Solutions For Claris' Contest # 2 Day 1

Claris

Hangzhou Dianzi University

2016 年 9 月 4 日

String Master(master.c/cpp/pas)

给定两个字符串，求它们的最长公共子串，允许不超过 k 次失配。

String Master(master.c/cpp/pas)

给定两个字符串，求它们的最长公共子串，允许不超过 k 次失配。

- 对于 30% 的数据，$n \leq 10$，$k = 0$。

String Master(master.c/cpp/pas)

给定两个字符串，求它们的最长公共子串，允许不超过 k 次失配。

- 对于 30% 的数据，$n \leq 10$，$k = 0$。
- 对于 50% 的数据，$n \leq 10$，$k \leq 1$。

String Master(master.c/cpp/pas)

给定两个字符串，求它们的最长公共子串，允许不超过 k 次失配。

- 对于 30% 的数据，$n \leq 10$，$k = 0$。
- 对于 50% 的数据，$n \leq 10$，$k \leq 1$。
- 对于 100% 的数据，$n \leq 300$，$0 \leq k \leq n$。

30 分做法

- 对于 30% 的数据，$n \le 10$，$k = 0$。

30 分做法

- 对于 30% 的数据，$n \leq 10$，$k = 0$。
- 暴力枚举两个串所在位置，暴力检验即可。

30 分做法

- 对于 30% 的数据，$n \leq 10$，$k = 0$。
- 暴力枚举两个串所在位置，暴力检验即可。
- 时间复杂度 $O(n^4)$。

50 分做法

- 对于 50% 的数据，$n \leq 10$，$k \leq 1$。

50 分做法

- 对于 50% 的数据，$n \leq 10$，$k \leq 1$。
- 当 $k = 1$ 时，暴力枚举修改哪个位置，再枚举修改成什么字符，然后用 $k = 0$ 的算法计算即可。

50 分做法

- 对于 50% 的数据，$n \leq 10$，$k \leq 1$。
- 当 $k = 1$ 时，暴力枚举修改哪个位置，再枚举修改成什么字符，然后用 $k = 0$ 的算法计算即可。
- 时间复杂度 $O(26n^5)$。

100 分做法

- 暴力枚举两个后缀，计算最长能匹配多少前缀。

100 分做法

- 暴力枚举两个后缀，计算最长能匹配多少前缀。
- 最优策略一定是贪心改掉前 k 个失配的字符。

100 分做法

- 暴力枚举两个后缀，计算最长能匹配多少前缀。
- 最优策略一定是贪心改掉前 k 个失配的字符。
- 时间复杂度 $O(n^3)$。

Tourist Attractions(tour.c/cpp/pas)

给定一张 n 个点的无向图，统计有多少条简单路径恰好经过了 4 个点。

Tourist Attractions(tour.c/cpp/pas)

给定一张 n 个点的无向图，统计有多少条简单路径恰好经过了 4 个点。

- 对于 40% 的数据，$n \leq 50$。

Tourist Attractions(tour.c/cpp/pas)

给定一张 n 个点的无向图，统计有多少条简单路径恰好经过了 4 个点。

- 对于 40% 的数据，$n \leq 50$。
- 对于 70% 的数据，$n \leq 300$。

Tourist Attractions(tour.c/cpp/pas)

给定一张 n 个点的无向图，统计有多少条简单路径恰好经过了 4 个点。

- 对于 40% 的数据，$n \leq 50$。
- 对于 70% 的数据，$n \leq 300$。
- 对于 100% 的数据，$n \leq 1500$。

40 分做法

- 对于 40% 的数据，$n \le 50$。

40 分做法

- 对于 40% 的数据，$n \leq 50$。
- 暴力枚举路径的起点，然后进行搜索即可。

40 分做法

- 对于 40% 的数据，$n \leq 50$。
- 暴力枚举路径的起点，然后进行搜索即可。
- 时间复杂度 $O(n^4)$。

70 分做法

- 对于 70% 的数据，$n \leq 300$。

70 分做法

- 对于 70% 的数据，$n \leq 300$。
- 假设路径是 $a-b-c-d$，考虑枚举中间这条边 $b-c$，计算有多少可行的 a 和 d。

70 分做法

- 对于 70% 的数据，$n \le 300$。
- 假设路径是 $a-b-c-d$，考虑枚举中间这条边 $b-c$，计算有多少可行的 a 和 d。
- 设 deg_x 表示点 x 的度数，那么边 $b-c$ 对答案的贡献为 $(deg_b-1)(deg_c-1)-$ 经过 $b-c$ 这条边的三元环个数。

70 分做法

- 对于 70% 的数据，$n \leq 300$。
- 假设路径是 $a - b - c - d$，考虑枚举中间这条边 $b - c$，计算有多少可行的 a 和 d。
- 设 deg_x 表示点 x 的度数，那么边 $b - c$ 对答案的贡献为 $(deg_b - 1)(deg_c - 1) -$ 经过 $b - c$ 这条边的三元环个数。
- 计算三元环的个数只需要枚举除 b, c 之外的另一个点即可。

70 分做法

- 对于 70% 的数据，$n \le 300$。
- 假设路径是 $a-b-c-d$，考虑枚举中间这条边 $b-c$，计算有多少可行的 a 和 d。
- 设 deg_x 表示点 x 的度数，那么边 $b-c$ 对答案的贡献为 $(deg_b-1)(deg_c-1)-$ 经过 $b-c$ 这条边的三元环个数。
- 计算三元环的个数只需要枚举除 b,c 之外的另一个点即可。
- 时间复杂度 $O(n^3)$。

100 分做法

- 70 分算法的瓶颈在于三元环计数。

100 分做法

- 70 分算法的瓶颈在于三元环计数。
- 设 S_x 表示所有和 x 有边的点的集合，那么其实就是统计 $card(S_b \cap S_c)$。

100 分做法

- 70 分算法的瓶颈在于三元环计数。
- 设 S_x 表示所有和 x 有边的点的集合，那么其实就是统计 $card(S_b \cap S_c)$。
- 将 S 用二进制压位存储即可并行计算。

100 分做法

- 70 分算法的瓶颈在于三元环计数。
- 设 S_x 表示所有和 x 有边的点的集合，那么其实就是统计 $card(S_b \cap S_c)$。
- 将 S 用二进制压位存储即可并行计算。
- 时间复杂度 $O(\frac{n^3}{32})$。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

另外再给定 m 条有向边，假设边权都是 1，求 1 到每个点的最短路。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

另外再给定 m 条有向边，假设边权都是 1，求 1 到每个点的最短路。

- 对于 20% 的数据，$n \leq 5$，$m \leq 10$，$val_i < 2^4$。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

另外再给定 m 条有向边，假设边权都是 1，求 1 到每个点的最短路。

- 对于 20% 的数据，$n \le 5$，$m \le 10$，$val_i < 2^4$。
- 对于 40% 的数据，$n \le 2000$，$m \le 5000$，$val_i < 2^{10}$。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

另外再给定 m 条有向边，假设边权都是 1，求 1 到每个点的最短路。

- 对于 20% 的数据，$n \leq 5$，$m \leq 10$，$val_i < 2^4$。
- 对于 40% 的数据，$n \leq 2000$，$m \leq 5000$，$val_i < 2^{10}$。
- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。

Walk(walk.c/cpp/pas)

给定一张有 n 个点的有向图，点 i 的权值为 val_i，若 val_i and $val_j = val_j$，则 i 向 j 连边。

另外再给定 m 条有向边，假设边权都是 1，求 1 到每个点的最短路。

- 对于 20% 的数据，$n \leq 5$，$m \leq 10$，$val_i < 2^4$。
- 对于 40% 的数据，$n \leq 2000$，$m \leq 5000$，$val_i < 2^{10}$。
- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。
- 对于 100% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{20}$。

20 分做法

- 对于 20% 的数据，$n \le 5$，$m \le 10$，$val_i < 2^4$。

20 分做法

- 对于 20% 的数据，$n \leq 5$，$m \leq 10$，$val_i < 2^4$。
- 暴力搜索出最短路即可。

40 分做法

- 对于 40% 的数据，$n \leq 2000$，$m \leq 5000$，$val_i < 2^{10}$。

40 分做法

- 对于 40% 的数据，$n \le 2000$，$m \le 5000$，$val_i < 2^{10}$。
- 按照题目要求建好图，然后 BFS 求出 1 到每个点的最短路即可。

40 分做法

- 对于 40% 的数据，$n \le 2000$，$m \le 5000$，$val_i < 2^{10}$。
- 按照题目要求建好图，然后 BFS 求出 1 到每个点的最短路即可。
- 时间复杂度 $O(n^2 + m)$。

70 分做法

- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。

70 分做法

- 对于 70% 的数据，$n \le 200000$，$m \le 300000$，$val_i < 2^{15}$。
- 考虑新增 2^{15} 个点，这些点中 i 向它所有的子集连一条权值为 0 的有向边。

70 分做法

- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。
- 考虑新增 2^{15} 个点，这些点中 i 向它所有的子集连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。

70 分做法

- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。
- 考虑新增 2^{15} 个点，这些点中 i 向它所有的子集连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。
- BFS 的时候，每次要把用 0 权值的边连接的所有点都加入队尾，以保证距离不降。

70 分做法

- 对于 70% 的数据，$n \leq 200000$，$m \leq 300000$，$val_i < 2^{15}$。
- 考虑新增 2^{15} 个点，这些点中 i 向它所有的子集连一条权值为 0 的有向边。
- 对于原来的 n 个点，先把 m 条边连好，然后对于 i 号点，由它向新增的第 val_i 个点连一条权值为 1 的有向边，再由新增的第 val_i 个点向它连一条权值为 0 的有向边。
- BFS 的时候，每次要把用 0 权值的边连接的所有点都加入队尾，以保证距离不降。
- 时间复杂度 $O(3^{15} + n + m)$。

100 分做法

- 依旧考虑新增 2^{20} 个点。

100 分做法

- 依旧考虑新增 2^{20} 个点。
- i 只需要向 i 去掉某一位的 1 的点连边。

100 分做法

- 依旧考虑新增 2^{20} 个点。
- i 只需要向 i 去掉某一位的 1 的点连边。
- 这样一来图的边数就被压缩到了 $20 \cdot 2^{20} + 2n + m$，然后 BFS 求出 1 到每个点的最短路即可。

100 分做法

- 依旧考虑新增 2^{20} 个点。
- i 只需要向 i 去掉某一位的 1 的点连边。
- 这样一来图的边数就被压缩到了 $20 \cdot 2^{20} + 2n + m$，然后 BFS 求出 1 到每个点的最短路即可。
- 时间复杂度 $O(20 \cdot 2^{20} + n + m)$。

Thank you!