**query\_file函数：按照local\_index获取hdfs中的某个文件**

public static void queryFile(FileSystem fs, Path hdfs\_file\_path, String name) throws IOException

{

int i = 0;

for(; i < file\_num\_now; i++)

{

if(f\_name[i].equals(name))

break;

}

if(i==file\_num\_now)

{

System.out.println(name+" doesn't exist");

return;

}

FSDataInputStream in = fs.open(hdfs\_file\_path);

FileOutputStream out = new FileOutputStream(new File("fetched\_"+name));

BufferedReader d = new BufferedReader(new InputStreamReader(in));

byte[] tmpb = new byte[4096];

int t;

in.seek(file\_off[i]);

for(int j = 0; j<file\_len[i]; j+=4096)

{

if(j + 4096 <= file\_len[i])

t = 4096;

else

t = file\_len[i] -j;

in.read(tmpb, 0, t);

out.write(tmpb,0,t);

}

out.close();

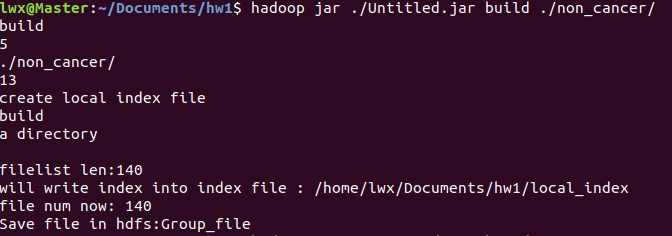
d.close();

in.close();

}

**截图**

上传文件夹到hdfs中



查找某个文件并下载到本地

