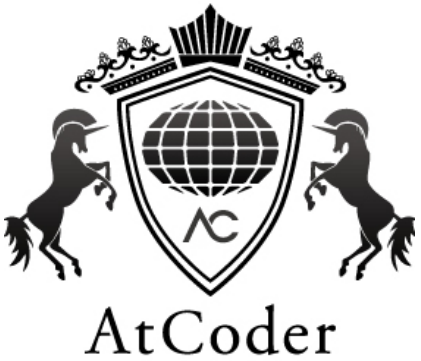
AtCoder Regular Contest 056
解説



AtCoder株式会社 代表取締役
高橋 直大

競技プログラミングを始める前に

AtCoder

- 競技プログラミングをやったことがない人へ
  - まずはこっちのスライドを見よう！
  - http://www.slideshare.net/chokudai/abc004

- 作題・準備:hogloid

A問題　みんなでワイワイみかん

1. 問題概要
2. アルゴリズム

A問題　問題概要

AtCoder

- みかんを少なくともK個買いたい
- みかんは1個A円、またはL個B円で買うことができる
- 払うお金の最小値は？
- 制約
  - $1 \leqq A, B, K \leqq 10^9$
  - $2 \leqq L \leqq 10^9$
  - $B \leqq A*L$

A問題　アルゴリズム

AtCoder

- B ≦A*L の条件より、1個ずつA円でL個買うならセットで買ったほうがお得
- よって、1個ずつ買うのはL個未満
- このことから、考えられるパターンは
  - K/L(切り捨て)回L個のセットを買う+K mod L回1個ずつ買う
  - K/L+1回L個のセットを買い L - K mod L 個余らせる
- のどちらか

B問題　駐車場

1. 問題概要
2. アルゴリズム

B問題　問題概要

AtCoder

- N頂点M辺の無向グラフが与えられ、ある始点Sが定まっている
- 頂点1からNまで順番に、
  - 始点から頂点iまで到達できるとき、頂点iを削除
  - 到達できないとき、何もしない
- 最終的に削除される頂点の番号を昇順に出力
- 制約
  - $1 \leqq N, M \leqq 200000$

B問題　アルゴリズム

AtCoder

- 部分点:
  - 毎回問題文通りの操作を実装すれば大丈夫
  - 必要な道具は、
    - 2頂点が辺でつながっているか判定
      - UnionFind/dfsなどを用いましょう
    - 頂点を削除
      - 有効かどうかの状態を頂点ごとに持っておきましょう
- O(MN)

B問題　アルゴリズム

AtCoder

- 始点からある頂点まで到達できない場合、何もしない、とあるが、始点から到達できない以上、削除してもそれ以後の答えに影響ない
- すなわち、以下のような問題と考えることができる
- 頂点1からNまで順番に、
  - 始点から頂点iまで到達できるとき、答えにiを追加
  - 頂点iを削除
- 頂点iまで到達できるか判定しているとき、番号i以上の頂点のみ残っている
- すなわち、i以上の頂点のみを通り始点から頂点iまで到達できる ことと、iが答えに含まれることは等価

B問題　アルゴリズム

AtCoder

- Sからある頂点iまでのパスのうち、通る頂点番号最小のものの最大値を$cost_i$とおく
  - すなわち、$cost_i$より大きい番号の頂点のみ使ってiに到達することはできないような数のうち最小のもの
- これは、ダイクストラのように$cost_i$が大きい頂点から決めていけばOK
  - $cost_i$は$cost_j \leqq cost_i$なる頂点jを使ってさらに小さくなることはないため、大きい頂点から決めていくことができる
- O(MlogN)

C問題　部門分け

1. 問題概要
2. アルゴリズム

C問題　問題概要

AtCoder

- N*Nの重み付き無向グラフが与えられる
- 頂点を彩色する
- 使われた色の数*K-異なる色の間の辺の重みの総和 をスコアとするとき、スコアの最大値はいくらか
- 制約
  - $1 \leqq w_{i,j} \leqq 100000$
  - $1 \leqq N \leqq 17$
  - $1 \leqq K \leqq 100000$

C問題　アルゴリズム

AtCoder

- 部分点:
  - 色は1以上N以下としていいので、頂点の彩色を全て試していく
  - 愚直に試すとO(N^N)となってしまうので、少し工夫
  - 頂点を番号が増える順番で塗っていくことにする
  - 頂点iを今までに使われていない色で塗るとき、1種類だけ試せば十分
    - それ以後の塗り分けの色を交換することで、同じスコアを達成できるから
  - これで、i番目の頂点での分岐が高々i個となり、塗り方は高々O(N!)通り

C問題　アルゴリズム

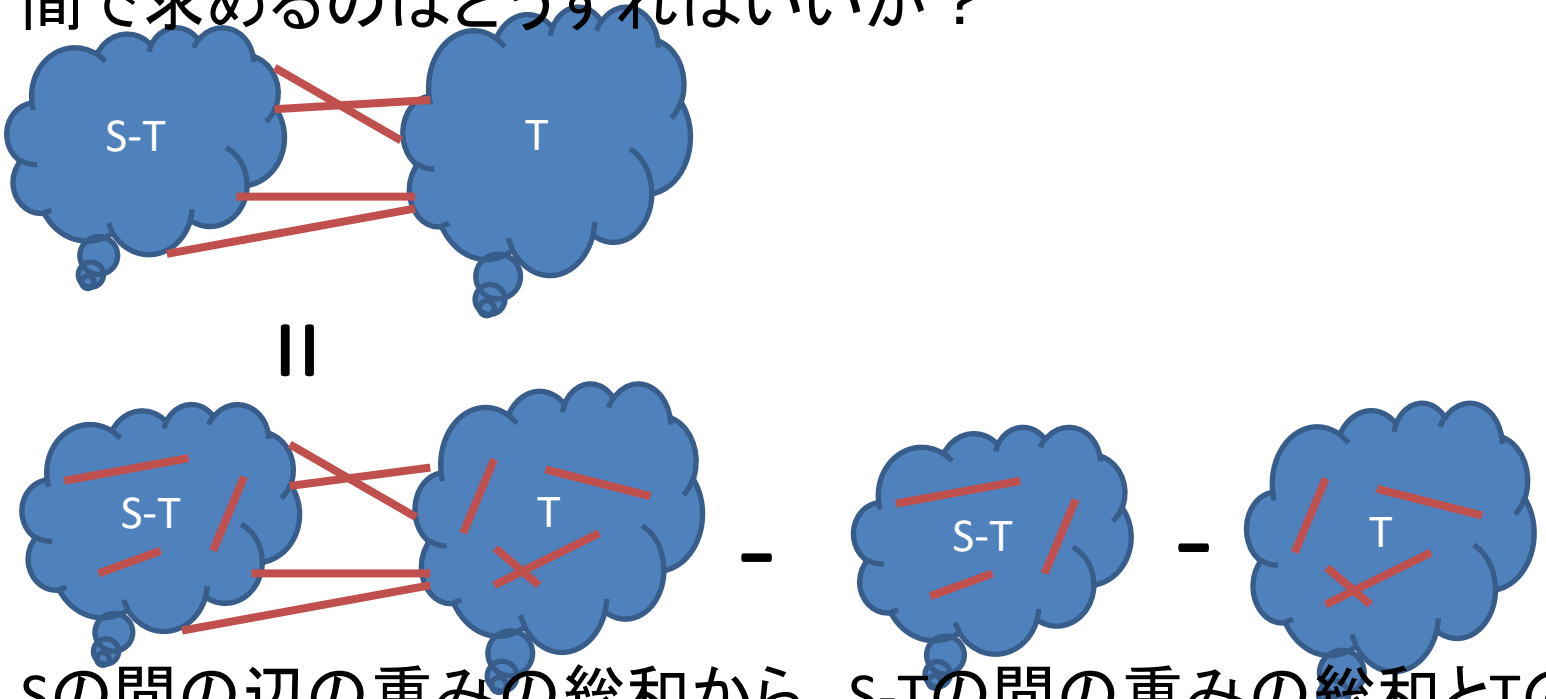
AtCoder

- dp[S]:=頂点集合Sのみに対する問題の答え、というDPを考えてみよう
- Sのうち、ある1つの色で塗られる集合Tを固定すると、そのスコアは
  dp[S-T]+K-(S-TとTの間の辺の重みの総和) となる
  - S-TはSのうちTに含まれない頂点
  - Tの頂点たちは他の色と異なる色で塗られることが決まったので、残りのS-Tのスコアは独立に決めることができる
- この計算量はO(3^N)
  - 全ての頂点集合(S)に対し、その全ての部分集合(T)を試す、ということは、N個の頂点をT、S-T、残りに分ける方法を試していることと等価
    - 後者はN個の頂点を3つに分けているので、3^N

C問題　アルゴリズム

AtCoder

- DPの遷移のうち、(S-TとTの間の辺の重みの総和) を定数時間で求めるのはどうすればいいか？



- Sの間の辺の重みの総和から、S-Tの間の重みの総和とTの間の重みの総和を引けばOK
  - ある部分集合の中の重みの総和は予め計算しておく(O(2^N))

D問題　サケノミ

1. 問題概要
2. 考察
3. アルゴリズム

D問題　問題概要

AtCoder

- N種類のドリンクとN種類のグラスがある
- それぞれのドリンクiは、対応するグラスiにM_i回決まった時間に補充される
- ドリンクには美味しさが決まっている
- 行える行動は、好きな時刻に注がれているドリンクを全て飲むことのみ
- 飲んだ美味しさの総和の最大値を求めよ
- 制約
  - $1 \leqq N \leqq 500{,}000$
  - $0 \leqq \sum M_i \leqq 500{,}000$

D問題　考察

AtCoder

- 部分点解法:
  - dp[i]:=時刻iで最後にドリンクを飲み干したときの美味しさの最大値、というDPを考えてみよう
  - 次にドリンクを飲み干す時刻jを全て試し、次に飲み干すときの美味しさの総和$d_{i,j}$を求め、dp[j]:=max(dp[j],dp[i]+$d_{i,j}$)と更新していく
    - i以後でそれぞれのドリンクが最初に補充される時刻を最初に計算し、jがその時刻を超えるごとに$d_{i,j}$に加えていくと上手くいく
  - O(N*max(t))

D問題　考察

AtCoder

- 満点解法
  - 上記のDPが高速にならないか考えてみる
  - 配るDPだったのを、もらうDPに書き直す
  - すなわち、iを昇順に、その中でjをj<iの範囲で昇順に走査し、dp[i]:=max(dp[i],dp[j]+$d_{j,i}$) で更新していく
  - dp[j]+$d_{j,i}$を1つの項として見てみる(dp2[j]と呼ぼう)
  - 時刻iで美味しさw_kのドリンクkが注がれ、ドリンクkがその前に注がれた時刻をlと置くと、l<j<iなるjに対し、dp2[j]:=dp2[j]+w_k となる
    - 最後に飲んだ時刻がjで、次に飲む時刻がi以後のとき、ドリンクkを新たに飲むことになるから
  - これは単純な区間add

D問題　考察

AtCoder

- dp2を使うと、dp[i]=max{dp2[j]|j<i}
  - 単純な区間max
    - 実際は、iを昇順に見ていくので、初期値を負の大きな数にしておけば、全体のmaxを取ってもOK
- よって、区間add区間maxの機能をもつセグメント木などを実装すればいいことが分かる
- O((N+∑M)log(max(t))