磁盘文件最优存储

算法思路

首先n个文件的检索时间为：

$$\sum_{1\leqslant i<j\leqslant n} p_i p_j d(i,j)$$

为了简化问题，首先将j的位置固定，同时假设每个点的访问概率相同，则问题化简为：

$$\mathrm{Min}\sum_{i=1}^{n}|i-j|$$

对目标函数进行进一步计算得到：

$$\begin{aligned}&\textstyle\sum_{i=1}^{n}|i-j| = \sum_{i=1}^{j}(j-i)+\sum_{i=j+1}^{n}(i-j)\\&= j\cdot j-\textstyle\sum_{i=1}^{j} i+\sum_{i=j+1}^{n} i-(n-j)j\\&= j^2-\frac{j(j+1)}{2}+\frac{n^2+n-j-j^2}{2}-nj+j^2\\&= j^2-(n+1)j+\frac{n(n+1)}{2}\end{aligned}$$

则以j为自变量，当且仅当在j = (n+1)/2 处取的最小值，同时根据二次函数的性质可知，j离（n+1）/2越远，在概率相等的条件下下单个文件的总检索时间越长。

为了最小化所有文件的检索时间，则采用贪心策略，在检索代价最小的地方，放检索概率最大的文件，使得总检索时间最小。

算法实现

```
#include <iostream>

using namespace std;
bool cmp(int pro1, int pro2){
    return pro1 > pro2;
}

double min_time(int num, double *pro){
    int mid = (num + 1) / 2 - 1;
    int right = -1;
    int offset = 0;
    double disk_pro[num];
    double time = 0;
    for (int i = 0; i < num; i++) {
```

```
        disk_pro[mid + right * offset] = pro[i];
        right *= -1;
        if (right == 1) {
            offset++;
        }
    }
    for (int i = 0; i < num; i++) {
        for (int j = i + 1; j < num ; j++) {
            time += disk_pro[i] * disk_pro[j] * abs(i - j);
        }
    }
    return time;
}

int main(int argc, const char * argv[]) {
    int num;
    cin >> num;
    double file_pro[num];
    double pro_sum = 0;
    for (int i = 0; i < num; i++) {
        cin >> file_pro[i];
        pro_sum += file_pro[i];
    }
    for (int i = 0; i < num; i++) {
        file_pro[i] /= pro_sum;
    }
    sort(file_pro, file_pro + num, cmp);
    cout << "output: "<< min_time(num, file_pro) ;
    return 0;
}
```

运行结果

```
5
33 55 22 11 9
output: 0.573432Program ended with exit code: 0
```