第３回　ドワンゴからの挑戦状　予選問題

A. 動画検索

問題概要

- n 本の動画があり、そのうちタイトルに
  - 「ドワンゴ」を含むものが a 本
  - 「ニコニコ」を含むものが b 本
- 両方を含むものは最低何本あるか？

解法

- $a+b \leqq n$ のとき
  両方含む動画は存在しないことがある→答えは 0

- そうでないとき

  どちらかを含む動画の総数

  - 両方含む動画が x 本あるとすると $\boxed{a+b-x} \leqq n$

  - つまり $x \geqq n-(a+b)$ でなければならない
    → 答えは $n-(a+b)$

第３回　ドワンゴからの挑戦状　予選問題

B. ニコニコレベル

問題概要

- 数字と ? からなる文字列 T が与えられる
- ? を数字に置き換えて数字列 T’ を作る
- T’ ができるだけ長いニコニコ文字列を含むように

”", ”25”, ”2525”, ”252525”, ...

T = 12??567890

↓

T’ = 1252567890

制約

・ T の長さは 10^5文字 以下

↓

置き換え方をすべて試すわけにはいかない

↓

どのように置き換えれば

長いニコニコ文字列が作れるか？ を考える

?の置き換え方

- まず ? を 2,5 以外に置き換える意味はない
- 2525… というのを作りたいので
- 2 の次には 5、5 の次には 2 が出てきてほしい

  →奇数文字目を 2、偶数文字目を 5 にすると？

252525252525

T = 1?2??6??2?2?

T’ = 152526252525

長いのが作れそう！

解法

- 奇数文字目を 2、偶数文字目を 5 にする　または
  奇数文字目を 5、偶数文字目を 2 にする

- 上の **2 通りの置き換え方だけ**を考えれば実は十分

  理由: T'　内に現れるニコニコ文字列は必ず

  「奇数文字目が 2 で偶数文字目が 5」または
  「奇数文字目が 5 で偶数文字目が 2」になっているから

- よって **2 通りの置き換え方を試し、それぞれについて現れるニコニコ文字列の長さを求める**ことで解ける

c. スキーリフトの相乗り

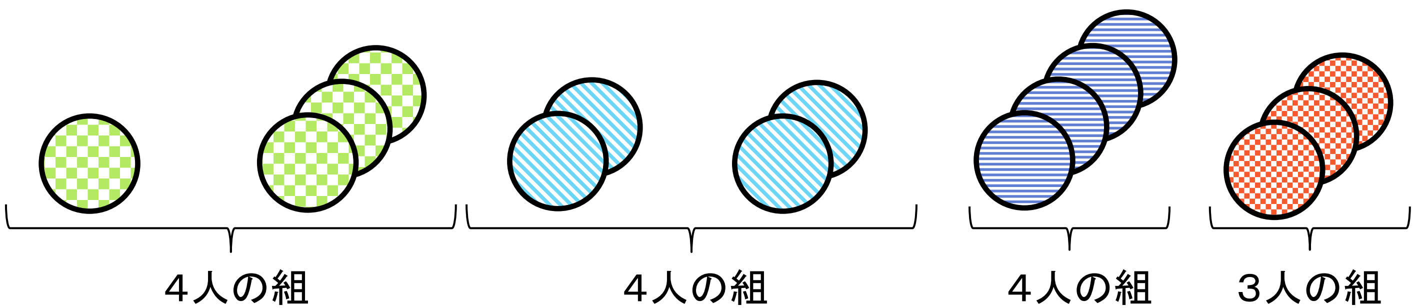
問題概要

- スキーリフトに並ぶグループの人数（1～4人）がリストで与えられる
  - （リストのサイズN <=10,000）
- 搬器（椅子）は1台あたり4人乗り
- タカシくんは、先頭に並ぶグループを搬器に案内した後、定員を超えない範囲で、任意のグループを同じ搬器に相乗りするよう案内できる
- 最短何台目の搬器で全グループを運びきれるかを求めてください

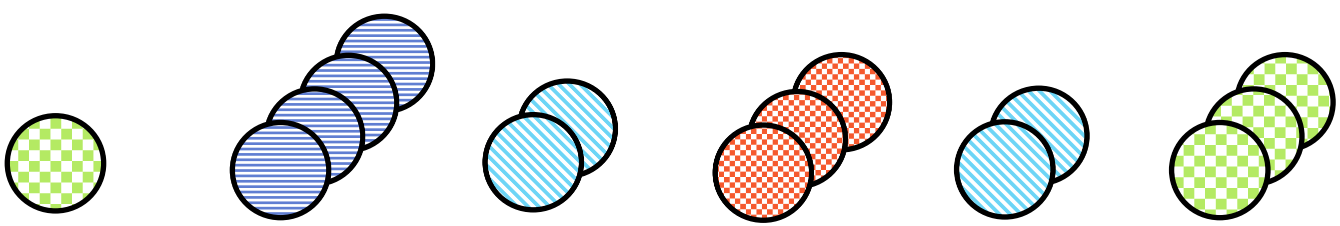
「先頭から順に案内」に対する考察

- スキーリフトに並ぶグループの順番は無視して良い
  - 同じ搬器に乗るグループ達（組と呼ぶ）を最適に決められれば、タカシくんは組分けに基づき最適な案内が可能

入力が1，3，2，2，4，3の時、4人、4人、4人、3人の組（ここでは色で表す）が最適



順番が変わっても、上と同様の組分けで案内すれば最適である

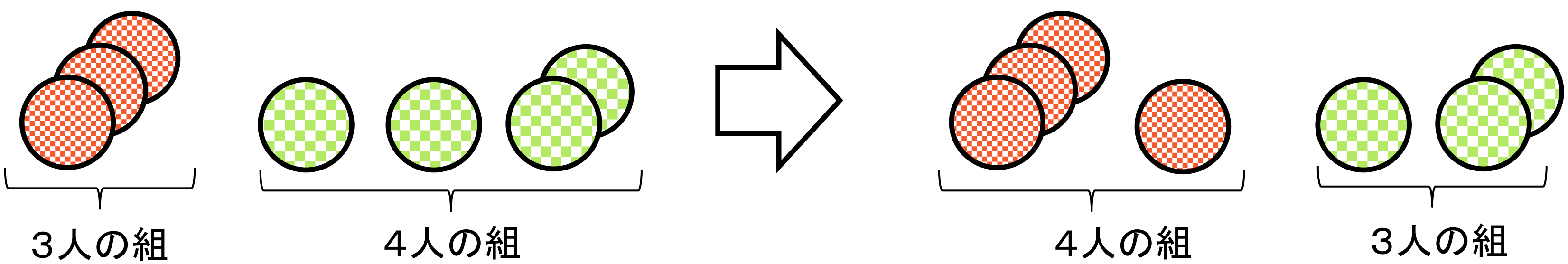


最適な組分けの作り方

- ４人グループはそのままリフトに案内すればOK
- ３人グループは、極力１人グループと組ませて良い（証明は次ページ）
- ３人グループと１人グループを極力組ませると…
  - ２人グループ・１人グループが残るケース
    - ２人グループ同士を極力組ませ、残りを１人グループで埋めると、ceil（残り人数/4）組の最適な組分けが可能。
  - ３人グループ・２人グループが残るケース
    - ２人グループ同士を極力組ませることしかできない。最適な組数は３人グループの数＋ceil（２人グループの数/2）

３人グループの取扱いに関する証明

とある最適な組分けで、
「１人グループと組んでいない３人グループ」と、
「３人グループと組んでいない１人グループ」が存在すると仮定。
この時、１人グループを所属する組から引き抜き、
３人グループと組ませても、合計の組数は変わらない



このことから、
３人グループの組を極力１人グループと組ませても、
最適性は失われない

想定解法

- 各人数のグループが幾つかを数えて計算
  → O(N)

- リストをソートした上で貪欲に処理、
  という考え方でも正答になりうる
  → O(N log(N))

第３回　ドワンゴからの挑戦状　予選問題

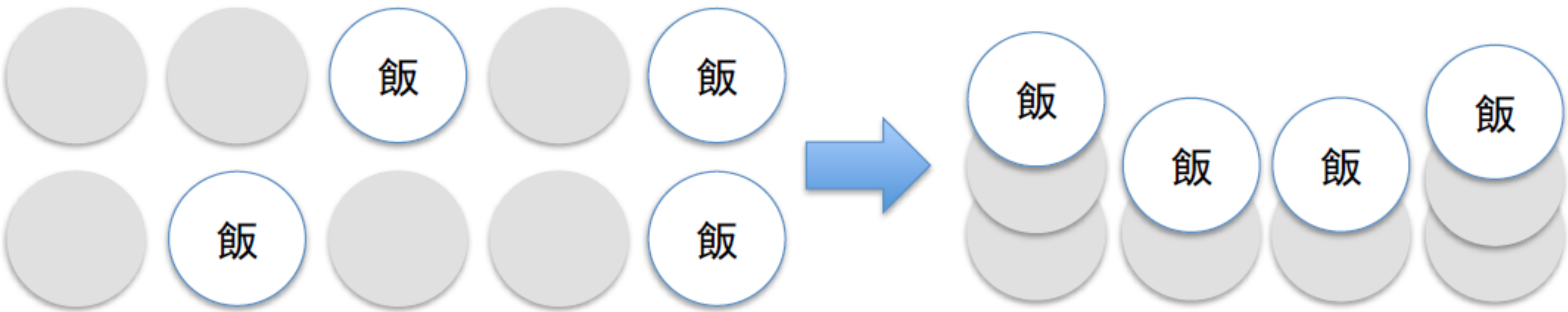
D.ネタだけ食べたい寿司

問題概要

- 回転寿司で回ってくる寿司のリスト(N)が与えられる
- 以下の条件を踏まえて幸福度を最大化する。
  - ネタだけ食べた方が寿司として食べた時よりも幸福度が高い
  - 流れてくる順番に皿をとり食べる必要がある
  - 食べ終わった皿を横に置けるスペースが限られている(M皿)
  - 縦方向には無限に皿を重ねられる
  - ネタだけ食べた皿の上に皿を重ねられない
- 1 <= N, M <= 100,000

考察

- 最後にネタだけ食べる皿を固定して考える
  - 残りのネタだけ食べる皿は左にあるどの組み合わせでも良い(順番に重ねていけば条件を満たす)



  - 左の皿の中で幸福度の差分が大きいM-1皿を選べば(i皿目で食べ終わった時の)幸福度が最大化できる

想定解法

- 幸福度の差分上位M-1個とその合計値を管理しながら左から走査していく
  - priority_queueなどを用いて管理する(管理してる中で一番差分が小さい値よりも新しい値が大きければ入れ替える)
- i皿目で食べ終わった時の最大の幸福度は以下の式で求められる
  - (i皿目をネタで食べた幸福度) + (i-1皿目までを寿司で食べた幸福度) + (i-1皿までの上位M-1皿の幸福度の差分)
- O(NlogM)

E.偶奇飴分け

問題概要

- N+2 個の皿が並んでいて、皿 1～N には飴がある
- 皿 i に飴が A[i] 個あり、以下の操作を行う:

操作 皿 1～N それぞれについて乗っている飴を すべて一斉に左右どちらかの皿に移す

- 飴の乗っていない皿を取り除いた後、
  奇数番目の皿に乗っている飴の総数を最大化
- A[i]を書き換えるクエリが Q 個来るので毎回答える

部分点解法

- まずクエリを気にせず 1 回だけ解く場合を考える
- これは以下のような状態を持つ DP で解ける

dp[n][p][q][s] :=

n 個目の皿まで左右を決め終わっていて
n-1個目とn個目の皿の左右がそれぞれ p, q で
n+1個目以降の皿を無視して飴を移動したとき
残る最後の皿の番号の偶奇が s のときの
(奇数番目の皿の飴数)-(偶数番目の皿の飴数)
の最大値

※要は「末尾2個の左右」と「最後の皿の偶奇」を持つ

満点解法の基本方針

- クエリに答えるためには先ほどの DP を適切なデータ構造上に乗せる必要がある

- ある皿の区間に対して、末尾だけでなく
「先頭2個の左右」「最初の皿の偶奇」
「末尾2個の左右」「最後の皿の偶奇」
をキーにした先程の DP の値を考える

- あとはこれを segment tree や平方分割で扱う

実装するに向けて

- 前述の DP をそのまま実装しようとすると
部分点解法の場合ですら遷移がとても煩雑になる

- 実装前に少し考察をして実装をシンプルにできる

この状態から皿 n+1 の飴を
r（左or右）に移すと決めたとき

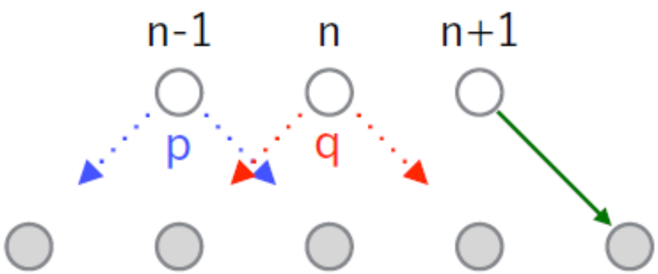
dp[n][p][q][s]

DP の値の変化
がどうなるか

最後の皿の偶奇の変化と

dp[n+1][q][r][s']

最後の皿の偶奇の変化

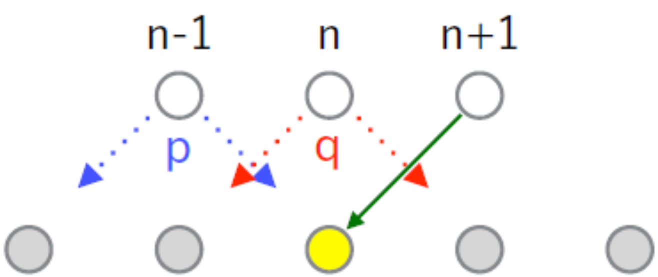


r=右なら

行き先の皿にはまだ飴がないので

必ず s’ = -s

（奇数,偶数を1,-1で表した場合）



r=左のときは

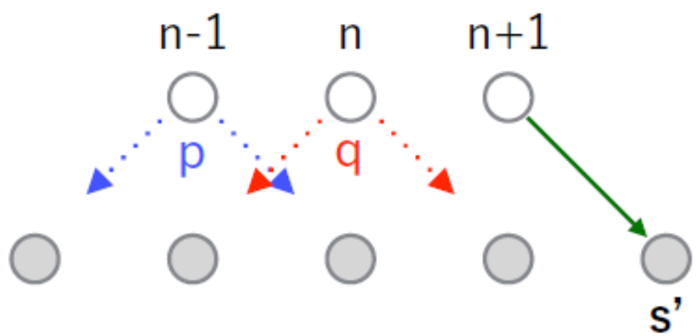
黄色く示した皿に既に飴があるか

だけが問題

⇒ p=右なら s’ = s

p=左なら s’ = -s

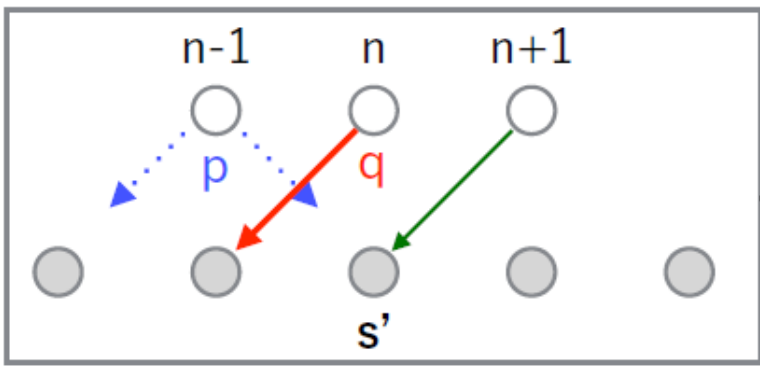
DP の値の変化



r=右なら

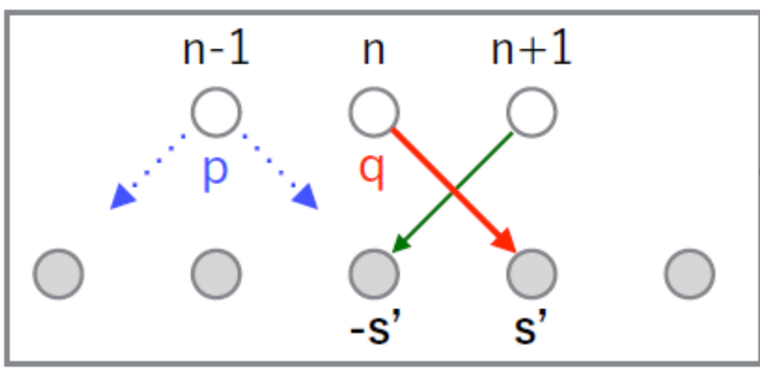
A[n+1]×s’ だけ変化

(s’ は奇数,偶数を1,-1で表しているため)



r=左のときは

q=左なら A[n+1]×s’



q=右なら A[n+1]×-s’

p=左なら更に -2A[n]×s’

(皿 n の飴の行き先の偶奇が変化するため)

まとめ

- 前述の考察により、8 状態からの 2 通りの遷移を
  - 状態の 2 通りの変化
  - 値の 3 通りの変化

  に分解することができる

（他の解法）

遷移方法の考察や実装量の落とし方は、この解説に書かれている方法のみではなく、区別するべき状態を考察することで、**状態のとり方自体を扱いやすいように変える**といった工夫も可能である。