Peter 笔记

Peterlits Zo

2020 年 7 月 7 日

目录

1 HDU-1358 题解 **3**

2 HDU-1690 题解 **3**

3 CF-1041-E 题解 **3**

4 CF-1023-D 题解 **3**

5 CF-1017-D 题解 **3**

6 CF-1206-D 题解 **3**

7 HDU-6581 题解 **3**

7.1 题干 3

7.2 我的见解 4

8 GoLang 的面向测试驱动开发 **5**

8.1 模仿和入门 5

8.2 更多 5

8.2.1 Benchmarks 5

8.2.2 Examples 5

8.2.3 Skipping 5

8.2.4 Subtests and Sub-benchmarks 5

8.2.5 Main 6

9 LaTeX 中可断开的 vbox **6**

9.1 vsplit 命令 6

9.2 pagegoal 和 pagetotal 命令 6

9.3 示例 6

10 LaTeX 中的宏和超长宏 **7**

11 CF1374E1 题解 **7**

11.1 贪心算法 7

1 HDU-1358 题解

2 HDU-1690 题解

3 CF-1041-E 题解

4 CF-1023-D 题解

5 CF-1017-D 题解

6 CF-1206-D 题解

7 HDU-6581 题解

今天打比赛，感觉脑袋晕晕呼呼的。状态的确有点点不太好。但是好歹算是做出来了。

这道题是 HDU-6581[1]。

7.1 题干

Tom and Jerry are going on a vacation. They are now driving on a one-way road and several cars are in front of them. To be more specific, there are n cars in front of them. The i-th car has a length of l_i, the head of it is s_i from the stop-line, and its maximum velocity is v_i. The car Tom and Jerry are driving is l_0 in length, and s_0 from the stop-line, with a maximum velocity of v_0.

The traffic light has a very long cycle. You can assume that it is always green light. However, since the road is too narrow, no car can get ahead of other cars. Even if your speed can be greater than the car in front of you, you still can only drive at the same speed as the anterior car. But when not affected by the car ahead, the driver will drive at the maximum speed. You can assume that every driver here is very good at driving, so that the distance of adjacent cars can be kept to be 0.

Though Tom and Jerry know that they can pass the stop-line during green light, they still want to know the minimum time they need to pass the stop-line. We say a car passes the stop-line once the head of the car passes it.

Please notice that even after a car passes the stop-line, it still runs on the road, and cannot be overtaken.

Input

This problem contains multiple test cases.

For each test case, the first line contains an integer n ($1 \leq n \leq 10^5, \sum n \leq 2 \times 10^6$), the number of cars.

The next three lines each contains $n+1$ integers, l_i, s_i, v_i ($1 \leq s_i, v_i, l_i \leq 10^9$). It's guaranteed that $s_i \geq s_{i+1} + l_{i+1}, \forall i \in [0, n-1]$.

[1] 原网址是`http://acm.hdu.edu.cn/showproblem.php?pid=6581`。

output

for each test case, output one line containing the answer. your answer will be accepted if its absolute or relative error does not exceed 10^{-6}.

formally, let your answer be a, and the jury's answer is b. Your answer is considered correct if $\frac{|a-b|}{\max(1,|b|)} \leq 10^{-6}$.

The answer is guaranteed to exist.

7.2 我的见解

我认为这个题目学长说应该用贪心算法来做，但是我其实不是很清楚什么是贪心。

我认为对于第 i-th 的车子来说，它能完全过完所谓的 stop-line 的所耗时间，其实取决于两点。第一点，就是它本身到停车线的时间，也就是说应该为 $\frac{s_i}{v_i}$，另外一点，就是需要考虑到前面一辆车从一开始出发到完全越过 stop-line 的时间了，也就是说应该为 $t_{i+1}(s_{i+1}+l_{i+1}) = \max\left(\frac{s_{i+1}+l_{i+1}}{v_{i+1}}, \ldots\right)$[2]。只有从这两点出发都够到线的时候，才是真正的所花最短时间，我将其定义为 $t_i(s_i)$。其中参数 s_i 表示了研究路程的长度。

综上，有关系式

$$t_i(s_i) = \max\left(\frac{s_i}{v_i}, t_{i+1}(l_i+s_{i+1})\right) = \max\left(\frac{s_i}{v_i}, \frac{l_{i+1}+s_{i+1}}{v_{i+1}}, \ldots, \frac{\sum_{g=i}^{k} l_g + s_k}{v_k}, \ldots\right)$$

以上可以刻画对于 s_i 路程的 i-th 的车子而言，它的最短时间。

不妨设置一个结构来表示车子，其中 `l` 是车长，`s` 是车子到停车线的距离，`v` 是车子的速度。这三个变量是由输入提供的，而 `sum` 则表示了 k-th 的车子从出发行进距离 $\sum_{g=i}^{k} l_g + s_k$ 比距离 s_k 所多走的距离，即有 $\sum_{g=i}^{k} l_g$，很明显有，$\texttt{sum}_{i+1} = \texttt{sum}_i + l_{i+1}$。而 $\frac{\texttt{sum}_k + l_k}{v_k} = \frac{\sum_{g=i}^{k} l_g + s_k}{v_k}$

有伪代码:

```
struct Car {
    int l, s, v, sum
}

vector<Car> cars
PER(i, n) cin >> cars[i].l
PER(i, n) cin >> cars[i].s
PER(i, n) cin >> cars[i].v

cars[0].sum = 0
FOR(i, 1, n) cars[i].sum = cars[i-1].sum + cars[i].l

double result = 0.0
PER(i, n) result = max(result, (cars[i].s + cars[i].sum) / cars[i].v)

cout << result
```

[2]这里 t_{i+1} 为一个函数。

8 GoLang 的面向测试驱动开发

本笔记仅仅基于 GoLang 的标准库 `testing` [3]。

该测试库，可以使用 `go test` 来进行测试[4]。

它提供了一个看起来非常 amazing 的接口，即，形如 `TestXxx(*testing.T)` 的函数。看起来和 python 的 TDD 模块特别像，但这是因为 python 的函数也是一个一等对象，可以获取相应的名字。GoLang 身为一个静态编译性语言是如何做到这一点的，我暂且不知。

8.1 模仿和入门

想要新建一个测试文档，首先需要新建一个以 `_test.go` 结尾的文档[5]。

最简单的测试函数如下：

```
func TestAbs(t *testing.T) {
    got := Abs(-1)
    if got != 1 {
        t.Errorf("Abs(-1) = %d; want 1", got)
    }
}
```

在测试函数中，`Error` ，`Fail` 和一些相关的函数都会导致测试失败。

8.2 更多

其实和 python 差不多，但是还有一些 GoLang 的进阶内容。

8.2.1 Benchmarks

形如`BenchmarkXxx(b *testing.B)`的函数会被认为是一个`benchmark`，只有在`go test`使用了`-bench`标志的时候会被顺序运行。

它对于计算运算性能很有帮助。

8.2.2 Examples

使用`Examples`会将标准输出和注释相比较。

8.2.3 Skipping

`*testing.T`支持跳过测试函数。

8.2.4 Subtests and Sub-benchmarks

支持运行子命令（通过一个函数对象）或者子板凳。

[3]官方文档在`https://golang.org/pkg/testing/`

[4]记得之前说了要看 Rust，结果到现在才看来几页而已，不应该呀。

[5]记得之前看 GoLang 的一些相关源代码的时候，的确有很多以这个结尾的文件。没有想到这不仅仅是一个编程习惯，原来它更是一种用来测试的必须的约定。

8.2.5 Main

有的时候，一个主入口也是很有帮助的。

9 LaTeX 中可断开的 vbox

大家都知道 LaTeX 中的盒子是不可被分页算法分割的，除非把一个盒子变成两个分别的盒子[6]。所以一个，递归的、一次断裂的 vbox 的重要性就越来越显著了。

9.1 vsplit 命令

命令\vsplit <number> to <dimen>可以把 <number> 对应和盒子切割成两个部分，第一部分的会输出而第二部分的则会保存到 <number> 对应的盒子。(注，<number> 可能之后会变成空盒子。

如：

```
\setbox 20 = \vsplit 30 to 7in
```

9.2 pagegoal 和 pagetotal 命令

背景知识：vsize 是定义单个页面的主方框的高度，hsize 则是设定主方框的宽度[7]。

命令 pagetotal 提供了当前页面的累计高度。而 pagegoal 则指明了当前页面所需要的高度。pagegoal 在一个页面开始时会检查并赋值自身为 vsize 。

9.3 示例

我们打算分割段落，所以我们打算用 splitable 来制作宏命令。

```
\newbox\splitablebox
\def\splitable#1{%
    \setbox \splitablebox = \vbox{#1}%
    \box \splitable
}
```

目前为止还是中规中矩的一个不能分段的 vbox ，为了实现分段我们也把一个 box 分为两个 box ，分别是 totalbox 和 partialbox 。

```
\newbox\splitableTotalbox
\newbox\splitablePartialBox
\def\splitable#1{%
    \setbox \splitableTotalbox = \vbox{%
        \advance\hsize by -2\fboxsep
        #1%
    }%
    \ifvoid \splitableTotalbox \else \splitableMainLoop \fi
```

[6]本节参考：https://tex.stackexchange.com/questions/20901/breakable-vboxes

[7]但是奇怪的是，我好像没有办法使用 vsize 来设定之后页面的高度，相反，使用 pdfpageheigtht 倒是很有帮助。见页面https://tex.stackexchange.com/questions/299005/automatic-page-size-to-fit-arbitrary-content

```
}
\def\splitableMainLoop{%
    % \pageshrink 是目前页面的所有的胶水。
    \dimen255 = \dimexpr \pagegoal - \pagetotal - \pageshrink - 2\fboxsep \
        relax
    \ifdim\ht\splitableTotalbox < \dimen255
        \noindent\fbox{\box \splitableTotalbox}%
    \else
        \setbox \splitablePartialBox = \vsplit \splitableTotalbox to \dimen255
        \noindent\fbox{\box \splitablePartialBox}%
        \eject % eject 用来强制分页
        \splitableMainLoop
    \fi
}
\splitable{\longtext\longtext\longtext\longtext\longtext\longtext}
```

太棒了，我就是分割 vbox 大师！（话说，有的时候需要加上 `null` ，我也不知道为什么🄵，总之，用递归来实现循环，总感觉怪怪的）。

10 LaTeX 中的宏和超长宏

在我定义本文档的例子命令的时候，出现了一个意想不到的错误：

```
Paragraph ended before \mymacro complete.
```

经过反复试错之后，我发现了在`#n`中对应的组中如果出现了自动或者手动生成的`\par`命令之后才会出现这种错误。看来在`\def`中不支持分段。

最后在 Stack Overflow 中找到了解决方案：使用`\long\def`来代替`\def`。这是因为前面一个宏才会定义出超长宏，而后者只是短宏，不支持组中含有`\par`命令（虽然不知道是怎么检测到的）。

11 CF1374E1 题解

我感到很挫败，没有想到 100 题计划的第一题都这么出乎我的意料。

11.1 贪心算法

没有看懂源代码，但是评论区中有人说和贪心算法有关。所以就先预习一下贪心算法。

贪心算法，总的来说，就是只要每次都选出最优解，那么他就一定是最优解。而相对而言，DP 问题就是子问题不一定是最优解，但是可以让其成为最优解。这个题目我是有想过用动态规划的。但是维护什么书借过什么没有的话，对我来说还是太困难一点了。

下面是 IO 网站说是贪心算法的一个例题。

示例：

在一个月黑风高的暴风雨夜，Farmer John 的牛棚的屋顶、门被吹飞了。好在许多牛正在度假，所以牛棚没有住满。

牛棚一个紧挨着另一个被排成一行，牛就住在里面过夜。有些牛棚里有牛，有些没有。所有的牛棚有相同的宽度。

自门遗失以后，Farmer John 必须尽快在牛棚之前竖立起新的木板。他的新木材供应商将会供应他任何他想要的长度，但是吝啬的供应商只能提供有限数目的木板。Farmer John 想将他购买的木板总长度减到最少。

给出 m, s, c，表示木板最大的数目、牛棚的总数、牛的总数；以及每头牛所在牛棚的编号，请算出拦住所有有牛的牛棚所需木板的最小总长度。

输入格式

一行三个整数 m, s, c，意义如题目描述。接下来 c 行，每行包含一个整数，表示牛所占的牛棚的编号。

输出格式

输出一行一个整数，表示所需木板的最小总长度。

解答：

假设一开始只用一块，那么就是从头到尾一整块，这样虽然满足了要求，但是很明显 $1 \leq m$，不够划算。

那么从这个为起点，那么要选择从什么地方断开就最好了。

既然要断开的话，那么就要选择在中间最大的地方、在两端断开。