Atcoder Beginner Contest 076 解説

問題作成者: E869120 / square1001

2017.10.28

[[For international readers: English editorial starts at page 5.]]

A: Rating Goal / 目標レーティング

もともとのレーティングが a で，パフォーマンスが b の時，レーティングは $0.5(a+b)$ に変動します。高橋君の現在のレーティングが R，目標レーティングが G なので $0.5(R+b)=G$ が成り立てばよいことになります。すると，取るべきパフォーマンスはこの一次方程式の解である $b=2G-R$ であることが分かります。

擬似コード：

```
input R, G
output (2 * G - R)
```

問題 B：Addition and Multiplication / 足し算と掛け算

次のような方法で最終的な整数を最小化することができます。

1. 現在の電光掲示板に書かれている数を x とする。
2. 電光掲示板に書かれている数を「$x+K$ と $2x$ のうち小さい方の数」にする。
3. 1. と 2. を N 回繰り返す。

ここから、なぜこのような方法ができるのか説明します。

整数 a が電光掲示板に書かれているとします。そのとき、次の値は $\min(a+K, 2a)$ となります。また、整数 b が電光掲示板に書かれているとすると、次の値は $\min(b+K, 2b)$ となります。$a<b$ のとき、$a+K<b+K$ でかつ $2a<2b$ となるので、必ず $\min(a+K, 2a)<\min(b+K, 2b)$ となることが分かります。

よって、値が小さいと、次の値も小さくできるので、「各ステップに対して次に書かれる数をできるだけ小さくするようにする」ことを繰り返せば、最終的な数も小さくなります。よって、上の方法で答えを出すことができます。

このような「次の状態の値を最適にすることだけを考える」という方法をコンピューターサイエンスの世界では「貪欲法 (Greedy Algorithm)」といいます。問題によっては貪欲法で最適解が出ない場合もありますが、この問題のように貪欲法で最適解が出せるような問題もあります。知らない人はぜひ知っておくとよいでしょう。

擬似コード：

```
input N, K
number = 1
for i = 1 to N:
    if number * 2 < number + K:
        number := number * 2
    else:
        number := number + K
print number
```

C: Dubious Document 2 / 怪文書 2

$|S| < |T|$ の場合明らかに $|S|$ に $|T|$ を部分文字列として含むことができないので，ここでは $|S| \geq |T|$ の場合について考えます。

まず、文字列 S のどの連続した $|T|$ 文字が文字列 T と一致するか，というのを見つけることを考えます。文字列 S' のある連続した $|T|$ 文字を X' とし，X (X' に対応する S の区間) と T を一致させることができるか，すなわち X' の '?' の部分を適切に埋めることで T と一致させることができるかを判定することになります。

同じ長さの文字列 A と B (長さを L とする) が一致するのは，「1 以上 L 以下の整数 i 全てに対して，(A の i 文字目) = (B の i 文字目) が成立する」と同じであり，つまり同じ「i 文字目」の文字の関係だけについて考えればよさそうです。

つまり，(X' の i 文字目) と (T の i 文字目) の関係が重要です。それぞれの文字を x', t と置くことにして，次の 3 パターンに分けることができます。

a. $x' = t$ のとき：これは明らかに一致します。

b. x' が '?' ではなく，$x' \neq t$ のとき：これは明らかに一致させることができません。

c. x' が '?' のとき：'?' の部分を t にすれば，一致させることができます。

すなわち，「X を T と一致させることができる」とは，すべての i に対して a. または c. の関係が成り立っていることで，もし 1 つでも b. の関係があればその文字は一致することができないので X と T を一致させることができません。

一致させることが出来る場合、S' の選んだ区間に入っていない部分の '?' を全部 'a' に変えることで、選ぶ区間が確定しているという条件の中では辞書順最小が達成できます。

選べる区間は $|S| - |T| + 1$ 個あるので，それらの区間すべてに対して「選んだ区間が決定している状態での辞書順最小」を求め，その中での辞書順最小なものが，全部の中の辞書順最小となります。また，この $|S| - |T| + 1$ 個の区間に対して全部一致させることができない場合は，どうやっても一致させることができないということになります。

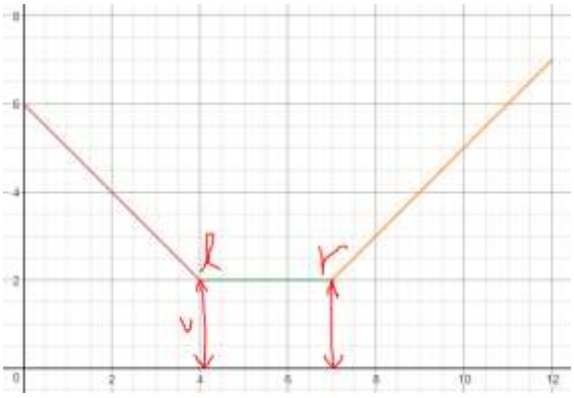
よって，計算時間は $O\big(|T|(|S'| - |T| + 1)\big) = O(|S'||T|)$ となります。

問題 D：Atcoder Express / Atcoder 特急

まず，最初と最後の速度の制限がなく，制限が「加速度が $\pm 1\ [m/s^2]$ 以内で，l 秒から r 秒までの間の制限速度が $v\ [m/s]$」だけであったとします。そのとき，x 秒後に出しうる最大の速度は，次のようになります（ただし，最初から最後までの時間を T とします）：

- $0 \leq x \leq l$ のとき $v + (l - x)\ [m/s]$
- $l \leq x \leq r$ のとき $v\ [m/s]$
- $r \leq x \leq T$ のとき $v + (x - r)\ [m/s]$

グラフで表すと，右図のようになります。横軸が時間で，縦軸が出しうる最大の速度です。



そのような条件が N 個あり，また，最初と最後の速度を $0\ [m/s]$ にしなければならないという条件もあります。これらは，先ほどの条件の次のような場合です：

- 最初の速度が $0\ [m/s]$：$l = 0, r = 0, v = 0$ の場合
- 最後の速度が $0\ [m/s]$：$l = T, r = T, v = 0$ の場合

そこで，同じような条件が $N + 2$ 個できました。これらを条件 $1, 2, 3, \ldots, N + 2$ と番号付けするとします。関数 $f_i(x)$ を条件 i に対して出しうる最大の速度とすると，x 秒後に出しうる最大の速度は $\min\{f_1(x), f_2(x), f_3(x), \ldots, f_{N+2}(x)\}$ となります。すなわち，グラフを $N + 2$ 個の条件について重ねてみたときに一番下の部分のグラフ（これをここからは「グラフ」と言うことにする），ということになります。

そこで，入力値が整数なので，グラフの傾きが変わる x は，必ず 0.5 の倍数になります。条件式は必ず $y = -x + c$，$y = c$，$y = x + c$（c は整数）のいずれかになるので，どの 2 つの式の組（ただし傾きが異なるもの）について連立させて解いても x の解が 0.5 の倍数になることによって証明できます。

そこで，$v(x) = \min\{f_1(x), f_2(x), f_3(x), \ldots, f_{N+2}(x)\}$ を 0.5 間隔で求めることを考えます。a を 0.5 の倍数としたとき，グラフの a と $a + 0.5$ の範囲と，$x = a, x = a + 0.5, y = 0$ が作る四角形は，a と $a + 0.5$ の範囲でのグラフの傾きが変わらないことから，台形であることがわかります。また，この区間で動く距離は，この面積に等しいので，$0.25\big(v(a) + v(a + 0.5)\big)$ である。この合計を求めれば，走れる最長の長さを求めることができます。

a は $2T$ 通りの場合しか取りえず，各 a に対しても $v(a)$ を求めるのが $O(N)$ でできるので，計算量 $O(NT)$ で解くことができる。$N \leq 100, T \leq 20000$ なので実行時間制限に間に合わせることができます。

Atcoder Beginner Contest 076 Editorial

Problem Setter: E869120 / square1001

2017.10.28

A: Rating Goal

Let a the rating, and let b the performance of the next contest. In this case, the rating will change to $0.5(a+b)$.

Takahashi's current rating is R, and his rating goal is G. So, he wants to establish the equation $0.5(R+b)=G$. Solving this linear equation, you can get $b=2G-R$. Therefore, Chokudai has to get performance $2G-R$.

Pseudocode:

```
input R, G
output (2 * G - R)
```

Problem B: Addition and Multiplication

You can minimize the final value in following way:

Let a the current value on the board. If you rewrite to $\min(a+K, 2a)$, the value at the end of this step will be minimized.

If you continue doing this, the final value will be minimized. That's because, for two values a and b, if you do the rewriting operation, in the end of this operation the value will be $\min(a+K, 2a)$ and $\min(b+K, 2b)$. If $a < b$ it is definitely $\min(a+K, 2a) < \min(b+K, 2b)$, so the higher initial value, the higher final value.

This type of algorithm is called "greedy algorithm". Greedy algorithm is like "only considering goodness of the next step". In this problem greedy algorithm works optimally, but in some problem greedy algorithm doesn't work optimally.

Pseudocode:

```
input N, K
number = 1
for i = 1 to N:
    if number * 2 < number + K:
        number := number * 2
    else:
        number := number + K
output number
```

C: Dubious Document 2

In the case of $|S| < |T|$, obviously, $|T|$ cannot be existed as a substring of $|S|$. So now we think about the case of $|S| \geq |T|$.

First, you have to search which substring in $|S|$ matches $|T|$. Now, let's think about how to check "does $|T|$ can appear in the substring between i-th character and $(i + |T| - 1)$-th character?".

Let X' the substring of S' between i-th character and $(i + |T| - 1)$-th character. You have to check whether it is possible to match X' to T with replacing character '?' in X' to an arbitrary small-case English alphabet. You should only see the relationship between $x' =$ (i-th character of X') and $t =$ (i-th character of T), for each integer i which is between 1 and $|S| - |T| + 1$. The relation of x' and t can divide into three following patterns:

a. Case $x' = t$: you can match x' to t, obviously.
b. Case $x' \neq t$ and x' is not '?': you can't match x' to t because you can't change the character of x.
c. Case x' is '?': you can match x' to t by changing '?' to t.

In order to establish $X = T$ (let X the substring of S which corresponds to X'), you are not allowed to make an index which is $X_i \neq T_i$. Therefore, if there at least one index which is in "case b.", it is not possible to match X to T. Otherwise, it is possible.

If it is possible to match X to T, you can make S lexicographically smallest, in the condition of "T appears in selected interval in S", with replacing '?' which is not in selected interval to 'a'.

The answer is the lexicography smallest one, among "the lexicographically smallest S which T appears in selected interval in S" for all length-$|T|$ intervals in S. If there is no interval which is possible to match X to T, there is no answer for S.

Since there are $|S| - |T| + 1$ intervals to select, so you can solve this problem in $O\big(|T|(|S| - |T| + 1)\big) = O(|T||S|)$ time complexity.

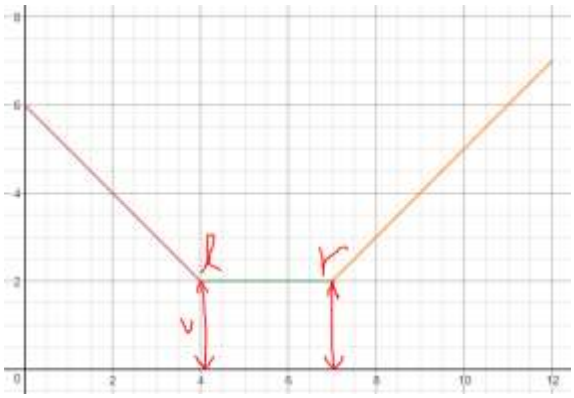
Problem D: Atcoder Express

Think about the situation that there are no speed restrictions about speed at start and at end, and the only restrictions are “the acceleration speed is between -1.0 m/s^2 and 1.0 m/s^2, and the maximum speed in l seconds from start to r seconds from start is v meters per second”. And let T the time from start to end in seconds.

The maximum possible speed at x seconds from start is following:

- If $0 \leq x \leq l$, the maximum possible speed is $v + (l - x)$ meters per second.
- If $l \leq x \leq r$, the maximum possible speed is v meters per second.
- If $r \leq x \leq T$, the maximum possible speed is $v + (x - r)$ meters per second.

The relation graph between the time and maximum possible speed is like the figure on the right. The horizontal axis represents the time and the vertical axis represents the maximum possible speed.



Since there are N intervals, you can make N conditions which forms “the acceleration speed is between -1.0 to 1.0 m/s^2, the maximum speed in l seconds from start to r seconds from start is v meters per second”.

You also have the restriction that “the speed at start point is zero” and “the speed at end point is zero”. You can translate to condition which forms same as other N conditions:

- The speed at start point is zero: the case $l = 0, r = 0, v = 0$
- The speed at end point is zero: the case $l = T, r = T, v = 0$

Now you have $N + 2$ conditions, and that’s all – you need no more conditions.

Let $f_i(x)$ the maximum possible speed in i-th condition at x seconds from start.

The maximum possible speed at x seconds from start (let this speed $v(x)$) is, $v(x) = \min\left(f_1(x), f_2(x), f_3(x), \ldots, f_{N+2}(x)\right)$

If you make the relation graph between t and $y = v(t)$, the slope of the graph doesn’t change between $t = a$ and $t = a + 0.5$, if a is multiple of 0.5. So, the quadrilateral which is formed by the graph and lines $t = a, t = a + 0.5$, and $y = 0$ forms a trapezoid which $t = a$ and $t = a + 0.5$ are parallel.

Since the maximum possible total moving distance is equal to the area which is formed by the graph and $y = 0$, you can calculate the maximum possible total

moving distance by summing up the area of these trapezoids. And, the area of a trapezoid of interval $[a, a+0.5]$ is $0.25\big(v(a)+v(a+0.5)\big)$.

There are $2T$ intervals of $[a, a+0.5]$ because of $0 \leq a, a+0.5 < T$ and "a is multiple of 0.5". And you can calculate the value of $v(x)$ in $O(N)$.

Therefore, you can calculate the maximum possible total moving distance in $O(NT)$. Since $N \leq 100$ and $T \leq 20000$, you can write a code which is fast enough for time limit.