flare解题报告

福州一中　卓亮

April 27, 2012

Contents

List of Figures

List of Tables

1 题意简述

对于无向图$G_0 = (V_0, E_0), G_1 = (V_1, E_1)$。我们称$G_1$是由$G_0$变换得到的，当且仅当

- 每个G_1的顶点和G_0的一条边一一对应；
- G_1的一对顶点间有一条边，当且仅当在G_0中的对应边有一个公共顶点。

任务：已知G_1，求一个可能的G_0，或者宣告这样的G_0不存在。

2 命题思路

CodeChef上有一道题：已知$G_0 = (V_0, E_0)$。G_0变换一次得到G_1，G_1变换一次得到G_2，G_2变换一次得到G_3。求G_3的点数和边数。其中$|V_0|, |E_0| \leq 1000$。[1]

我们设$G_i = (V_i, E_i)$。最直接的想法是，直接求一个合法的G_3，然后返回它的点数和边数。这显然可以分成3步。第一步，根据G_0求G_1；第二步，根据G_1求G_2；第三步，根据G_2求G_3。

不失一般性，我们考虑如何根据G_0求G_1。

我们对G_0的边编号，例如1到$|E_0|$。在G_1中建立$|E_0|$个点。接着，枚举两条边e_1, e_2，若e_1, e_2有一个公共顶点，那么我们在G_1中，e_1, e_2对应的点间连一条边。容易看出，这个做法的复杂度是$O(|E_0|^2)$的。

如果直接执行本算法3次，算法的复杂度可能是$O(|E_0|^8)$。对给出的数据范围，显然无法承受。

不过这个复杂度似乎无法达到。注意到$|E_1|$有可能达到$\Theta(|E_0|^2)$。图(1)就指出了这样一种情形。$|E_2|$看起来就很难达到$\Theta(|E_1|^2)$，更难达到$\Theta(|E_0|^4)$了。这是因为，变换以后边数平方的图，看起来不大可能由一个图变换而来。什么样的图才能由一个图变换而来呢?

虽然如此，这个算法的时间复杂度显然高于$O(|E_0|^2)$，因而我们不能直接使用这个算法。

另一种想法是直接计算点数和边数。

[1] http://www.codechef.com/problems/BEARS. 命题者Gennady Korotkevich.

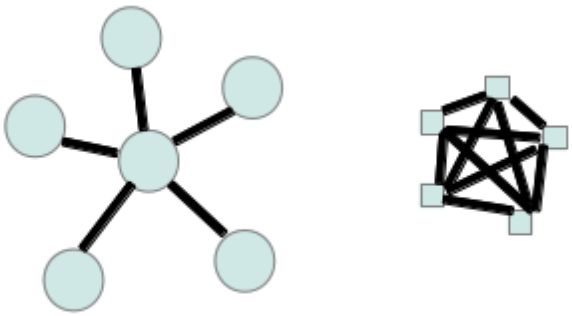

Figure 1: 一个$E_1 = \Theta(|E_0|^2)$的情形

首先看如何根据G_0，求G_1的点数和边数。显然$|V_1| = |E_0|$，而G_0中凡是有一个公共顶点的边对，会对$|E_1|$贡献1。

改变枚举的对象，考虑公共顶点i。设点i的度为d_i。由于i连出的每对边，给答案贡献1。因而

$$|E_1| = \sum_{v \in V_0} \mathrm{C}_{d_v}^2 \tag{1}$$

再来看如何求G_2的点数和边数。显然$|V_2| = |E_1|$。G_2中的点，在G_0中对应一个共顶点的边对，如图(2)。

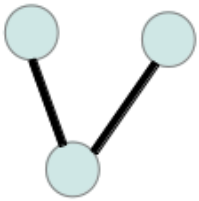

Figure 2: G_2中的点对应至G_0

G_1中的点对应G_0的一条边。因而G_2的边，对应于一对边对，这两个边对共享一条边。也就是说，G_2的边，对应于连通的3条边。有如图(3)所示的两种情况。

求$|E_2|$，可以分别考虑这两种情况。对于第一种情况，可以枚举图中绿色的边(u, v)。一条连着u的边，一条连着v的边，再搭配上边(u, v)，就对答案贡献一。对于

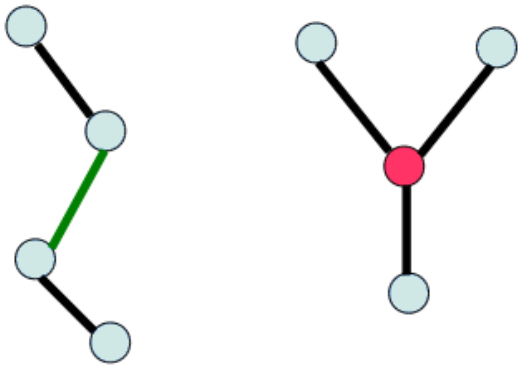

Figure 3: G_2中的边对应至G_0

第二种情况，可以枚举图中红色的点v，与v连着的3条边对答案贡献一。形式化地描述，就是

$$|E_2| = \sum_{(u,v)\in E_0} (d_u - 1)(d_v - 1) + \sum_{v\in V_0} \mathrm{C}_{d_v}^3 \tag{2}$$

结合这两个想法，我们实际上解决了这个问题。算法包含以下两步。

1. 根据G_0求一个合法的G_1。这一步的时间复杂度为$O(|E_0|^2)$。
2. 根据G_1直接计算G_3的点数和边数。这一步的时间复杂度是$O(|V_1| + |E_1|) = O(|E_0|^2)$。

本人在解决此题的过程中，想到，这个问题的逆问题是否可以解决呢？在研究逆问题过程中，发现这是一个很好的问题，它涉及到图论知识的综合应用，且能够给选手逆思维方面的启迪。于是本题就出现了。

3 考查点

本题是一道图论问题，注意考察选手对图论知识的综合运用。本题涉及了团、补图、二分图，以及图的遍历等知识。这些知识都是一个具有NOI水平的选手应当掌握的。同时，本题需要选手深入细致的分析，从而得到完整的解决方案。

4 问题分析

原问题变为逆问题后，遇到的最直接的障碍就是“谁对应谁”。

已知的图是G_1，而G_1的每个点对应于G_0的一条边。我们可以假设G_1的每个点v，对应于G_0的一条边e_v。如果我们把假想中的G_0标号，e_v连接着的是x_v与y_v。那么，相当于我们给G_1的每个点v带上两个标号$(x_v, y_v)(x_v \neq y_v)$。而变换的要求，相当于对任何相邻的两个G_1中的点u, v，以下4个条件恰好满足一个。

- $x_u = x_v$
- $x_u = y_v$
- $y_u = x_v$
- $y_u = y_v$

而任何两个不相邻的点u, v，这4个条件均不满足。

这样，容易想到一个最简单的方法。对含有n个点的G_1，每个点需要一对标号，因而我们可以直接搜索每个点的标号是什么。注意到最多只会有$2n$个可以填的空，因而，至多只需要$2n$个数。之后，我们可以检查一下搜索结果是否符合要求。最朴素的检查方法，即枚举两个G_1的点，判断是否满足要求。因而直接搜索的复杂度是$O((2n)^n \cdot n^2)$的。由于这个算法时间复杂度实在过高，因而期望得分是0分。

这个算法未能解决问题的原因是，它只对问题进行了大致的描述。容易发现，如图(4)所示的情形，一张图可能有两种不同的标号，但它们的效果完全相同。这就说明，搜索存在冗余。

改进方法便是，对于效果相同的标号方式，只搜索一次。实际上，我们只关心哪些数字相同，而不在乎数字具体是几。因而，可以采取的搜索方法是，记录之前用过几个数字，对于当前的空位，要么填上之前用过的数字中的一个，要么写一个新的数字。

我们来计算这个算法最多会访问多少状态。如果用$S(i, j)$表示填i个空，用了j个数字。那么$S(i, j) = S(i-1, j-1) + jS(i-1, j)$。[2] 设$B_n = \sum_{i=1}^{n} S(n, i)$ [3]，那么这个算法的时间复杂度是$O(B_{2n} \cdot n^2)$。由于这个算法复杂度仍然很高，因而期望得分仍是0分。

[2] $S(i, j)$是第二类Stirling数。第二类Stirling数的定义及相关性质可参考[2]。

[3] B_n是Bell数。Bell数的定义和性质可参考[3]，这个数列的前几项可以在[4]看到。

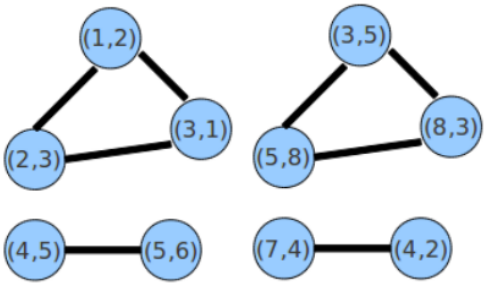

Figure 4: 效果相同的两种对G_1的标号

容易看出，简单变换不改变连通性。一张图可以分成若干连通块。如果我们能处理清楚每一个连通块，那么问题就得到了解决。故下文中，如无特别说明，均只考虑G_1是一个连通块的情形。

注意到搜索结果需要满足要求，而上述搜索方法在搜索过程中没有利用到这些要求。因而可以考虑由此下手。如果一个G_1的点被标为(a,b)，那么根据要求，与它相邻的点的标号的一个数，需要是a或者b。这个优化，可以使得搜索的状态数减少到$O(2^{n-1}B_{n+1})$。再配合上一些显然的性质，如同一点的2个标号不能相同，同一点的2个标号的顺序无关，可以使得这个状态数难以达到。算法的复杂度降至$O(2^{n-1}B_{n+1}\cdot n^2)$。期望得分是10分。

进一步考虑。一开始，选择一个G_1点v，直接标号为(a,b)。然后，与v相邻的点的一个标号，要么是a，要么是b，我们可以搜索这些点的一个标号。接下来每一步，选择一个只确定了一个标号的点，搜索它的另一个标号。有两种情形。一种是新建一个标号，另一种是选一个已经有的标号。注意到选择已经有的标号，最多的可能只有1种。这是因为，选择已经有的标号，只能从和它相邻的点中选。而如果可选的有两种，会导致不符合要求。因而，利用这个发现，搜索的状态数只有$O(2^n)$。因而这个算法的时间复杂度是$O(2^n\cdot n^2)$，期望得分20分。

我们总是先搜索一个状态，再判断是否可行。而两个状态，如果它们相同的部分很多，重新判断一次就重复了对相同部分判断。因而此处存在冗余。如果我们边搜索边判断，就能减少冗余。利用这个方法，可以使上述算法的复杂度降低至$O(2^n\cdot n)$，期望得分30分。

进一步分析，可以发现如果G_1是一些特殊图，如：链、环、完全图。我们可以很容易地看出G_0的模样。链变换以后还是链，环变换以后还是环。如果一张图是形如一个点连出若干条边，那么它变换以后就是完全图。因而对这些情况，可以直接通过构造获得。如果分析到这一点，期望得分20分。

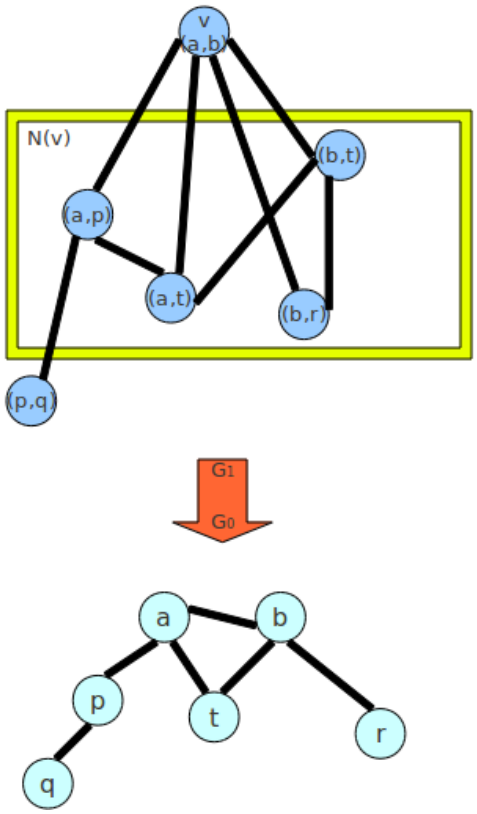

Figure 5: 算法示例

要更好地解决问题，需要更进一步分析性质。如图(5)所示，抓取一个G_1点v，它在G_0中是一条边，假设两端点分别为(a,b)。在G_1中和v相邻的点，在G_0中对应的边，要么连着a，要么连着b。根据简单变换的规则，凡是在G_0中连着同一个点的边，在G_1中都有边相连，因而这些在G_0中的共点边，在G_1中形成一个团。因此，v的邻域$N(v)$的导出子图有最多两个团，外加一些“杂边”。

对于杂边，由图(6)可以观察出，一个点不能连出两条杂边。否则就会出现重边

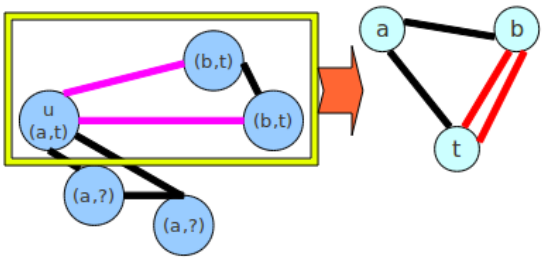

Figure 6: 一个点不能连出两条杂边示例

了。[4]

注意到$N(v)$的导出子图$G_1[N(v)]$最多含有两个团。如果取一个补图，就是二分图了。

在$N(v)$中随便选一个初始点u，让它的一个标号是a。 在补图中，利用二分图染色法，尝试确定每个点的一个标号是a还是b。如果不是二分图，说明无解。否则一些点会被确定。图根据标号是a还是b，被分为X, Y两部。接下来看没有被确定的点。

分几种情况讨论。

$|X| = 1, |Y| = 0$。即没有点的一个标号是b。这是一种十分特殊的情况。这意味着，在$\overline{G_1[N(v)]}$中，没有一个点与u相邻。应当考虑换一个点作为初始点。如果无法找到在$\overline{G_1[N(v)]}$中度大于0的点，说明在$G_1[N(v)]$，任两点都相邻。亦即，$N(v)$是一个团。如果未确定的点数大于等于2，说明所有的点都应该有标号a。这是根据对杂边的观察。可以观看图(7)来更加清晰地认知。

$|X| = 1, |Y| > 1$，或$|X| > 1, |Y| = 1$。即有一部只有1个点，剩下的一部有超过1个点，如图(8)所示，那么未确定的点应该归大于1个点的那部。

$|X| > 1, |Y| > 1$。即每部都大于1个点。如图(9)所示，如果还有未确定的点，就无解。

$|X| = 1, |Y| = 1$。这个情况表明每部都恰有一个点。图(10)显示了此情况。如果未确定的点大于2个，那么无解。

[4]要说明这一点，不妨假设连出杂边的点是u，其编号为(a, t)，与其通过杂边相邻的点的编号必有一个标号是t。而由于这些点都与v相邻，而且与u不同属一个团，因而必有一个标号是b。故与u通过杂边相邻的点的标号只能是(b, t)。

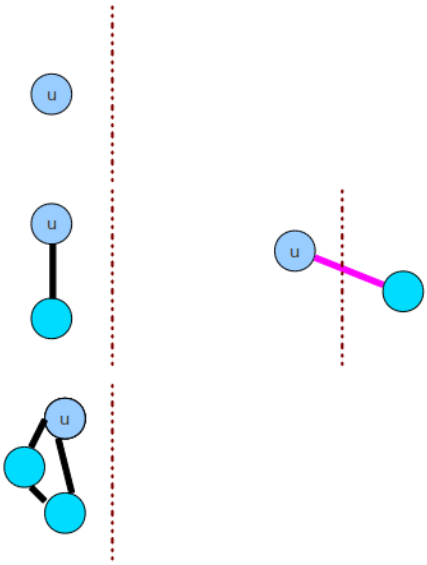

Figure 7: $|X| = 1, |Y| = 0$情形的讨论

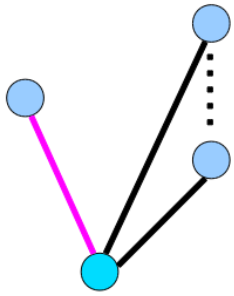

Figure 8: $|X| = 1, |Y| > 1$或$|X| > 1, |Y| = 1$情形的讨论

这样未确定的点最多2个。我们枚举它们属于哪一部。然后采用BFS，确定其余点的标号。最后检查是否合法即可。

这里说明在算法一开始抓取点v的时候的问题。如果抓取度数最小的点，是没有问题的。如果这个点的度数是d，容易看出，$nd \leq 2m$，因而$d \leq \frac{2m}{n}$。

如果随意抓取点是否有问题呢？事实上我们可以用一些不等式来确定范围。我们希望补图的边尽量多，而原图的边尽量少，因为我们有两个团，所以边根本少不到哪里去。亦即$\frac{1}{2}x(x-1) + \frac{1}{2}(d-x)(d-x-1) \leq m$，这样$d \leq \sqrt{2m}$，也就是说，度数不能太

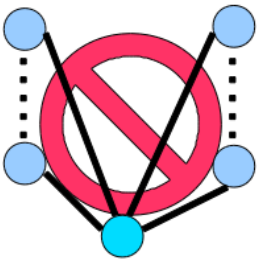

Figure 9: $|X| > 1, |Y| > 1$情形的讨论

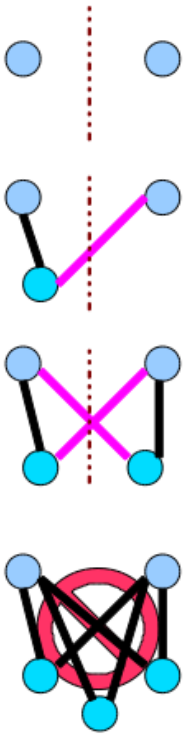

Figure 10: $|X| = 1, |Y| = 1$情形的讨论

大。这个观察使得求补图的时间复杂度是$O(m)$的。

接下来说明如何BFS。BFS的目的是确定G_1每个点的标号，代表G_0中对应的边。一开始所有点的标号都形如$(?, ?)$。被抓取的点标号已确定。与被抓取点相邻的点，标号确定了一半，有的是$(a, ?)$，有的是$(b, ?)$。每一次，选一个标号已经确定了一半的点$(x, ?)$。查看是否存在与它相邻的，标号确定了一半的点$(y, ?)$。如果存在，那么这

两个点的标号都应该是(x,y)。与该点相邻的，完全未确定的点$(?,?)$，改为$(y,?)$。否则，我们选一个新的数z，将其标为(x,z)。与该点相邻的，完全未确定的点$(?,?)$，改为$(z,?)$。这样，就能在$O(n+m)$的时间内确定每个点的标号了。

最后说明如何检查合法性。首先，我们检查是否有重边，自环。接着，我们检查求得的G_0生成的G_1的边数。这一步，只要计算每个点的度数d_i，然后$\sum \mathrm{C}_{d_i}^2$ 即是答案。最后，我们检查在G_1中相邻的两个点在G_0中对应的两条边是否在G_0中有公共顶点。

这是这个问题的参考解法，其时间复杂度是$O(n+m)$，期望得分100分。

5 数据生成方法

本题有20%的数据是特殊图（链、环、完全图）。其中15%的数据只有一个连通块。5%的数据有3个连通块，分别是链、环、完全图。

其他的数据如何生成呢？如果直接随机生成G_1，那么将会有很大概率出现无解。

一种很简单的方法是，随机生成G_0，然后，将G_0变换成G_1。

这种方法虽然十分简单，但不能完全体现本题的复杂性。如果在算法中漏考虑了一些情况，仍然可以通过随机生成的数据。

解决的办法是，本人手画了一些，和用程序生成了一些特殊情况的数据，使得需要考虑到特殊情况。这些数据并不是很大。用随机的方法生成了数据之后，我们对其中大部分数据配上1到2组特殊数据，这样就使得选手需要充分考虑了各种情形才能够通过本题。

除了使用随机方法生成G_0，本人还考虑了一些其他的图，例如毛毛虫、树、环套树、二分图等。部分测试数据的G_1就是由这样的图生成的。[5]

6 选手得分估计

NOI选手估计的得分如表(1)所示。NOI铜牌选手应当能够发现特殊图形的还原方法，利用构造法获得20分。而具有国家集训队水平的选手应当不仅能发现特殊图形的还原方法，还能使用搜索算法获得50分。而更少的选手能够分析到问题的不同程度，以及利用贪心随机化等方法获得70分或者更高的分数。

[5]针对国家集训队选手与NOI水平选手，出了两套有所区别的测试数据。

人数/总人数(百分比)	分数
5%	≥ 70
25%	≥ 50
65%	≥ 20

Table 1: 得分估计

本题对选手综合分析能力要求比较高，估计NOI水平选手的平均分大致在20分左右。

7 后记

集训队测试后，顾昱洲同学发现，变换后的图有一个名字，叫做Line graph。他给出了一个链接[6]，和一篇文章的名称[7]。这使本人大为惊讶。

在链接中，提到了有人曾对此进行研究，并以线性时间解决了它，但没有指明具体的做法。这篇文章，虽然有名字，但本人和他都没有找到免费的查看或下载的途径，因而其做法不得而知。因而无法比对本文所指出的做法与其做法的异同。本人正在寻求该论文，探讨该问题有否更好的解法。

[6] http://en.wikipedia.org/wiki/Line_graph

[7] Roussopoulos, N. D. (1973), "A max m,n algorithm for determining the graph H from its line graph G", Information Processing Letters 2 (4): 108–112

References

[1] *February 2012 Cook-off Problem Editorials*[EB/OL]. http://www.codechef.com/wiki/february-2012-cook-problem-editorials.

[2] *Stirling number*[EB/OL]. [2012-3-24]. http://en.wikipedia.org/wiki/Stirling_number.

[3] *Bell number*[EB/OL]. [2012-3-20]. http://en.wikipedia.org/wiki/Bell_number.

[4] *Bell or exponential numbers: ways of placing n labeled balls into n indistinguishable boxes.*[EB/OL]. [2012-4-17]. http://oeis.org/A000110.

Index

Appendices

A flare完整题面

Time Limit: 7s, Memory Limit: 128MB

这是一个很有趣的想法，如果熊是蜜蜂，
他们会将他们的巢建在树下。
而且如此的话（如果蜜蜂是熊），
我们就不用爬上楼梯了。

这是小熊维尼的一首充满抱怨的歌。你是否曾经想过，有些关于无向图的问题，如果把边看成点，点看成边，会更容易解决？这道题目正与此有关。

假设有一张无向图G_0，让我们对G_0执行一次简单变换，来得到无向图G_1。G_1满足，每个G_1的顶点和G_0的一条边一一对应，G_1的一对顶点间有一条边，当且仅当在G_0中的对应边有一个公共顶点。

可能与你猜测的不一样，本题给出了G_1，求一个满足条件的G_0。

Task: flare

Input: stdin

输入的第一行含两个数n, m，代表点数和边数。接下来m行，每行有两个数$a, b(1 \leq a, b \leq n)$，表示a, b间有一条边。保证每条边连接了两个不同的顶点，每对顶点最多有一条边相连。

Output: stdout

如果这样的G_0不存在，输出"-1"。否则，输出n行，每行两个数，代表G_0的一条边的两个顶点。G_0的第i条边对应于G_1的顶点i。输出需要满足，每条边连接了不同的顶点，每对顶点间至多只有一条边。你可以自行给顶点标号，但顶点的标号需要是正整数，而且不应超过10^9。

Sample Input	Sample Output
3 3 1 2 1 3 2 3	1 2 1 3 2 3

Hints

样例的图是“三角形”（三个点，两两间均有边）。容易看到“三角形”简单变换后仍是“三角形”。当然，样例的解并不唯一。

Constraints

对10%的数据，$1 \leq n \leq 10$。

对20%的数据，$1 \leq n \leq 15$。

对30%的数据，$1 \leq n \leq 20$。

另有5%的数据，图为一条链；5%的数据，图为一个环；5%的数据，图为完全图；5%的数据，图的每个连通块为上述三种情形之一。

对70%的数据，$1 \leq n \leq 10^3$。

对100%的数据，$1 \leq n, m \leq 10^6$。

B Glossary of graph theory

图由点集V和边集E构成。点对(u, v)称为边。

图G的点集可以用$V(G)$或V来表示。一张图的阶就是点的个数，表示成$|V(G)|$。图G的边集可以用$E(G)$或E来表示。一张图的大小就是边的数目，表示成$|E(G)|$。

图G的补图$\overline{G}$，和G有相同的点集，而边集为$\{(x, y)|(x, y) \notin E(G)\}$。

一个点v的度$d_G(v)$是与v有关的边的数目。一张图的总度数等于边数的两倍。

图G的团是两两有边的点集。

C Usage of the program

本题给出的参考程序具有多种功能。下面介绍在Linux平台下的用法。

获得本题的解。运行./flare，然后用屏幕输入数据，程序将在屏幕输出结果。如果想从<infile>中读取文件，输出到<outfile>中，可以用命令./flare < <infile> > <outfile>。

作为清橙测试器的比较程序。清橙测试器会执行./flare <in> <out> <ans> <log>，4个参数分别是读入文件，选手输出文件，标准输出文件，结果文件。

检查答案是否合法。对一组输入输出文件<infile> <outfile>，运行./flare -check <infile> <outfile>，将判断输出文件是否能对应输入文件。你也可以运行./flare -check <number>，来检查文件名为flare<number>.in和flare<number>.out。

生成数据。运行./flare -make <number>，将生成一组名为flare<number>.in和flare<number>.out的数据。