深 圳 大 学 实 验 报 告

课程名称：操作系统

实验项目名称：实验 4 文件管理

学院：计算机与软件学院

专业：计算机科学与技术

指导教师：谭舜泉

报告人：冯海月 黎浩然 学号：2018191116 2018112061

实验时间：2021 年 6 月 15 日-2021 年 6 月 20 日

实验报告提交时间：2021 年 6 月 20 日

教务部制

实验目的与要求：

实验目的：

(1)、掌握计算机操作系统管理进程、处理机、存储器、文件系统的基本方法。

(2)、了解进程的创建、撤消和运行，进程并发执行；自行设计解决哲学家就餐问题的并发线程，了解线程（进程）调度方法；掌握内存空间的分配与回收的基本原理；通过模拟文件管理的工作过程，了解文件操作命令的实质。

(3)、了解现代计算机操作系统的工作原理，具有初步分析、设计操作系统的能力。

(4)、通过在计算机上编程实现操作系统中的各种管理功能，在系统程序设计能力方面得到提升。

实验要求：

(1)、题目 1：

xv6 采用混合索引分配方式。阅读以下英文介绍：

The on-disk inode structure, struct dinode, contains a a size and a list of block numbers. The inode data is found in the blocks listed in the dinode's addrs array. The first NDIRECT blocks of data are listed in the first NDIRECT entries in the array; these blocks are called "direct blocks". The next NINDIRECT blocks of data are listed not in the inode but in a data block called the "indirect block". The last entry in the addrs array gives the address of the indirect block.

请联系代码分析，回答以下问题：

只采用直接分配方式，xv6 支持的最大文件尺寸是多少？

结合直接分配方式和索引分配方式，xv6 支持的最大文件尺寸是多少？

说明：

（1）本次实验课作业满分为 100 分，占总成绩的比例（待定）。

（2）本次实验课作业截至时间 2021 年 6 月 20 日（周日）23:59。

（3）报告正文：请在指定位置填写，本次实验不需要单独提交源程序文件。

（4）个人信息：WORD 文件名中的“姓名”、“学号”，请改为你的姓名和学号；实验报告的首页，请准确填写“学院”、“专业”、“报告人”、“学号”、“班级”、“实验报告提交时间”等信息。

（5）提交方式：请在 BLACKBOARD 平台中按时提交；延迟提交不得分。

（6）发现抄袭（包括复制&粘贴整句话、整张图），该次作业记零分。

（7）期末考试阶段补交无效。

(1)、题目 1：

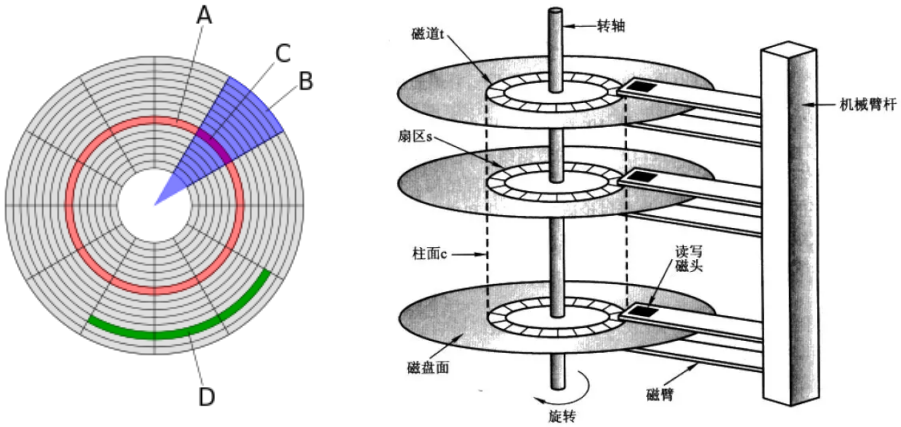
xv6 采用混合索引分配方式。阅读以下英文介绍：

The on-disk inode structure, struct dinode, contains a a size and a list of block numbers. The inode data is found in the blocks listed in the dinode's addrs array. The first NDIRECT blocks of data are listed in the first NDIRECT entries in the array; these blocks are called "direct blocks". The next NINDIRECT blocks of data are listed not in the inode but in a data block called the "indirect block". The last entry in the addrs array gives the address of the indirect block.

请联系代码分析，回答以下问题：

只采用直接分配方式，xv6 支持的最大文件尺寸是多少？

结合直接分配方式和索引分配方式，xv6 支持的最大文件尺寸是多少？



在计算机磁盘存储器中，比如传统的机械硬盘驱动器 (HDD)由多个盘片构成。每一个盘片有两个盘面(surface)；如上图，每一个盘面由多个同心环构成，其中每一个同心环就是一个磁道(track)；而每一个磁道又被分为多个扇区(sector)。

因此扇区是磁盘的最小读/写单位。

```
#define BSIZE 512  // block size

#define NDIRECT 12
#define NINDIRECT (BSIZE / sizeof(uint)) = 512/4 = 128
```

在 xv6 中，一个磁盘块(block)的大小是 512 字节，也就是一个扇区。

每一个文件，都有一个与之对应 inode 数据结构。Inode 本身不存放文件数据本身，而是存放这个文件的元信息(metadata)：如文件类型，文件所在的设备、文件大小等。

```
// On-disk inode structure
struct dinode {                                  用于区分 -- 文件、目录、特殊文件...
  short type;           // File type ──→ type=0意味着空闲节点
  short major;          // Major device number (T_DEV only)
  short minor;          // Minor device number (T_DEV only)
  short nlink;          // Number of links to inode in file system ──→ 指向这个节点的链接数
  uint size;            // Size of file (bytes)
  uint addrs[NDIRECT+1];   // Data block addresses ──→ 对应这个节点的数据块
};
```

在 xv6 中，磁盘上的 inode 由一个 struct dinode 定义：

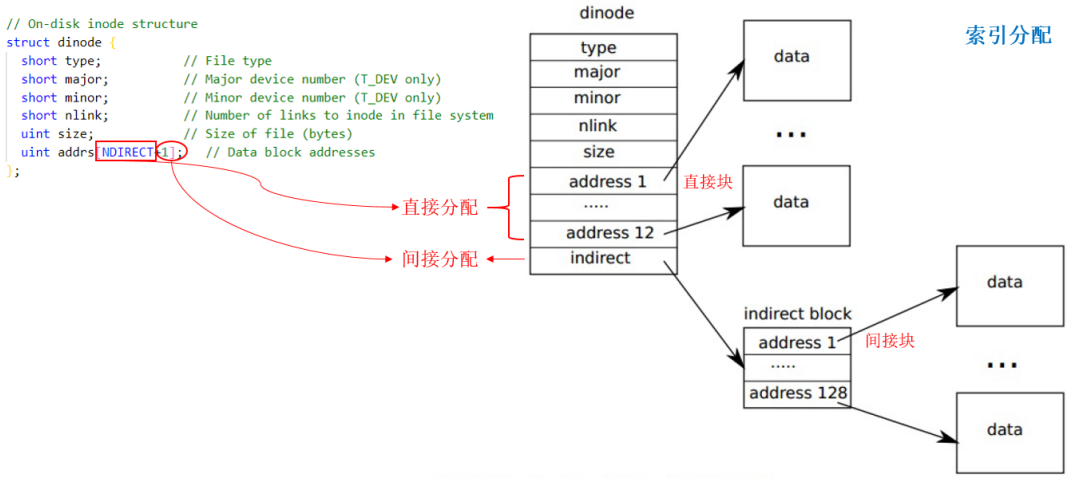
Inode 只有在活动的状态会被载入内存中。结构体 inode 是磁盘中的结构体 dinode 在内存中的拓展。内存中的 inode 定义如下。

```
// in-memory copy of an inode
struct inode {
  uint dev;           // Device number
  uint inum;          // Inode number
  int ref;            // Reference count
  int flags;          // I_BUSY, I_VALID

  short type;         // copy of disk inode
  short major;
  short minor;
  short nlink;
  uint size;
  uint addrs[NDIRECT+1];
};
```

统计有多少个指针指向这个节点。如果ref-0，这个节点将被丢弃

在上面结构体的最后一个成员 addrs 是一个地址数组，这个数组的每一个元素存放着一个块地址。由于 NDIRECT 被宏定义为 12，因此这个数组的大小为 13。



其中，最开始的 NDIRECT 个块是直接索引块，接下来的 KINDIRECT 个块没有直接存在 inode 中，而是存在 addrs 最后一个地址指向的数据块中，称为间接块。

一个间接块的大小为 512 字节，存放着 512/4=128 个块地址，这个 128 个块就是文件的通过间接索引分配的数据块。

```
246  void
247  iappend(uint inum, void *xp, int n)
248  {
249    char *p = (char*)xp;
250    uint fbn, off, n1;
251    struct dinode din;
252    char buf[512];
253    uint indirect[NINDIRECT];
254    uint x;
255
256    rinode(inum, &din);
257
258    off = xint(din.size);
259    while(n > 0){
260      fbn = off / 512;
261      assert(fbn < MAXFILE);
262      if(fbn < NDIRECT){
263        if(xint(din.addrs[fbn]) == 0){
264          din.addrs[fbn] = xint(freeblock++);
265          usedblocks++;
266        }
267        x = xint(din.addrs[fbn]);
268      } else {
269        if(xint(din.addrs[NDIRECT]) == 0){
270          // printf("allocate indirect block\n");
271          din.addrs[NDIRECT] = xint(freeblock++);
272          usedblocks++;
273        }
274        // printf("read indirect block\n");
275        rsect(xint(din.addrs[NDIRECT]), (char*)indirect);
276        if(indirect[fbn - NDIRECT] == 0){
277          indirect[fbn - NDIRECT] = xint(freeblock++);
278          usedblocks++;
279          wsect(xint(din.addrs[NDIRECT]), (char*)indirect);
280        }
281        x = xint(indirect[fbn-NDIRECT]);
282      }
283      n1 = min(n, (fbn + 1) * 512 - off);
284      rsect(x, buf);
285      bcopy(p, buf + off - (fbn * 512), n1);
286      wsect(x, buf);
287      n -= n1;
288      off += n1;
289      p += n1;
290    }
291    din.size = xint(off);
292    winode(inum, &din);
293  }
```

在 iappend 函数中：当 fbn 小于 12 的时候，xint 返回的块地址被直接放入 addrs 数组中；当 fbn 大于等于 12 的时候，xint 返回的块地址被放入 addrs 的最后一个元素所指向的数组/间接块中。

由此观之，

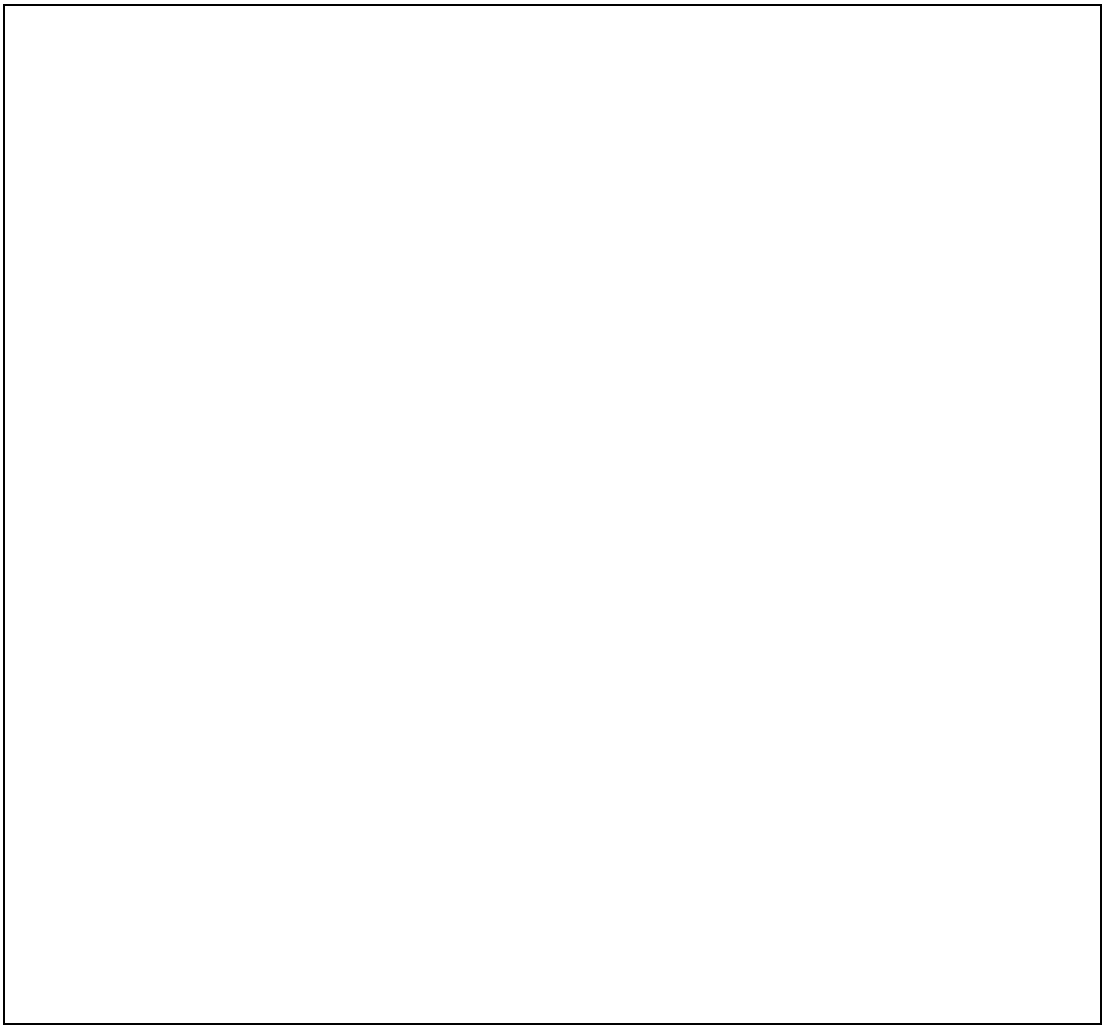
（1） 若只采用直接分配方式， xv6 支持的最大文件尺寸是 NDIRECT*BSIZE = 12*512B = 12*0.5KB = 6KB。

（2） 若采用直接分配和间接索引分配形式，xv6 所支持的最大文件尺寸可以扩展到 6KB + NINDIRECT*BSIZE = 6KB + 128*512B = 70KB

++

其他（例如感想、建议等等）。

通过代码初步了解 xv6 文件系统的组织形式，理解 xv6 的文件存储结构，经过计算得到不同分配形式下 xv6 所支持的最大文件尺寸。加深了对 inode 等文件相关的数据结构的理解。



指导教师批阅意见：   成绩评定：   指导教师签字：谭舜泉 2021 年　6 月 27 日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。