上机 9

张配天-2018202180

2020 年 11 月 11 日

目录

1 问题描述

在滑完全程的前提下求最小的补水次数。

1.1 输入

补水点位置的数组 $pos[n]$；

1.2 输出

最少的补水次数和补水位置；

2 算法思路

2.1 最优子结构

假设最终最少的补水位置的序列为 $\mathcal{H}^k = \{h_1, ..., h_k\}$，$h_*$ 为补水站位置，其中有 $h_i = pos[j]$，那么 $\mathcal{H}^{i-1}, \mathcal{H}^{i+1:k}$ 一定是最少补水位置序列；否则若存在更少的补水位置序列 $\mathcal{H}^{i-1'}$，则由剪贴复制法，得到 $\mathcal{H}^k$ 不是最少补水序列。

2.2 设计递归算法

在不补水就无法前往下一个补水点的时候补水，即若 $(h_{i+1} - h_i) \cdot \frac{2}{m} > remain$，则在 h_i 补水。

2.3 证明贪心是安全的且最优解包含其中

Theory 2.1. *设 $\mathcal{H}_j \in \mathcal{H}^{j:k}$ 是第 j 次补水后的最少补水序列，h_m 是第 j 次补水后最远能滑行到的补水地点，则 h_m 包含在 $\mathcal{H}^{j:k}$ 中*

证明：

1. 如果 $\mathcal{H}_j[0] == h_m$，那么已经得证
2. 否则，令 $\mathcal{H}'_j = \mathcal{H}_j[1:] \cup h_m$，由于 $h_m > \mathcal{H}_j[0]$，则替换 $\mathcal{H}_j[0]$ 不会影响之后的补水序列 $\mathcal{H}_{j+1}$，且总补水次数不变，因此 $\mathcal{H}'_j$ 也是最优补水序列，得证。

3 复杂度分析

记补水站序列长度为 n

- 第 4-5 行初始化，消耗时间 $O(n)$
- 第 12-17 行便利补水站序列，按照上述规则挑选补水站进行补水，每次补水最多计算 $O(1)$ 次，因此总时间复杂度 $O(n)$
- 无需回溯，直接打印，复杂度 $O(n)$

综上，整体算法时间复杂度 $O(n)$，空间复杂度 $O(n)$

4 源代码

```python
import numpy as np
total_length = 13
m = 2
pos = [0,1,2.6,4,4.2,4.4,5,6,7,8.1,9.2,9.5,10,11,12]
pos.append(total_length)

water_per_mile = 2/2

hydrate = []
remain = 2

for i,position in enumerate(pos[0:-1]):
    if (pos[i+1]-pos[i])*water_per_mile > remain:
        hydrate.append(position)
        remain = 2-(pos[i+1]-pos[i])*water_per_mile
    else:
        remain -= (pos[i+1]-pos[i])*water_per_mile

hydrate.append(total_length)
print(hydrate)
```

5 运行截图

- 自定义补水站序列为 $[(0), 1, 2.6, 4, 4.2, 4.4, 5, 6, 7, 8.1, 9.2, 9.5, 10, 11, 12, (13)]$
- 自定义每公里耗水量为 1 公升
- 自定义线路总长度 13 公里

```
d:\repositories\Algorithm\Instances>python -u "d:\repositories\Alg
orithm\Instances\16.2-4.py"

fewest hydration time is 9, hydrating position sequence is :[1, 2.
6, 4.4, 6, 7, 8.1, 10, 12, 13]
```

经过验算，结果正确。