コピー ＆ ペースト 3　解説

松尾 凜太朗 /QCFium

問題概要

x, y は最初空文字列。x を S に等しくしたい。

できる操作

A) x の末尾に任意の 1 文字追加

B) y を x に等しくし、x を空文字列にする

C) x の末尾に y を結合

それぞれ A, B, C のコストがかかるとき、S を入力するためにかかるコストの最小値は？

考察1

できる操作

A) x の末尾に任意の 1 文字追加

B) y を x に等しくし、x を空文字列にする

C) x の末尾に y を結合

どう考えても

- 操作 A で入力する文字は S に含まれる文字のみ
- x,y を S の長さ以上にするのは無駄

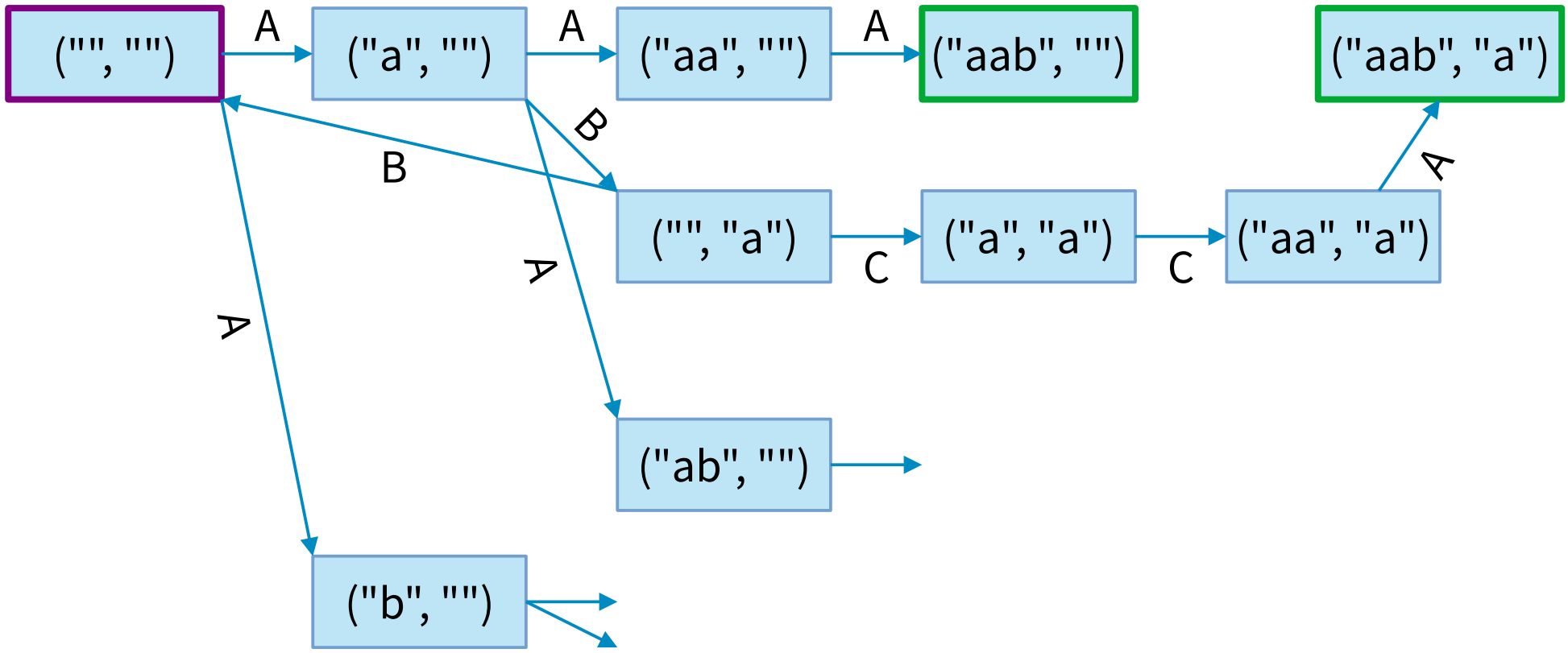
小課題 1

制約 : N = 3

N が小さい → x,y の状態を全て管理してダイクストラ法
あり得る x,y の種類はそれぞれ 1+3+9+27=40(S が全て異なる文字の場合)
40×40 頂点のグラフでダイクストラ法

小課題 1

例 ：S = ″aab″



小課題１

頂点数は 40×40 以下

各頂点からは
- 操作Ａの辺が高々３本
- 操作Ｂの辺が１本
- 操作Ｃの辺が高々１本

→ 間に合う

小課題２

制約：Ｓは’a’のみからなる

x,y として考えられるのはそれぞれ O(N) 個
→　最短経路を求めるグラフの頂点数は O(N^2)

各頂点からでる辺の本数は多くとも

- 操作Ａの辺が１本
- 操作Ｂ, Ｃも１本ずつ

→ O(N^2 log(N))

考察2

できる操作

A) xの末尾に任意の1文字追加

B) yをxに等しくし、xを空文字列にする

C) xの末尾にyを結合

xは空文字列 → yを捨てるのは無駄
→ 切り取り直後のyはxの部分文字列になる
→ yはSの(連続する)部分文字列になるとしてよい

任意の時点でのxもSの(連続する)部分文字列になるとしてよい

小課題 3

制約 : N ≦ 30

x, y は S の連続する部分列なので O(N^2) 個

少し工夫して辺を張れば最短経路を求めるグラフ (頂点数 O(N^4) 辺数 O(N^5)) を O(N^6) で構築できる

- 操作 A
  各 x について x+c で表すことのできる S の部分文字列の位置を前計算 (O(N^4)) しておけばよい
- 操作 B
  頂点ごとに S の部分文字列で x+y に等しいものを検索
  (愚直にやって O(N^6))

考察3

できる操作

A) x の末尾に任意の 1 文字追加

B) y を x に等しくし、x を空文字列にする

C) x の末尾に y を結合

グラフを作って最短経路を求めるのは限界そう

((A,C を何回か) → B) の繰り返しと考える
S[l:r] : S の l 番目から r−1 番目までの部分文字列 とする
dp[l][r] : 操作 B の直後で x=″″,y=S[l:r] である状態にするまでの最小コスト
r−l の昇順に遷移していけばよい
dp[s][t] から dp[u][v] への遷移は？

考察3

できる操作

A) x の末尾に任意の 1 文字追加

B) y を x に等しくし、x を空文字列にする

C) x の末尾に y を結合

S[s:t] : bba

S[u:v] : aabbaabbab

A(t-s) < C だと S[s:t] をクリップボードにコピーした意味がない
A(t-s) ≧ C ならできる限りクリップボードを使う回数を多くしたい
→ 前から貪欲に貼り付けを使っていく

考察3

クリップボード ：bab

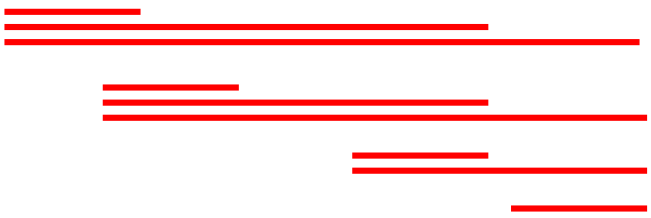
abababaababbab

太青線部は貼り付け、それ以外は
1文字入力するのが最適

考察3

クリップボード ：bab

abababaababbbab



赤線（遷移先）が多すぎるので左のような赤線のみへの遷移を考える
自分と被らない次の青線に移ることを繰り返して更新可能

考察3

クリップボード ：bab

abababaababbab

はみ出した分は１文字入力するしかない
→ 赤線に遷移した後左右に１伸ばすのをそれぞれコストＡで自由にできることにすればよい

考察3

最初に全ての s,t について S[s:t]=S[i:i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の各要素 i について、 a[s,t] に含まれ j ≧ i+t−s を満たす最小の要素 j を前計算する

dp[s][t] からの遷移は

- dp[s][t+1] にコスト A で遷移 ：S[s:t] に遷移した過程の一番最後に操作 A で S[t] を入力したことにする
- dp[s−1][t] にコスト A で遷移 ：S[s:t] に遷移した過程の一番最初に操作 A で S[s−1] を入力したことにする
- a[s,t] に含まれる全ての i について ：
  a[s,t] のうち i+(t−s) 以上の最小のものを辿っていき、更新

考察3

最初に全ての s,t について S[s:t]=S[i,i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の各要素 i について、 a[s,t] に含まれ j ≧ i+t−s を満たす最小の要素 j を前計算する

全部分文字列の Rolling Hash の結果を map に入れれば O(N^2 log(N))

考察3

最初に全ての s,t について S[s:t]=S[i,i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の要素 j で j ≧t を満たす最小のものも前計算しておく

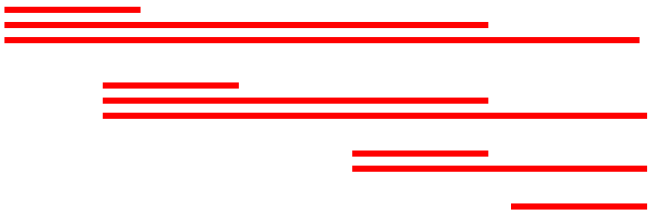
dp[s][t] からの遷移は

- dp[s][t+1] にコスト A で遷移 ：S[s:t] に遷移した過程の一番最後に操作 A で S[t] を入力したことにする
- dp[s−1][t] にコスト A で遷移 ：S[s:t] に遷移した過程の一番最初に操作 A で S[s−1] を入力したことにする
- a[s,t] に含まれる全ての i について ：
  a[s,t] のうち i+(t−s) 以上の最小のものを辿っていき、更新

小課題４

クリップボード ：bab

abababaababbab

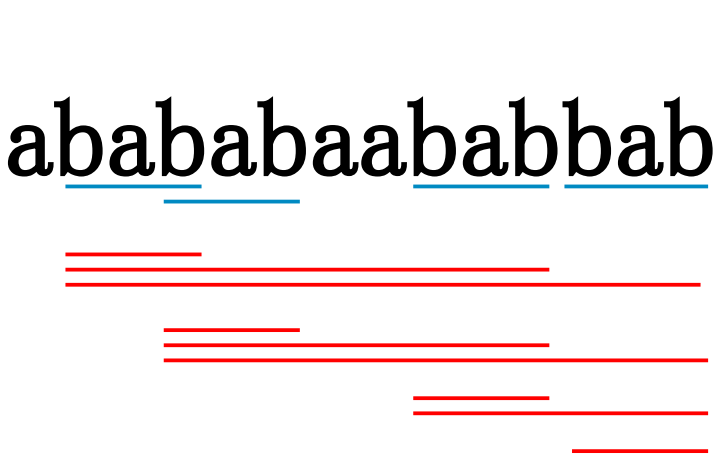


dp[1][4], dp[3][6],dp[8][11],dp[11][14]をまとめて遷移
→１遷移あたりO(N)

全体でO(N^3)

小課題 5

N ≦ 1000



よく見ると 1 遷移あたり
O(N / (長さ)) になっているので
合計 O(N^2 log(N))

小課題６（満点）

N ≦ 2500
O(N^2 log(N)) で通る場合と通らない場合がある

小課題6（満点）

$N \leqq 2500$

最初に全ての s,t について S[s:t]=S[i,i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の各要素 i について、 a[s,t] に含まれ $j \geqq i+t-s$ を満たす最小の要素 j を前計算する

全部分文字列の Rolling Hash の結果を map に入れれば O(N^2 log(N))

小課題6（満点）

$N \leqq 2500$

最初に全ての s,t について S[s:t]=S[i,i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の各要素 i について、 a[s,t] に含まれ $j \geqq i+t-s$ を満たす最小の要素 j を前計算する

全部分文字列の Rolling Hash の結果を map に入れれば O(N^2 log(N))

小課題6（満点）

$N \leqq 2500$

最初に全ての s,t について S[s:t]=S[i,i+(t−s)] となる i のリスト a[s,t] を作る
a[s,t] の各要素 i について、 a[s,t] に含まれ j ≧ i+t−s を満たす最小の要素 j を前計算する

全部分文字列の Rolling Hash の結果を map に入れれば O(N^2 log(N))

全部分文字列のハッシュを求めたあと長さごとに分け map で処理すると TLE が消える
Z algorithm 等で上の処理を O(N^2) にすることもできる

おわり

ありがとうございました