**广州大学学生实验报告**

开课学院及实验室: 计算机科学与网络工程学院 2024 年 5月 31日

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 学院 | 计算机科学与网络工程学院 | 年级/专业/班 |  | 姓名 |  | 学号 |  |
| 实验课程名称 | 操作系统实验 | | | | | 成绩 |  |
| 实验项目名称 | 磁盘管理 | | | | | 指导老师 |  |

**实验五 磁盘管理**

1. **实验目的：**

**1、了解磁盘调度的策略和原理；**

**2、理解和掌握磁盘调度算法——先来先服务算法（FCFS）、最短寻道时间优先算法（SSTF）、电梯扫描算法（SCAN）。**

**二、基本要求：**

**1、输入为一组请求访问磁道序列，该序列和所选磁道个数要求随机生成，输出为每种调度算法的磁头移动轨迹和移动的总磁道数；**

**2、输入磁道范围 0~1000 ，输入所选磁道个数0~1000；**

**3、画出主程序流程图；**

**4、编写程序并调试；**

**5、截屏输出实验结果；**

**6、根据实验结果与理论课讲述的原理进行实验分析**

**三、实验软件：**

**visual studio C++、Windows操作系统**

**四、实验内容：**

**1.随机生成磁道:**

**int** N**;***//磁道个数*

**int\*** track\_list**;***//磁道序列*

**int** head\_begin**;***//初始磁头位置*

*//初始化，随机生成磁道序列，磁头位置*

**void** init**()**

**{**

    srand**(**time**(**NULL**));***//设置随机数种子*

    N **=** 10**;**

*//(rand() % 1001)+1;//随机生成磁道个数*

    track\_list **=** **new** **int[**N**];**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

        track\_list**[**i**]** **=** rand**()** **%** 1001**;***//随机生成磁道序列*

    head\_begin **=** rand**()** **%** 1001**;***//初始化磁头位置*

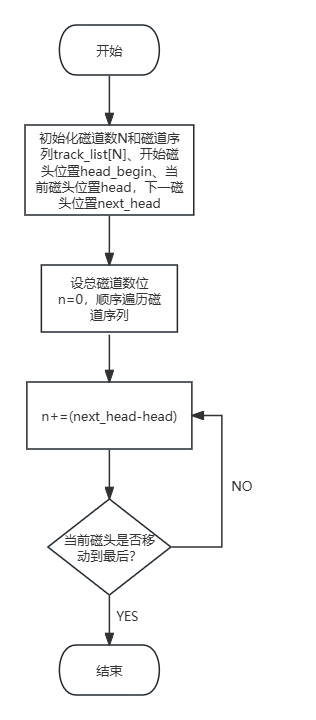
**}**

**实验要求磁道个数N随机数生成1到1000个，但生成数量大，不便于实验结果展示，因此将N==10来演示实验结果，随机生成N的代码已加以注释。**

**2.模拟先来先服务法（First-Come, First-Served，FCFS），最短寻道时间优先法（Shortest Seek Time First， SSTF），电梯扫描算法（SCAN）三种磁盘调度算法；**

**(1)先来先服务法(FCFS):**

**①实验流程图:**

****

**②程序代码:**

*//先来先服务算法:*

**void** FCFS**()**

**{**

**int** head **=** head\_begin**;**

**int** next\_head **=** head\_begin**;**

**int** n **=** 0**;***//总磁道位数*

    cout **<<** **"FCFS磁头移动顺序:";**

**for** **(int** i **=** 0**;**i**<**N**;** **++**i**)***//遍历磁道序列*

**{**

        next\_head **=** track\_list**[**i**];**

        n **+=** abs**(**next\_head **-** head**);**

        head **=** next\_head**;**

**if** **(**i **!=** N **-** 1**)**cout **<<** next\_head **<<** **"->";**

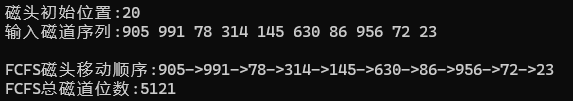
**else** cout **<<** next\_head **<<** endl**;**

**}**

    cout **<<** **"FCFS总磁道位数:"** **<<** n **<<** endl**;**

**}**

**③结果演示:**

****

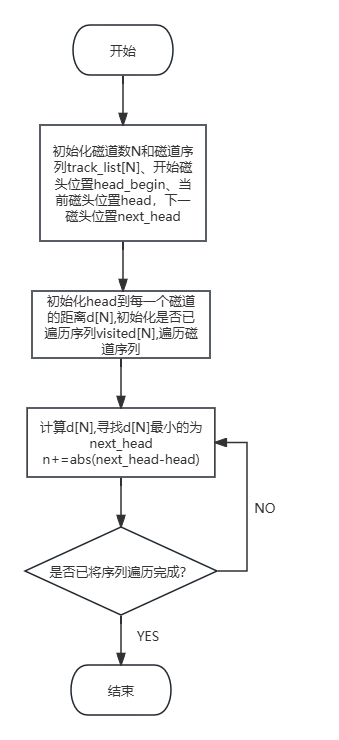
**④结果分析:**

**先来先服务，磁头移动顺序由磁道序列从左往右寻道:**

**总磁道位数=|905-20|+|991-905|+|991-78|+|314-78|+|145-314|+|630-145|+|86-630|+|956-86|+|72-956|+|23-72|=5121**

**(2)最短寻道时间优先法(SSTF):**

**①实验流程图:**

****

**②程序代码:**

**核心代码:**

**void** SSTF**()**

**{**

**int** n **=** 0**;***//总磁道数*

**int** head **=** head\_begin**;***//当前磁头位置*

**int** next\_head **=** head\_begin**;***//下一磁头位置*

**int\*** d **=** **new** **int[**N**];***//距离*

**bool\*** visited **=** **new** **bool[**N**];**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)***//初始距离d[N]*

**{**

        d**[**i**]** **=** INT32\_MAX**;**

        visited**[**i**]** **=** **false;**

**}**

    cout **<<** **"SSTF磁头移动顺序:";**

*//未遍历完成时*

**while** **(!**is\_visited**(**visited**))**

**{**

*//计算d[N]*

        upgrade\_d**(**head**,**d**);**

*//d中最小值的位置k*

**int** k **=** 0**;**

**int** min **=** INT32\_MAX**;**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

**if** **(!**visited**[**i**]** **&&** min **>=** d**[**i**])**

**{**k **=** i**;**min **=** d**[**i**];}**

*//计算n*

        next\_head **=** track\_list**[**k**];**

        n **+=** abs**(**head **-** next\_head**);**

        visited**[**k**]** **=** **true;**

        head **=** next\_head**;**

*//输出寻道顺序*

**if** **(**is\_visited**(**visited**))**cout **<<** next\_head **<<** endl**;**

**else** cout **<<** next\_head **<<** **"->";**

**}**

    cout **<<** **"SSTF总磁道数:"<<**n**<<**endl**;**

**}**

**判断是否寻道完成函数is\_visited(bool visited[]):**

*//判断是否已经寻道完毕*

**bool** is\_visited**(bool** visited**[])**

**{***//寻道完毕返回true，否则返回false*

**bool** res **=** **true;**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

        res **\*=** visited**[**i**];**

**return** res**;**

**}**

**计算距离upgrade\_d():**

*//计算下一磁道与当前磁头的距离*

**void** upgrade\_d**(int** head**,int** d**[])**

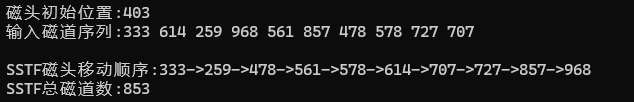
**{**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

        d**[**i**]** **=** abs**(**head **-** track\_list**[**i**]);**

**}**

**③结果演示:**

****

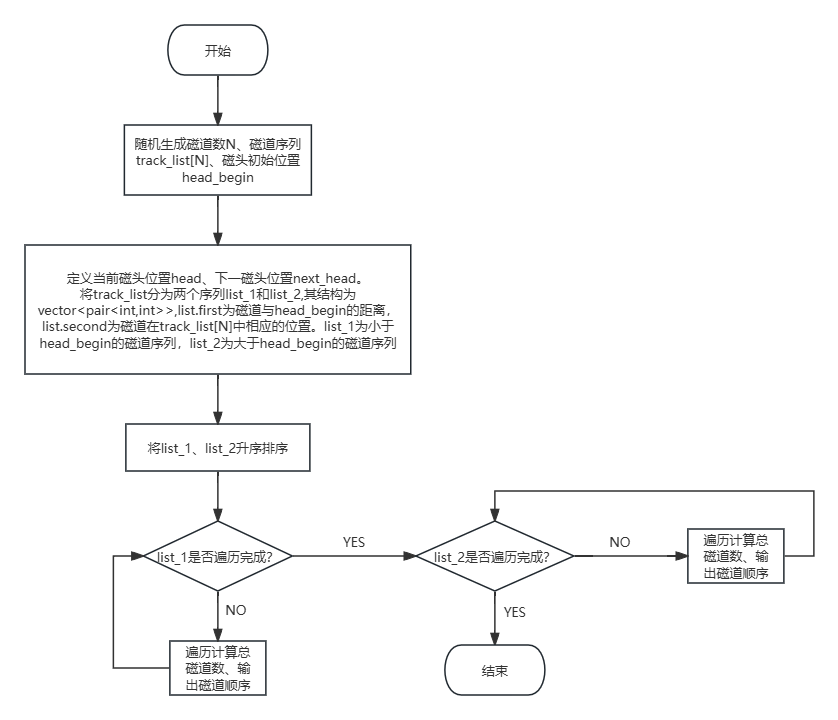
**④结果分析:**

**在SSTF算法中，磁头寻道总是找距离磁头最近的磁道:**

**磁道总数=|403-333|+|333-259|+|259-478|+|478-561|+|561-578|+|578-614|+|614-707|+|707-727|+|857-727|+|968-857|=853**

**(3)电梯扫描算法(SCAN):**

**①实验流程图:**

****

**②程序代码:**

**核心代码:**

*//电梯扫描算法*

**void** SCAN**()**

**{**

**bool\*** visited **=** **new** **bool[**N**];***//标记是否已寻道*

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**visited**[**i**]** **=** **false;**

**int** n **=** 0**;***//磁头移动总数*

*//将track\_list分为两个序列,一个大于head\_begin,一个小于head\_begin*

    vector**<**pair**<int,** **int>>**list\_1**;***//小于head\_begin*

    vector**<**pair**<int,** **int>>**list\_2**;***//大于head\_begin*

*//list.first为head\_begin到点track\_list[list.second]的距离*

**int** head **=** head\_begin**;**

**int** next\_head **=** head\_begin**;**

**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

**{**

**if** **(**track\_list**[**i**]** **<=** head\_begin**)**

            list\_1**.**push\_back**(**{ abs**(**head\_begin **-** track\_list**[**i**]),**i }**);**

**else**

            list\_2**.**push\_back**(**{ abs**(**head\_begin **-** track\_list**[**i**]),**i }**);**

**}**

*//对上述两个得到的序列进行升序排序，使得让磁头每次移动都是递归下降或递归上升的*

    sort**(**list\_1**.**begin**(),** list\_1**.**end**());**

    sort**(**list\_2**.**begin**(),** list\_2**.**end**());**

    cout **<<** **"SCAN磁头移动顺序:";**

**for** **(int** i **=**0**;** i**<**list\_1**.**size**();** **++**i**)**

**{**

        next\_head **=** track\_list**[**list\_1**[**i**].**second**];**

        n **+=** abs**(**head **-** next\_head**);**

        head **=** next\_head**;**

        visited**[**list\_1**[**i**].**second**]** **=** **true;**

*//输出磁头移动*

**if** **(!**is\_visited**(**visited**))**cout **<<** next\_head **<<** **"->";**

**else** cout **<<** next\_head**;**

**}**

**for** **(int** i **=** 0**;** i **<** list\_2**.**size**();** **++**i**)**

**{**

        next\_head **=** track\_list**[**list\_2**[**i**].**second**];**

        n **+=** abs**(**head **-** next\_head**);**

        head **=** next\_head**;**

        visited**[**list\_2**[**i**].**second**]** **=** **true;**

*//输出磁头移动*

**if** **(!**is\_visited**(**visited**))**cout **<<** next\_head **<<** **"->";**

**else** cout **<<** next\_head**;**

**}**

    cout **<<** endl **<<** **"SCAN总磁道数:"** **<<** n **<<** endl**;**

**}**

**判断是否寻道完成函数is\_visited(bool visited[]):**

*//判断是否已经寻道完毕*

**bool** is\_visited**(bool** visited**[])**

**{***//寻道完毕返回true，否则返回false*

**bool** res **=** **true;**

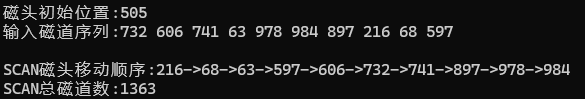
**for** **(int** i **=** 0**;** i **<** N**;** **++**i**)**

        res **\*=** visited**[**i**];**

**return** res**;**

**}**

**③结果演示:**

****

**④结果分析:**

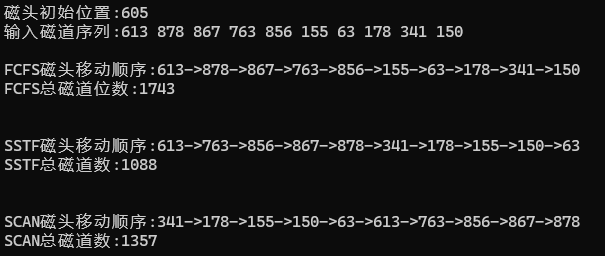
**SCAN的寻道顺序为:先递减地移动到最左端，再递增地移动到最右端（先递减地移动到最右端，再递增地移动到最左端）**

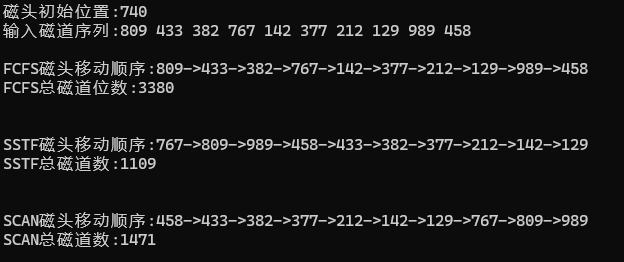
**总磁道数=|505-216|+|216-68|+|68-63|+|63-597|+|597-606|+|606-732|+|732-741|+|897-741|+|978-897|+|984-978|=1363**

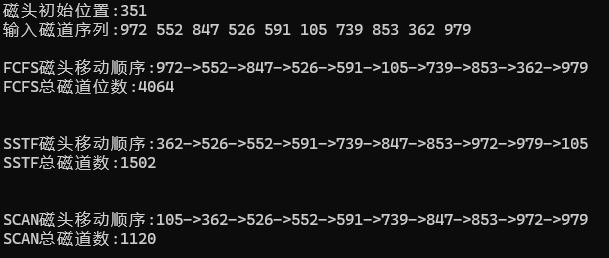
**3.对三种算法进行对比分析。**

**输入为一组请求访问磁道序列，输出为每种调度算法的磁头移动轨迹和移动的总磁道数。**

**进行多次试验结果如下:**

****

****

****

**结果分析:**

**由实验结果可知，先来先服务算法(FCFS)移动总磁道数最大，效率最低。**

**最短寻道时间优先算法(SSTF)和电梯扫描算法(SCAN)，两者之间差别不大，两者的差别主要来与初始磁头的位置，当磁头初始位置位于磁道两端时(0或1000)，两者总磁道数相等。**

**五、实验思考：**

**1、通过对每个算法进行时间复杂度分析对比，每个算法的效率如何？**

**答:**

**FCFS算法来一个请求即寻道一次，若有n个请求，时间复杂度为O（n）;**

**SSTF算法，若有n个请求，在处理每个请求前需要对请求序列进行一次遍历，找出距离最小的磁道，时间复杂度为O(n^2);**

**SCAN算法，若有n个请求，先对n个请求进行排序，随后直接遍历即可，取决于排序的算法;**

**2、若所有硬盘全部设计成电子硬盘，哪个磁盘调度算法最合适？**

**答:**

**电子硬盘不存在物理寻磁道的操作，因此选择算法复杂度最简单的先来先服务算法（FCFS）。**