**《计算机操作系统》课程**

**实验报告**

 实验日期：\_2024.12.2\_\_\_\_\_\_\_

实验名称： 实验四 内存管理及算法实验

班级：\_22计算机3\_ 学号：\_2200303310\_ 姓名：\_陈忠鹏 成绩：\_ \_\_\_\_

# 一、实验目的

了解磁盘调度的策略和原理；

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、 实验内容、算法流程图及步骤

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；

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。





# 三、实验结论和体会

这次实验让我体会到选择一个合适的算法可以提高资源访问的效率，而设计一个好的算法更不易中的不易。

# 附录（源代码）

// #pragma GCC optimize(2)

#include <algorithm>

#include <bits/stdc++.h>

#include <iomanip>

// #include "testlib.h"

typedef long long ll;

using namespace std;

const int maxn = 2e2 + 7;

const int N = 8;

const ll INF = 1e18;

ll n, m, k, x, y, z;

#define rep(i, n) for (int i = 0; i < (int)n; i++)

#define FOR(n) for (int i = 0; i < (int)n; i++)

#define repi(i, a, b) for (int i = (int)a; i < (int)b; i++)

vector<ll> cd;

ll len;

const int st = 100;

void FCFS() {

 ll tot = 0;

 ll pre = st;

 vector<ll> res;

 FOR(len) {

 tot += abs(cd[i] - pre);

 pre = cd[i];

 res.push\_back(cd[i]);

 }

 cout << "FCFS 平均寻道长度: " << fixed << setprecision(6) << 1.0 \* tot / len

 << '\n';

 cout << "寻道顺序:\n";

 for (auto v : res) {

 cout << v << ' ';

 }

 cout << '\n';

}

void SSTF() {

 vector<ll> tmp = cd;

 ll dir = 0;

 sort(tmp.begin(), tmp.begin() + len);

 vector<ll> res;

 if (tmp[0] > st) {

 dir = 0;

 }

 if (tmp[len - 1] < st) {

 dir = len;

 }

 repi(i, 1, len) {

 if (tmp[i] > st && tmp[i - 1] < st) {

 dir = i;

 }

 }

 ll ld = dir - 1, rd = dir;

 ll tot = 0;

 ll pre = st;

 while (ld >= 0 && rd < len) {

 if (pre - tmp[ld] <= tmp[rd] - pre) {

 tot += pre - tmp[ld];

 pre = tmp[ld];

 res.push\_back(tmp[ld]);

 ld--;

 } else {

 tot += tmp[rd] - pre;

 pre = tmp[rd];

 res.push\_back(tmp[rd]);

 rd++;

 }

 }

 while (ld >= 0) {

 tot += pre - tmp[ld];

 pre = tmp[ld];

 res.push\_back(tmp[ld]);

 ld--;

 }

 while (rd < len) {

 tot += tmp[rd] - pre;

 pre = tmp[rd];

 res.push\_back(tmp[rd]);

 rd++;

 }

 cout << "SSTF 平均寻道长度: " << fixed << setprecision(6) << 1.0 \* tot / len

 << '\n';

 cout << "寻道顺序:\n";

 for (auto v : res) {

 cout << v << ' ';

 }

 cout << '\n';

}

void SCAN() {

 vector<ll> tmp = cd;

 ll dir = 0;

 sort(tmp.begin(), tmp.begin() + len);

 if (tmp[0] > st) {

 dir = 0;

 }

 if (tmp[len - 1] < st) {

 dir = len;

 }

 repi(i, 1, len) {

 if (tmp[i] > st && tmp[i - 1] < st) {

 dir = i;

 }

 }

 ll ld = dir - 1, rd = dir;

 ll tot = 0;

 ll pre = st;

 vector<ll> res;

 cout << "递减方向访问\n";

 while (ld >= 0) {

 tot += pre - tmp[ld];

 pre = tmp[ld];

 res.push\_back(tmp[ld]);

 ld--;

 }

 while (rd < len) {

 tot += tmp[rd] - pre;

 pre = tmp[rd];

 res.push\_back(tmp[rd]);

 rd++;

 }

 cout << "SCAN 平均寻道长度: " << fixed << setprecision(6) << 1.0 \* tot / len

 << '\n';

 cout << "寻道顺序:\n";

 for (auto v : res) {

 cout << v << ' ';

 }

 res.clear();

 cout << '\n';

 ld = dir - 1, rd = dir;

 tot = 0;

 pre = st;

 while (rd < len) {

 tot += tmp[rd] - pre;

 pre = tmp[rd];

 res.push\_back(tmp[rd]);

 rd++;

 }

 while (ld >= 0) {

 tot += pre - tmp[ld];

 pre = tmp[ld];

 res.push\_back(tmp[ld]);

 ld--;

 }

 cout << "递增方向访问\n";

 cout << "SCAN 平均寻道长度: " << fixed << setprecision(6) << 1.0 \* tot / len

 << '\n';

 cout << "寻道顺序:\n";

 for (auto v : res) {

 cout << v << ' ';

 }

 cout << '\n';

}

void solve() {

 FCFS();

 SSTF();

 SCAN();

}

int main(int argc, char \*argv[]) {

 ll t = 1;

 // registerGen(argc, argv, 1);

 srand(2014);

 n = N;

 // FOR(320) { printf("%d ", rand() % 10 + 1); }

 FOR(n \* 2) { cd.push\_back(rand() % 201 + 1); }

 len = unique(cd.begin(), cd.end()) - cd.begin();

 cout << "初始位置： " << st << '\n';

 cout << "请求队列数量: " << len << '\n';

 FOR(len) { cout << cd[i] << " \n"[i == len - 1]; }

 while (t--)

 solve();

 return 0;

}