AGC 039 解説

DEGwer

2019/10/05

For International Readers: English editorial starts on page 4.

A: Connection and Disconnection

同じ文字が k 個連続する場合、そのうちの $\lfloor k/2 \rfloor$ 文字を変更する必要があります。逆に、それだけ変更すれば十分なことも分かります。

S の全文字が同じ場合、答えは $\lfloor |S|K/2 \rfloor$ です。そうでない場合、同じ文字の連続は S の内部または 2 つの S の接続部に現れます。S の先頭と末尾の文字が異なる場合、答えは S に対する答えの K 倍です。同じ場合、S の先頭・末尾に続く同じ文字の個数を a, b として、S に対する答えの K 倍から $\lfloor a/2 \rfloor + \lfloor b/2 \rfloor - \lfloor (a+b)/2 \rfloor$ の $K-1$ 倍を引いたものになります。時間計算量は $O(|S|)$ です。

B: Graph Partition

条件を満たす分割が可能だとすると、$V_1 \cup V_3 \cup \ldots$ と $V_2 \cup V_4 \cup \ldots$ は頂点集合の二分割であって同じ側の頂点どうしを結ぶ辺のないものになります。よって、与えられるグラフが二部グラフでないなら、分割は不可能です。

そうでない場合、与えられるグラフの直径を d とし、距離 d の 2 点を s, t とします。V_i を s からの距離が $i-1$ であるような頂点集合とすれば、条件を満たす $d+1$ 分割が得られます。逆に、$d+2$ 個以上の分割ができた場合、V_1 の頂点と V_{d+2} の頂点の間の距離は $d+1$ 以上必要であり、矛盾します。よって、分割の最大数は $d+1$ です。時間計算量は、Warshall-Floyd のアルゴリズムなどを用いれば $O(N^3)$ です。

C: Division by Two with Something

X の二進表記を考えれば、この操作は $2N$ 回行うと元に戻ることが分かります。各 $2N$ の約数 k に対し、k 回の操作で (初めてとは限らず) 元にもどるような整数の個数を求められれば、あとは包除原理を用いて答えを求められます。以下、X と言った時は X の N 桁での 二進表記を表すことにします。

X の後ろに X のビットをすべて反転したものを連結した長さ $2N$ の文字列を S とします。k 回の操作で元に戻ることは、S が周期 k を持つことと同値です。S の作り方より、$2N/k$ が奇数となる必要があることも分かります (偶数の場合、X と X のビット反転が一致することになるため)。よって、S が周期 k を持つのは、k が偶数かつ、ある $k/2$ 文字の列 T が存在して、S が T と T のビット反転を交互に並べてできる場合のみであることが分かります。

あとはこのようにして作った S の先頭 N 文字が辞書順で X 以下であるような T の個数を求めればよいです。X の先頭 $k/2$ 文字を T' として、T が辞書順で T' より小さい場合は条件を満たします。大きい場合は条件を満たしません。等しい場合は、$T = T'$ から実際に S を構成し、X と比較すればよいです。

時間計算量は、N の奇数の約数の個数を d として、$O(Nd)$ です。問題の制約下で $d \leq 72$ なので、実行時間制限に間に合います。

D: Incenters

単位円周上の 3 点 A, B, C に対し、点 A' を B, C を結ぶ弧であって A を含まないものの中点とします。点 B', C' も同様に定義します。このとき、三角形 ABC の内心 I は三角形 $A'B'C'$ の垂心に一致することが証明できます (円周角の定理より A, I, A' が同一直線上にあることに注意しながら、円周角の定理を繰り返し使って角度計算を行う)。一般に三角形の垂心 H 、重心 G 、外心 O はこの順に直線上にのっており、$|HG| : |GO| = 2 : 1$ であることが知られています (オイラー線)。三角形 $A'B'C'$ の外心の座標は $(0, 0)$ であり、重心の座標は A', B', C' の座標の和の $1/3$ 倍であるので、I の座標は A', B', C' の座標の和となります。

A' の座標は B, C の選び方と A, B, C の位置関係だけに依存し、(A が 直線 BC に対して A' と異なる側にある限り) A の具体的な位置には依存しません。よって、A' のそれぞれの選び方 $O(N^2)$ 通りに対し、A' の座標の最終的な答えへの寄与分を求め、すべて足し合わせることで答えを求めることができます。時間計算量は $O(N^2)$ です。

E: Pairing Points

頂点 $2N, 1$ の間で円環を切り開き、直線として考えます。頂点 1 のマッチング相手を頂点 i とします。

もし $j < k < i < j' < k'$ であって (j, j') と (k, k') がともにマッチされているものが存在すれば、$(1, i)$ をあわせてサイクルができ、矛盾します。よって、($N \geq 2$ なら) $(1, i)$ と交わるマッチングの辺 (a, b) $(a < b)$ であって、i から見て一番外側にあるものを取ることができます。$(1, i)$ を取り除いたときの頂点 a, b を含む連結成分に属する頂点であって添字が i 未満のものの集合を A とし、添字が i より大きいものの集合を B とし、$1, A, B$ 以外の頂点集合を C とします。このとき、A, C, B はこの順に $[2, \ldots, 2N]$ の 3 つの区間への分割になっていることが証明できます (そうでない場合、サイクルができます)。A, C, B それぞれに対し、マッチングの辺 $(1, i)$ に対応するものをそれぞれ $(b, a), (1, i), (a, b)$ とすることで、再帰的に同様の議論を行うことができます。

$\mathrm{DP}[s][t][u]$ を、区間 $[s, u]$ の頂点のうち t 以外からなる完全マッチングであって、マッチングの辺からなるサイクルが発生しないものの個数とします。上の考察より、$\mathrm{DP}[s][t][u]$ は、$(r_1, r_2) \in S$ となるような $s \leq r_1 \leq p_1 < t < p_2 \leq r_2 \leq u$ すべてに対して、$\mathrm{DP}[s][r_1][p_1] \times \mathrm{DP}[p_1 + 1][t][p_2 - 1] \times \mathrm{DP}[p_2][r_2][u]$ を足し合わせたものになります。よって動的計画法が動作し、時間計算量 $O(N^7)$ でこの問題を解くことができます。定数倍を評価すれば、適当な実装でも十分高速に動作することが分かります。

F: Min Product Sum

この問題は以下の問題と等価です。

問題 1　$N \times M$ の 2 つのグリッド A, B に 1 以上 K 以下の整数を書き込む方法であって、以下を満たすも

のの個数を　mod 998244353 で求めよ。

- 任意の $1 \leq i, k \leq N, 1 \leq j \leq M$ に対し、$A_{i,j} \leq B_{k,j}$
- 任意の $1 \leq i \leq N, 1 \leq j, k \leq M$ に対し、$A_{i,j} \leq B_{i,k}$

これは、もとの問題のグリッドを B としたときの A としてありうるものの個数が求めて足し合わせるべき値に一致することから分かります。

X_i を A の i 行目の整数の最大値、Y_j を B の j 列目の整数の最小値とします。$X_i = 1, Y_j = 1, X_i = 2, Y_j = 2, \ldots, X_i = K, Y_j = K$ の順に X_i, Y_j の値を決めていく DP を考えます。具体的には、DP[p][q][k][0] は以下のように定まります。

- $X_i \leq k$ なる i に対応する行集合を X とする
- $Y_j \leq k$ なる j に対応する列集合を Y とする
- X は p 行からなり、Y は q 行からなる
- X, Y の和集合に属する A のマスと X, Y の共通部分に属する B のマスに、条件を満たすように値を書き込む方法の個数が DP[p][q][k][0] である

同様に、DP[p][q][k][1] は以下のように定まります。

- $X_i \leq k+1$ なる i に対応する行集合を X とする
- $Y_j \leq k$ なる j に対応する列集合を Y とする
- X は p 行からなり、Y は q 行からなる
- X, Y の和集合に属する A のマスと X, Y の共通部分に属する B のマスに、条件を満たすように値を書き込む方法の個数が DP[p][q][k][1] である

このように定めれば、DP[p][q][k][0] たちの値から DP[p'][q'][k][1] たちの値を、DP[p][q][k][1] たちの値から DP[p'][q'][$k+1$][0] たちの値を、それぞれ $O(NM(N+M))$ 時間で求めることができます。直感的には、前者の遷移は $X_i = k+1$ なる行を、後者の遷移は $Y_j = k+1$ なる列を、それぞれ決めていることになります。合計の時間計算量は $O(NM(N+M)K)$ となり、制限実行時間に収まります。

AGC 039 Editorial

DEGwer

2019/10/05

A: Connection and Disconnection

When the same character occurs k times in a row, we have to change $\lfloor k/2 \rfloor$ of them. We can also see that no more change is necessary.

When all the characters in S are the same, the answer is $\lfloor |S|K/2 \rfloor$. Otherwise, contiguous segments of the same character appear within a single copy of S or straddle between two copies of S. If the first and last characters of S are different, the answer is the answer for a single copy of S multiplied by K. If these characters are the same, let a and b be the number of the same characters at the beginning and the end of S, respectively. The answer for this case is the answer for a single copy of S multiplied by K, minus $\lfloor a/2 \rfloor + \lfloor b/2 \rfloor - \lfloor (a+b)/2 \rfloor$ multiplied by $K-1$. Thus, the problem can be solved in $O(|S|)$ time.

B: Graph Partition

Suppose that the division is possible. Then, $V_1 \cup V_3 \cup \ldots$ and $V_2 \cup V_4 \cup \ldots$ is a bisection of the vertex set such that no edge connects two vertices on the same side. Thus, if the given graph is not bipartite, the division is impossible.

Otherwise, let d be the diameter of the given graph, and let s and t be two vertices with the distance of d. By letting V_i be the set of vertices whose distances from s are $i-1$, we can divide the vertices into $d+1$ sets and satisfy the condition. On the other hand, if we could divide the vertices into $d+2$ or more sets, the distance between a vertex in V_1 and a vertex in V_{d+2} has to be at least $d+1$, and we have a contradiction. Thus, the maximum possible number of sets is $d+1$. Using Floyd-Warshall algorithm, for example, the problem can be solved in $O(N^3)$ time.

C: Division by Two with Something

Considering the binary representation of X, we can see that after $2N$ operations the integer returns to its original value. Thus, if we find the number of integers that return to the original value (not necessarily for the first time) after k operations for each divisor k of $2N$, we can find the answer using inclusion-exclusion principle. Below, we denote by X the binary representation of X with N digits.

Let S be the string of length $2N$ obtained by concatenating X and a copy of X where each bit is

inverted in this order. X returns to the original value after k operations if and only if S has a period of k. From the construction of S, we can also see that $2N/k$ needs to be odd (otherwise X and its inversion would be equal). Thus, S has a period of k only if k is even and there exists a sequence T of length $k/2$ such that S can be made by alternating T and its inversion.

Now, we only have to find the number of T such that the first N characters of S, which is made as above, is lexicographically less than or equal to X. Let T' be the first $k/2$ characters of X. If T is lexicographically smaller than T', this condition is satisfied. If T is lexicographically greater than T', the condition is not satisfied. If T is equal to T', we can actually construct S from $T = T'$, then compare it with X.

Thus, the problem can be solved in $O(Nd)$ time where d is the number of odd divisors of N. Under the constraints of the problem, $d \leq 72$, so the solution will run in time.

D: Incenters

For points A, B and C on the circumference, let A' be the midpoint of the arc connecting B and C not passing through A. B' and C' are also defined similarly. It can be proved here that the incenter I of triangle ABC coincides with the orthocenter of triangle $A'B'C'$ (by noticing that, from the theorem of the angle of circumference, A, I, A' lie on the same line, and repeatedly use this theorem to calculate angles). Also, it is known that, for a general triangle, the orthocenter H, the centroid G, and the circumcenter O, in this order, lie on the same line (Euler line) and $|HG| : |GO| = 2 : 1$. The circumcenter of triangle $A'B'C'$ has the coordinates $(0, 0)$, and the centroid has the coordinates equal to the average of the coordinates of A', B', and C', so the coordinates of I is the sum of those of A', B', and C'.

The coordinates of A' only depend on the choices of B and C and positional relation of A, B, and C, and do not depend on the specific position of A (as long as A is on the different side from A' across line BC). Thus, we can find the answer by, for each of the $O(N^2)$ choices of A', finding the contribution of the coordinates of A' to the answer and summing them up, in $O(N^2)$ time.

E: Pairing Points

Let point a be the opponent of point 1. Let m be the number of segments that intersect with the segment between 1 and a (call it the main segment). Then, there exist $1 < b_1 < ... < b_m < a < c_m < ... < c_1$ such that for each i, the segment between b_i and c_i intersect with the main segment.

Let's remove the main segment. Now the remaining segments are divided into m components. For some (p, q), the points $[2, p] \cup [q, 2N]$ are in the first component (i.e., the component with a_1 and b_1), and the points $[p+1, a-1] \cup [a+1, q-1]$ are in the other components. We can repeat the same observation recursively and get a DP solution.

- Let $DP[a][b][c][d]$ be the number of ways to connect points $[a, b] \cup [c, d]$ such that they form a tree and there is exactly one segment between the left part ($[a, b]$) and the right part ($[c, d]$).
- Let $DP2[a][b][c][d]$ be the number of ways to connect points $[a, b] \cup [c, d]$ such that they form a forest (i.e., possibly multiple trees).

The recurrence is easy:

- $DP[a][b][c][d] = \sum_{a \le p \le b, c \le q \le d} DP2[a][p-1][p+1][b] \dot{D}P2[c][q-1][q+1][d] \dot{A}[p][q]$
- $DP2[a][b][c][d] = \sum_{a \le p \le b, c \le q \le d} DP[a][p][q][d] \dot{D}P2[p+1][b][c][q-1]$

This leads to an $O(N^6)$ solution, but if you count the number of iterations more carefully it's about $(2N)^6/6!$ and it works in time.

F: Min Product Sum

The problem is equivalent to the following:

Problem 1 Let A, B be two grids of dimensions $N \times M$. Count the number of ways to fill the grids with integers between 1 and K, such that for each cell of A, the number written on the cell is less than or equal to the minimum of the $N + M - 1$ integers written on the corresponding row or column of grid B.

We can see B as the original grid in the statement; then it's clear that the number of ways to fill A is equal to the product in the statement.

This is also equivalent to:

Problem 2 Let A, B be two grids of dimensions $N \times M$. Count the number of ways to fill the grids with integers between 1 and K, such that

- For each i, (the maximum of row i in grid A) $\leq$ (the minimum of row i in grid B).
- For each i, (the maximum of column j in grid A) $\leq$ (the minimum of column j in grid B).

Here's the tricky part: let's fix (the **maximum** of **row** i in grid A) for each i and (the **minimum** of **column** j in grid B) for each j. Once we fix these values, for example, we want to count the number of ways to fill the grid in the following way:

- See picture 1 (in the next page). We want to count the number of ways to fill these two grids.
- See picture 2. We assigned candidate of integers that can be written on each cell.
- See picture 3. There are actually more constraints: each row of the red subrectangle must contain at least one 1 (though this says nothing), each row of the orange subrectangle must contain at least one 2, and each row of the yellow subrectangle must contain at least one 3. Similar conditions for columns in grid B exist.
- See picture 4. Now the number of ways to fill the grids when rowmax/columnmin are fixed can be rephrased as the number of paths along the bold line. For example, when we move from P to Q, we multiply the coefficient (the number of ways to fill the orange rectangle) times (the number of ways to fill the purple rectangle).

By counting the total number of paths from the top-left corner to the bottom-right corner (possible along non-bold ways), we can count the number of ways to fill the grid when rowmax/columnmin are not fixed. This leads to an $O(NM(N + M)K)$ DP.

Picture 1

A	Max<=1	Max<=1	Max<=2	Max<=2	Max<=3	Max<=3
Max=1						
Max=1						
Max=2						
Max=2						
Max=3						
Max=3						

B	Min=1	Min=1	Min=2	Min=2	Min=3	Min=3
Min>=1						
Min>=1						
Min>=2						
Min>=2						
Min>=3						
Min>=3						

Picture 2

A	Max<=1	Max<=1	Max<=2	Max<=2	Max<=3	Max<=3
Max=1	1	1	1	1	1	1
Max=1	1	1	1	1	1	1
Max=2	1	1	12	12	12	12
Max=2	1	1	12	12	12	12
Max=3	1	1	12	12	123	123
Max=3	1	1	12	12	123	123

B	Min=1	Min=1	Min=2	Min=2	Min=3	Min=3
Min>=1	123	123	23	23	3	3
Min>=1	123	123	23	23	3	3
Min>=2	23	23	23	23	3	3
Min>=2	23	23	23	23	3	3
Min>=3	3	3	3	3	3	3
Min>=3	3	3	3	3	3	3

Picture 3

A	Max<=1	Max<=1	Max<=2	Max<=2	Max<=3	Max<=3
Max=1	1	1	1	1	1	1
Max=1	1	1	1	1	1	1
Max=2	1	1	12	12	12	12
Max=2	1	1	12	12	12	12
Max=3	1	1	12	12	123	123
Max=3	1	1	12	12	123	123

B	Min=1	Min=1	Min=2	Min=2	Min=3	Min=3
Min>=1	123	123	23	23	3	3
Min>=1	123	123	23	23	3	3
Min>=2	23	23	23	23	3	3
Min>=2	23	23	23	23	3	3
Min>=3	3	3	3	3	3	3
Min>=3	3	3	3	3	3	3

Picture 4

A	Max<=1	Max<=1	Max<=2	Max<=2	Max<=3	Max<=3
Max=1						
Max=1			P			
Max=2						
Max=2						
Max=3		Q				
Max=3						

図 1　ガザニア