其他技巧

Speaker 李昕威
2022/07/29

大綱

- 改進暴力
    - 折半枚舉
    - 啟發式合併
    - 均攤分析
- 壓常優化
- 根號算法

硬幣加總問題

- **題目**

給定 n 種硬幣，第 i 個硬幣的面額是 a_i。
每種硬幣至多取一個。
請問是否存在一種硬幣的取法，使得面額加總為 k？

數字範圍

- $n \leq 40$
- $1 \leq a_i \leq 10^9$
- $1 \leq k \leq 10^9$

想法一

- 對每個硬幣枚舉取或不取，共有 2^n 種選擇。
  對於每種選擇，需要 $O(n)$ 的時間去加總。
  時間複雜度 $O(n2^n) \Rightarrow$ TLE

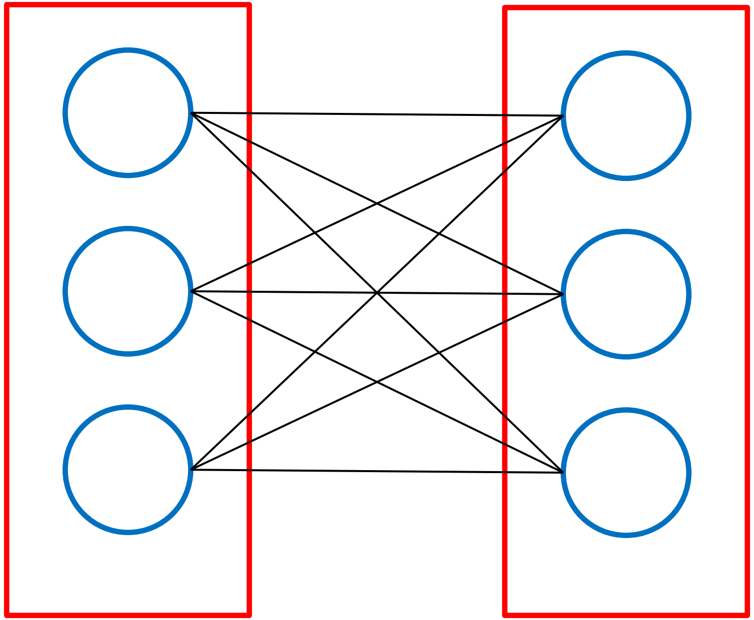
想法二

- 先枚舉 $a_1 \sim a_{\frac{n}{2}}$ 的取法。

  再枚舉 $a_{\frac{n}{2}+1} \sim a_n$ 的取法。

  最後加起來檢查。

  複雜度 $O\left(2^{\frac{n}{2}} \times 2^{\frac{n}{2}} \times n\right) = O(n2^n) \Rightarrow$ TLE



折半枚舉

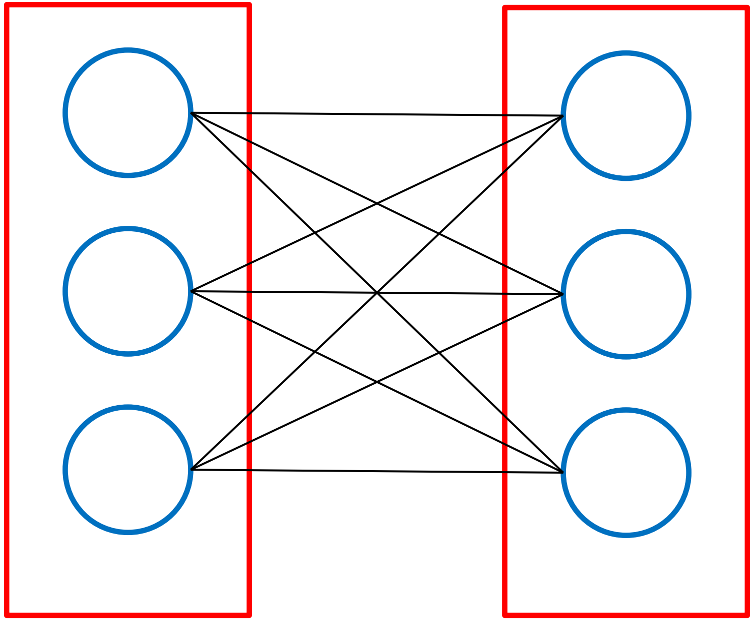
- 先枚舉完 $a_1 \sim a_{\frac{n}{2}}$ 的取法。

  再枚舉完 $a_{\frac{n}{2}+1} \sim a_n$ 的取法。

  檢查看看兩邊是否有個配對和為 k。

  可以將一邊 sort 後二分搜。

  複雜度 $O\left(2^{\frac{n}{2}} + 2^{\frac{n}{2}} + 2^{\frac{n}{2}} \log 2^{\frac{n}{2}}\right) = O\left(n 2^{\frac{n}{2}}\right)$



折半枚舉

- 一般的枚舉稱為 1-way search。

  折半枚舉則稱為 2-way search 或 meet in the middle。

  若是要枚舉的對象可以分成 A、B 兩部分的組合，

  就可以先枚舉完 A、再枚舉完 B，最終再嘗試合在一起。

XOR-Path

- **題目**

給定一個 $n \times n$ 的棋盤，每個格子上有個數值 $v_{i,j}$。

一開始有個棋子在左上角。它只能向右或向下移動。

是否存在移動到右下角的方式，使經過格子的數值 XOR 起來等於 k？

數字範圍

- $n \leq 20$
- $1 \leq v_{i,j} \leq 10^9$
- $1 \leq k \leq 10^9$

XOR-Path

- **Example**
- $K = 001$
- $(1,1) \to (2,1) \to (2,2) \to (2,3) \to (3,3)$
- $001 \oplus 000 \oplus 010 \oplus 111 \oplus 101 = 001$

001	011	111
000	010	111
100	111	101

想法一

- 暴力枚舉所有 $(1,1)$ 到 (n,n) 的路徑。

複雜度 $O\left(\binom{2n}{n} \times n\right) \Rightarrow$ TLE

折半枚舉

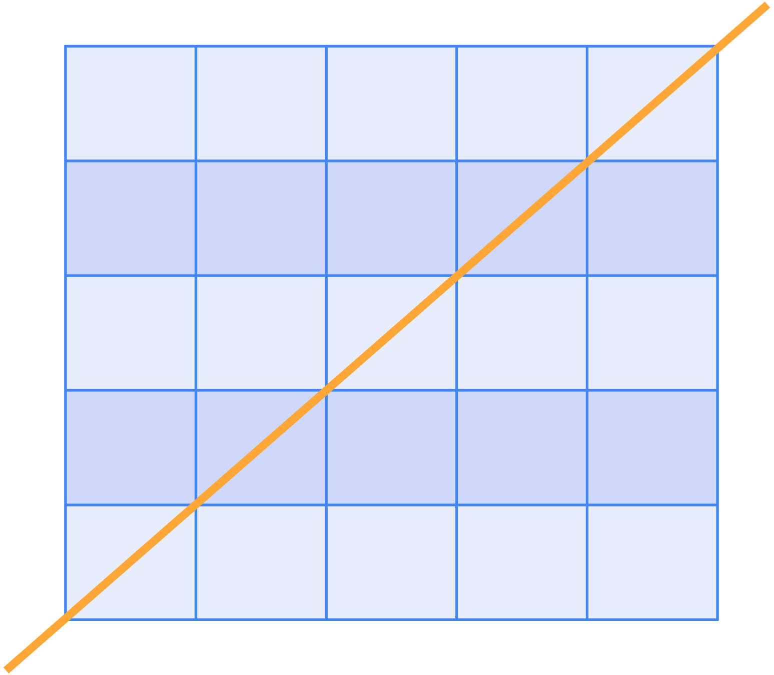
- 枚舉 $(1,1)$ 走到對角線的走法。

  枚舉 (n,n) 走到對角線的走法。

  最後在對角線上的每個格子

  檢查是否存在能匹配的走法

  複雜度 $O(n \times 2^n)$



大綱

- 改進暴力
  - 折半枚舉
  - 啟發式合併
  - 均攤分析
- 壓常優化
- 根號算法

Disjoint Set

- **題目**

給定 n 個集合 $S_1, S_2, \dots, S_n$。一開始 $S_i = \{i\}$。

有 q 筆操作，分別為以下兩種之一：

1. $Union(i, j)$ 合併 i 和 j 所在的集合
2. $Same(i, j)$ 回答 i 和 j 是否在同個集合內

數字範圍

- $n, q \le 10^5$

想法一

- $S(i) =$ 第 i 個集合包含的元素。
  $belong(i) = i$ 所在的集合編號。
  也就是 $i \in S\big(beglong(i)\big)$。

想法一

- 對於操作一，遍歷 $t \in S\left(belong(j)\right)$，將 $belong(t)$ 改成 $belong(j)$。
  對於操作二，檢查 $belong(i) == beglong(j)$。
  複雜度 $O(qn) \Rightarrow$ TLE。

啟發式合併

- 第一筆操作太慢了，需要嘗試優化它。
  如果 $belong(i) = belong(j)$ 則不用做。
  反之則比較 $S\big(belong(i)\big)$ 和 $S\big(belong(j)\big)$ 的大小，將小的合併進大的。

複雜度分析

- 令 $T(n)$ = 將 n 個 $S_i = \{i\}$ 透過操作一，合併成一個集合的時間複雜度。

  Claim：$T(n) \leq n \log_2 n = O(n \log n)$。

複雜度分析 – 數學歸納法

Base case

$n = 1$ 時，$T(n) = 0 = 1\log_2 1$。

複雜度分析 – 數學歸納法

Recursive Case

假設對於 $k < n$，都有 $T(k) \le k \log_2 k$。

考慮最後一次合併，是將一個大小為 a 的集合與大小為 b 的集合合併。

注意到 $a + b = n$。

不失一般性，假設 $a \le b \Rightarrow a \le \frac{n}{2}$。

則 $T(n) = T(a) + T(b) + a$。

複雜度分析 – 數學歸納法

- $T(n) = T(a) + T(b) + a$

$= a\log_2 a + b\log_2 b + a$

$\leq a\log_2 \frac{n}{2} + b\log_2 b + a$

$= a\left(\log_2 \frac{n}{2} + 1\right) + b\log_2 b$

$= a\log_2 n + b\log_2 b$

$\leq a\log_2 n + b\log_2 n$

$= (a+b)\log_2 n = n\log_2 n$

複雜度分析 – 另解

- 觀察操作一，當一個元素需要移動時，則其所在的集合大小至少變為兩倍。

  因此一個元素至多被移動 $\log_2 n$ 次。

  所以操作一的複雜度為 $O(n \log_2 n) = O(n \log n)$。

啟發式合併

當兩個集合要做合併時，固定大的，將小的合併進去。

樹上支配者

- **題目**

給定一顆大小為 n 的有根二元樹 T，每個點 i 有權重 v_i。

對於每個子樹 S，若 x 是其下的最小眾數，則稱 x 支配了 S。

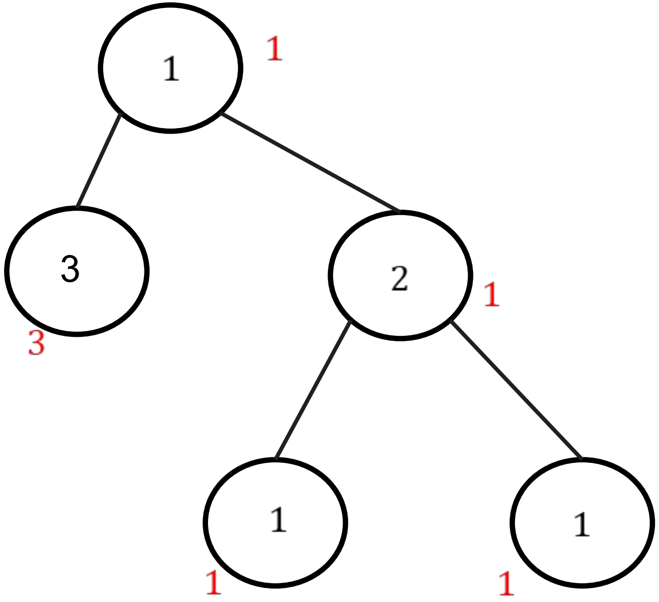
請對每個子樹求出是多少支配了它。

數字範圍

- $n \leq 10^5$
- $1 \leq v_i \leq n$

樹上支配者

Example



想法一

- 暴力走訪每顆子樹去求答案。
  複雜度 $O(n^2) \Rightarrow$ TLE。

想法二

- 令 $f(i)$ 為一個 std::map，維護了 i 為根的子樹下，每個數字出現的次數。
  注意到 $f(i)$ 可以由 $j \in children(i)$ 的 $f(j)$ 來湊出來。
  換句話說，將 $j \in children(i)$ 的 $f(j)$ 合併後能夠得到 $f(i)$。

啟發式合併

- 以 DFS 後序的順序去計算每個子樹的答案。

  對於節點 i，當左右子樹的答案都算好後，將小的 f 合併進大的 f。

  同時在合併的過程中維護好最小眾數。

  複雜度 $O(n\log^2 n)$。

樹上支配者

題目

給定一顆大小為 n 的有根樹 T，每個點 i 有權重 v_i。

對於每個子樹 S，若 x 是其下的最小眾數，則稱 x 支配了 S。

請對每個子樹求出是多少支配了它。

數字範圍

- $n \leq 10^5$
- $1 \leq v_i \leq n$

K-Path

- **題目**

給定一顆大小為 n 的樹 T，請問 T 有多少條長度為 k 的相異路徑？

數字範圍

- $n \le 10^5$
- $1 \le k \le n$

大綱

- 改進暴力
  - 折半枚舉
  - 啟發式合併
  - 均攤分析
- 壓常優化
- 根號算法

Stack 練習題

- **題目**

有個 stack，一開始是空的。有 q 筆操作，分別為

1. Push x 進入 stack。
2. 回答當前 stack 的大小，並清空 stack。

數字範圍

- $q \leq 10^6$
- $1 \leq x \leq 10^9$

複雜度分析

- 操作一 $O(1)$

  操作二 $O(q)$

  總時間複雜度 $O(q^2) \Rightarrow ？？？$

複雜度分析

- 注意到，一個數字只有被 Push 進去，才能被 Pop 出來。

每個數字至多被 Push、Pop 各一次。

總時間複雜度 $O(2q)$。

均攤分析

只看單個操作二的最差複雜度很糟糕。

但如果將所有的操作二合在一起考慮，

會發現它們的總和被操作一的數量 bound 住。

均攤分析

以位能來解釋。

每個操作一會使位能加一。

每個操作二會使位能歸零。

只有位能增加時，才能夠減少。

有效交換

- **題目**

給定一個 $1 \sim n$ ($n \leq 10^5$) 的 permutation $p_1, p_2, \ldots, p_n$。一開始 $p_i = i$。

有 q 筆操作，分別為：

1. $swap(p_x, p_y)$
2. 對於 $i = l \sim r$，令 $p_k = i$，$swap(p_k, p_i)$。

有效交換的定義是，$swap(p_i, p_j)$ 時 $i \neq j$。

請問一共有多少個有效交換呢？

觀察

- 操作一中若 $x \neq y$，則是有效交換。

操作二中，只有 $p_i \neq i$ 的那些才是有效交換。

將位能定義為 $p_i \neq i$ 的數量。

操作一可能將位能減二、減一、不變、加一、加二。

操作二中只會消耗位能。

解法

- 令 $S = \{i \mid p_i \neq i\}$。並以 std::set 維護 S。
  對於操作一，根據結果來更新 S。
  對於操作二，則二分搜並遍歷 S 內介於 $l \sim r$ 的元素，並對它們操作即可。
  時間複雜度 $O(q \log n)$。

有效交換

- **題目**

給定一個 $1 \sim n$ $(n \le 10^5)$ 的 permutation $p_1, p_2, \ldots, p_n$。

有 q 筆操作，分別為：

1. $swap(p_x, p_y)$
2. 對於 $i = l \sim r$，令 $p_k = i$，$swap(p_k, p_i)$。

有效交換的定義是，$swap(p_i, p_j)$ 時 $i \neq j$。

請問一共有多少個有效交換呢？

大綱

- 改進暴力
- 壓常優化
  - Bit Vector 優化
- 根號算法

Bit Vector

- std::bitset 可以想像是 Bool 型別的陣列。

  可以對其做 AND、OR、XOR、左右移、count……等等。

  std::bitset 的單次操作，約為一般 int 陣列上單次操作時間的 $\frac{1}{32}$ 倍

湊數字問題

題目

給定 n 個整數 $a_1, a_2, \ldots, a_n$。

每個數字至多被取一次。

請問是否存在一種取法使得總和為 k？

數字範圍

- $n \leq 10^5$
- $k \leq 5 \times 10^4$
- $1 \leq a_i \leq 5 \times 10^4$

想法

- $f(i,j) =$ 是否能用 $a_1, a_2, \ldots, a_n$ 湊出 j。
  則 $f(i,j) = f(i-1,j)\ OR\ f(i-1,j-a_i)$。
  時間複雜度 $O(nk) \Rightarrow$ TLE。

Bit Vector 優化

- 注意到 $f(i,)$ 是 bool 陣列，可以使用 std::bitset 維護。

  並且 $f(i-1,j)$ 和 $f(i-1,j-a_i)$ 間相當於右移 a_i 個 bits。

  因此 $f(i,) = f(i-1,)\ OR\ (f(i-1,) \gg a_i)$。

  時間複雜度 $O\left(\frac{nk}{32}\right)$。

Bit Vector 優化

當題目中牽扯到 bool 陣列，並且和 AND、OR 等運算相關時。

或許就能夠試著使用 bit vector 去加速。

長方形數量

- **題目**

給定 $n \times m$ 的棋盤。格子是黑色或者白色。

請問棋盤上有幾個長方形的四個角落都是黑色的？

數字範圍

- $1 \leq n \leq 10^3$
- $1 \leq m \leq 10^3$

想法一

- 先枚舉任兩個橫排 r_1, r_2。
  再枚舉任兩個直排 c_1, c_2。
  再檢查四個交點是否都是黑色。
  時間複雜度 $O(n^2m^2) \Rightarrow$ TLE。

想法二

- 先枚舉任兩個橫排 r_1, r_2。

  計算這兩個橫排有幾個共同 column 都是黑色。

  若有 x 個，代表這兩個橫排間有 $\binom{x}{2}$ 個黑色長方形。

  時間複雜度 $O(n^2 m) \Rightarrow$ TLE。

想法二

- 將每個橫排看成 bit vector，黑色是 1、白色是 0。

  計算兩個橫排共同黑色 column，相當於兩個 bit vector AND 後數 1 的數量。

  時間複雜度 $O\left(\frac{n^2 m}{32}\right)$。

大綱

- 改進暴力
- 壓常優化
- 根號算法
    - 分塊
    - 分段討論

區間最大值

- **題目**

給定一個長度為 n 的序列 $a_1, a_2, \ldots, a_n$。支援 q 次以下兩種操作：

1. 修改某個元素的值
2. 查詢 $a[l \sim r]$ 的最大值

數字範圍

- $1 \leq n, q \leq 10^5$
- $1 \leq a_i \leq 10^9$

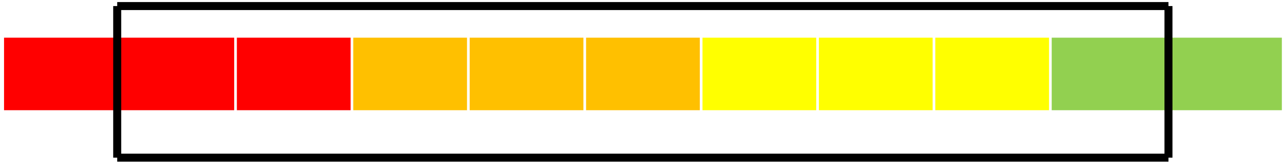
分塊

- 每 k 個數字分成一塊，並對每塊維護其最大值。

  區間查詢時，完整包含的塊就直接回傳其最大值，

  對於未完整包含的，則暴力掃過一遍。

  時間複雜度 $O\left(k+\frac{n}{k}\right)$。



傳送門

- **題目**

有 n 個傳送門，分別在座標 $1 \sim n$ 上。

若座標 i 有顆球，則其會被傳送到 $i + a_i$，並重複直到最標超過 n。請支援 q 筆操作：

1. 修改 a_i
2. 在 x 放下一顆球，輸出被傳送的次數與最終停留的位置。

數字範圍

- $1 \leq n, q \leq 10^5$
- $1 \leq a_i \leq n$

想法

- 將座標每 k 個切成一塊。

對於每塊內的位置 x，維護從其出發，直到離開這塊，所需的次數和最終位置。

修改 a_i 的話就重建其所在的那塊。$O(k)$

對於詢問，則不斷沿著每塊往下傳送。$O\left(\frac{n}{k}\right)$。

總時間複雜度 $O\left(k+\frac{n}{k}\right)$。

大綱

- 改進暴力
- 壓常優化
- 根號算法
  - 分塊
  - 分段討論

跳著加值

題目

給定 n 個數字 $a_1, a_2, \ldots, a_n$，支援 q 筆以下兩種操作：

1. 給定 x, y, z，對所有滿足 $i \equiv x \pmod y$，將 a_i 加上 z。
2. 給定 j，詢問 a_j 的值。

數字範圍

- $\le n, q \le 10^5$
- $1 \le a_i \le 10^9$
- $1 \le x, y, z \le n$

想法

- 對於 $y \le \sqrt{n}$，y 只有 $\sqrt{n}$ 種，而對應的 x 也只有 $\sqrt{n}$ 種。
  開個 $n \times \sqrt{n}$ 的二維陣列 add，遇到一筆 x、y、z 的修改操作，
  就將 $add[x][y]$ 加上 z，代表模 z 為 x 那些位置要加上 z。

  最後在詢問 a_j 的值時，枚舉 $k = 1 \sim \sqrt{n}$，將 $add[j \bmod k][k]$ 的值加總即可。

想法

- $y > \sqrt{n}$，要修改的元素至多只有 $\frac{n}{y}$ 個，也就是最多只有 $\sqrt{n}$ 個。

  因此可以直接暴力加值。

  總時間複雜度 $O(n\sqrt{n})$。