构造循环赛日程表

MyriadDreamin 2017211279 2017211301

这些天一直都比较忙, 但有空的时候总在琢磨循环赛日程表如何去安排. 思考许久, 未得要领. 先给出循环赛日程表的定义吧:

Definition 1 (循环赛日程表) 定义选手个数为 n, 比赛天数为 $n-1$ 天的日程表为循环赛日程表, 当且仅当对于每个第 $i(1 \leqslant i \leqslant n)$ 个选手, 在接下来的 $n-1$ 天中, 它与其他 $n-1$ 名选手各比赛一次.

显然如果要构造循环赛日程表. 每天必须选出一半的选手与剩下一半的选手比赛, 所以选手的个数必须为偶数. 如果选手为奇数, 则只能增加一个虚拟的选手, 如果有人抽到这名选手, 则轮空.

0.1 约定

1. 循环赛日程表的选手个数总是为 n=2m 个, 标号为 $1, 2, \ldots, n$.

0.2 循环赛日程表表示

循环赛总是由 n 阶方阵 $T = \begin{pmatrix} \text{racer} & \text{day}_1 & \ldots & \text{day}_{n-1} \end{pmatrix}^T$ 表示, 每一个元素都是一个行向量.

Definition 2 (标准形式的循环赛日程表) 第一行和第一列为自然序列的循环赛日程表, 称为标准形式的循环赛日程表. 如下:

$$T = \begin{pmatrix} racer \\ day_1 \\ \vdots \\ day_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 2 & \ldots & n \\ 2 & a_{2,2} & \ldots & a_{2,n} \\ \vdots & \vdots & \ddots & n \\ n & a_{n,2} & \ldots & a_{n,n} \end{pmatrix}$$

Definition 3 J_n 矩阵 J_n 矩阵 $=[1]_{n \times n}$

1 从对称性构造循环赛日程表

1.1 正确性证明

Theorem 4 *(1)* 设 $T_{1,1}$ 是标准循环赛日程表 T 的余子式, 那么,$T_{1,1}$ 中 *1* 恰出现了 $n-1$ 次, 其余标号恰出现了 $n-2$ 次.

(2) 若定义一次比赛中, 两人 (i,j) 之间的距离为 $i-j+kn, k\in Z$. 则在 K_{n-1} 中, 距离为 $x, x\in Z$ 的边定有 $n-1$ 条.

Theorem 5 (轮转法) 下面方法被称为轮转法:

将 $1,2,3,\ldots,n$ 逆时针排列, 对于 $d=1,2,...,n-1$ 的每一天:

(1) 对于 $d \pmod{n-1}+2, d+1 \pmod{n-1}+2, ..., d+n-1 \pmod{n-1}+2$ 的每一个选手, 选择距离为 $2,4,...,2n-2$ 顺时针大于他的选手.

(2) 对于 1 号选手, 第 i 天选择 $d+1$ 号选手.

证明: 设取得的边集为 S, 所有需要分配的边集为 E. 显然 $S\subset E$.

将对局按长度分类取模, 长度为 $1,2,\ldots n-2$ 各有 $(n-1)$ 条. 因为对于第 i 天, 长度为 $1,2,\ldots n-2$ 的边起点为 $d \pmod{n-1}+2, d+1 \pmod{n-1}+2, ..., d+n-1 \pmod{n-1}+2$, 除了在同一天的第 $(d+i) \pmod{n-1}+2$ 和 $(d+n-1-i) \pmod{n-1}+2$ 配对, 配对后的边两两不同. 而 *1* 号选手上被分配了 $(n-1)$ 条边, 这 $n-1$ 条边与前面的每一条边都不同.

所以总共分配了 $\dfrac{(n-1)^2}{2}+(n-1)=\dfrac{n(n-1)}{2}$ 条边. 而 $|E|=\dfrac{n(n-1)}{2}$, 所以 $S=E$.

注意到第一段中有的性质, 所以轮转法得到的方案是合法的.

1.2 复杂度分析

空间复杂度为 $\Theta(1)$, 时间复杂度为 $T(n)=\Theta(n^2)$.

2 搜索优化构造循环赛日程表

Definition 6 (scenarios) *有序整数对数组 S 为循环赛的一个 scenarios, 如果 S 的每对整数不重不漏地涵括了 n 个选手的对局.*

2.1 思路

2.1.1 分层搜索

下面是搜索 dfs(i) 的思路:

假设已经生成了所有可能的方案 S_0, 初始化:$i \leftarrow 0, S_1, S_2, \ldots, S_n \leftarrow \varnothing$, 解数组 $D[n] = []$.

dfs(i):

1 设已选择 i 个 scenarios, 检查是否 $i = n$, 如果是, 则结束算法.

2 对于每个可选的 scenarios $s \in S_i$, 将所有与 s 不矛盾的 scenarios 推入 S_{i+1}.

  1 调用一次搜索 dfs($i+1$).

  2 如果搜索状态标记为已成功, 则停止搜索, 标记 $D[i] = s$.

  3 如果搜索状态标记为不成功, 则令 $S_i \leftarrow S_i - \{s\}, S_{i+1} \leftarrow \varnothing$.

下面是生成 (generate()) 的思路:

初始化可供选择的集合为 $\Omega = \{1, 2, \ldots, n\}$,scenarios $s = \varnothing$, 可行方案 $S_0 = \varnothing$.

generate():

1 如果 $\Omega = \varnothing$, 则 $S_0 \leftarrow S_0 + \{s\}$.

2 选择 $\mathsf{fir} = \min \Omega$, 对于每个 $\mathsf{sec} \in \{\mathsf{sec} > \mathsf{fir} | \mathsf{sec}\Omega\}$.

  1 令 $\Omega \leftarrow \Omega - \{\mathsf{fir}, \mathsf{sec}\}, s \leftarrow s + \{(\mathsf{fir}, \mathsf{sec})\}$, 调用 generate().

  2 令 $\Omega \leftarrow \Omega + \{\mathsf{fir}, \mathsf{sec}\}$

2.1.2 2.1.1 的复杂度分析

设空间复杂度为 $S(n)$. 显然有 $S(n) = \Theta(n-1)S(n-2)$. 所以 $S(n) = \Theta\left((n-1)!!\right)$ 根据观察, 搜索算法能够一次遍历成功, 从而时间复杂度为 $T(n) = \mathrm{O}((n-1)!!)$. 很不巧, 因为生成序列的选择问题 $T(n)$ 也是 $\Theta((n-1)!!)$ 的. 关于单次递进遍历不会回溯的证明见 Theorem 7.

2.1.3 改进搜索

先证明 Theorem 7.

Theorem 7 *如果 S 包含所有的规模为 n 的 scenarios, 那么 2.1.1 的搜索不会回溯.*

证明: 首先循环赛日程表规划在 n 为偶数的情况下必然有解. 取一解数组 D,D 的顺序可以调换, 这不影响 D 仍为解, 所以存在一种顺序 $D[i_1], D[i_2], \ldots, D[i_{n-1}]$ 使得其在 S 中依次出现.

所以可以将 dfs 嵌入 generate 中. 算法 (dfs_gen()) 如下:

初始化可供选择的集合为 $\Omega \leftarrow \{1, 2, \ldots, n\}$,scenarios $s \leftarrow \varnothing, i \leftarrow 0$, 解数组 $D[n] \leftarrow []$.

generate():

1 如果 $\Omega = \varnothing$, 则判断 s 是否与 D 中已有方案矛盾.

2 如果不矛盾:

   1 令 $D[i] \leftarrow s, i \leftarrow i+1$.

   2 如果 $i = n$, 结束算法.

3 选择 $\mathsf{fir} = \min \Omega$, 对于每个 $\mathsf{sec} \in \{\mathsf{sec} > \mathsf{fir} | \mathsf{sec} \Omega\}$:

   1 令 $\Omega \leftarrow \Omega - \{\mathsf{fir}, \mathsf{sec}\}, s \leftarrow s + \{(\mathsf{fir}, \mathsf{sec})\}$, 调用 generate().

   2 令 $\Omega \leftarrow \Omega + \{\mathsf{fir}, \mathsf{sec}\}$

2.1.4 2.1.3 的复杂度分析

复杂度不会变化, 时空复杂度均为 $\mathrm{O}(n!!)$.

3 分治法构造循环赛日程表

3.1 实现

考虑分治算法 (div_con(n)).

div_con(n) return: matrix $T_{n\times n}$ *or* $T_{(n+1)\times(n+1)}$:

1 如果 $n=2$:

1 返回 $T \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

2 如果 n 是奇数:

1 返回 div_con($n+1$).

3 如果 n 是偶数:

1 令 $T_s \leftarrow \text{div\_con}(\frac{n}{2})$

2 如果 $\frac{n}{2}$ 是偶数:

1 返回 $T \leftarrow \begin{pmatrix} T_s & T_s+\frac{n}{2}J \\ T_s+\frac{n}{2}J & T_s \end{pmatrix}$.

3 如果 $\frac{n}{2}$ 是奇数:

1 令 $T_s \leftarrow \begin{pmatrix} row_1 T_s \\ row_2 T_s \\ \vdots \\ row_{\frac{n}{2}} T_s \end{pmatrix}$

2 令 $U \leftarrow T_s, V \leftarrow T_s+\frac{n}{2}J$.

3 对于每一个 $i\in[1,\frac{n}{2}]$, 令 $U_{i,\frac{n}{2}-i+2} \leftarrow i+\frac{n}{2}, V_{i,\frac{n}{2}-i+2} \leftarrow i$.

4 令 $P = \begin{pmatrix} 1 & 2 & \dots & \frac{n}{2}-1 \\ 2 & 3 & \dots & \frac{n}{2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{n}{2} & \frac{n}{2}+1 & \dots & \frac{n}{2}+\frac{n}{2}-2 \end{pmatrix} \pmod{\frac{n}{2}} + (\frac{n}{2}+1)J$

5 令 $Q = \begin{pmatrix} 0 & -1 & \dots & 2-\frac{n}{2} \\ -1 & -2 & \dots & -\frac{n}{2}+1 \\ \vdots & \vdots & \ddots & \vdots \\ 1-\frac{n}{2} & -\frac{n}{2} & \dots & 3-\frac{n}{2}-\frac{n}{2} \end{pmatrix} \pmod{\frac{n}{2}} + J$

6 返回 $T \leftarrow \begin{pmatrix} U & P \\ V & Q \end{pmatrix}$.

因为是构造性算法, 故不作证明.

3.2 算法复杂度

空间复杂度为 $O(1)$(输入数组可重复利用空间), 时间复杂度为 $T(n) = T(n/2) + \mathrm{O}(n^2) = O(n^2)$.

4 DanceLinkX 算法构造循环赛日程表

构造矩阵 $P_{\frac{n(n-1)^2}{2} \times 2n^2}$.

对于每一个三元组 $t = (x, y, z), 0 \leqslant x \leqslant n-2, 0 \leqslant y < z \leqslant n-1$ 构造矩阵第 $f(x,y,z) = x \cdot n^2 + y \cdot n + z + 1$ 行,x, y, z 分别代表行号, 列号, 值.

1. 第 $y \cdot n + z + 1$ 列为 1.

2. 第 $z \cdot n + y + 1$ 列为 1.

3. 第 $n^2 + x \cdot n + z + 1$ 列为 1.

4. 第 $n^2 + x \cdot n + y + 1$ 列为 1.

设 $T' = \begin{pmatrix} t_1 & t_2 & \dots & t_k \end{pmatrix}$ 是 DanceLinkX 算法求出的 P 的精准覆盖.

Theorem 8 (1) f 是双射.

(2) $k = n(n-1)$.

(3) $T = \left[T_{f^{-1}(t_i).x, f^{-1}(t_i).y} = f^{-1}(t_i).z, T_{f^{-1}(t_i).x, f^{-1}(t_i).z} = f^{-1}(t_i).y \right]_{n \times n}$ 是循环赛日程表的第 $2 \sim n$ 行.

证明: 不对 (1), (2) 作说明, 现在来说明 (3).

假设第 $i (2 \leqslant i \leqslant n)$ 行出现了重复的数字, 设出现在第 y_1, y_2 列, 值为 z. 那么根据构造第 $n^2 + i \cdot n + z + 1$ 列中会同时有 $f(x, y_1, z), f(x, y_2, z)$ 两行被选中, 所以 P 不是精准覆盖.

假设第 $j (1 \leqslant j \leqslant n)$ 列出现了重复的数字, 设出现在第 x_1, x_2 行, 值为 z. 那么根据构造第 $y \cdot n + z + 1$ 列中会同时有 $f(x_1, y, z), f(x_2, y, z)$ 两行被选中, 所以 P 不是精准覆盖.

假设第 (i,j) 格中填入了多余一个数字, 设填入了 z_1, z_2, 根据构造第 $n^2+i\cdot n+y+1$ 列中会同时有 $f(i,j,z_1), f(i,j,z_2)$ 两行被选中, 所以 P 不是精准覆盖.

假设第 (i,j) 格中没有填入数字, 那么显然第 $n^2+i\cdot n+j+1$ 没有被覆盖 (实际上存在 1), 所以 P 不是精准覆盖.

附注:

$$\sum_{i=0}^{n-2}\sum_{j=0}^{n-1}\sum_{k=j+1}^{n-1} 1 = \sum_{i=0}^{n-2}\sum_{j=0}^{n-1}(n-1-j) = \sum_{i=0}^{n-2}\left[n(n-1) - \frac{n(n-1)}{2}\right] = \frac{n(n-1)^2}{2}$$

4.1 复杂度分析

DanceLink 仅仅是一种特殊的回溯结构, 时间复杂度仍为 $O(n!)$, 空间复杂度为 $O(n^5)$.

5 网络流模型构造循环赛日程表

6 总结

本实验选取了几种比较经典的算法构造模型实现构造循环赛日程表. 第一节给出了构造性的思路, 为第二节证明做铺垫. 第二节给出了朴素的搜索算法. 第三节给出了分治算法, 第四节给出了带数据结构优化的搜索算法.

轮转法一分钟能够构造出 80000(47.329s) 大小的日程表.

朴素搜索算法一分钟能够构造出 20(第一次优化能构造 18 长度 (6.315s),20 长度爆栈, 第二次优化 44.463s, 第三次优化 42.154s) 大小的日程表.

分治法一分钟能够构造出 65000(34.040s) 大小的日程表.

DanceLinkX 比较特别, 首先它的空间复杂度比较高, 其次在 $2m$,m 为奇数的情况下, 最多构造 26 大小的日程表, 剩下的情况, 它能构造 92(0.479ms, 更高的会爆栈) 大小的日程表. 所以最终认定它能构造 28 大小的日程表.