算法原理及测试环境搭建说明

1 算法原理

1.1 问题重述

本题要求祭品在 A、B 两部落的朋友间互相传递，祭品传递过程中除祭品发起人外，其他人不能重复接收该祭品，祭品再次回到祭品发起人手中时该祭品传递结束。求经手 4 人,6 人,8 人,10 人,12 人,14 人（含祭品发起人）的祭品分别最多能有多少个，对于经手人员相同，只是传递顺序不同的祭品，均算作 1 个有效祭品。

1.2 问题抽象

如果把 A、B 部落的每个成员都看成一个节点，则 A、B 部落的朋友关系，可以理解为 A 类节点和 B 类节点的连接情况，A 类节点内部、B 类节点内部不存在连接情况，祭品只能在有连接情况的节点间传递。以经手 14 个人的祭品为例，可以理解为连接了 14 个节点的环路。对应 1.1 中的问题重述，环路连接过程中，除起点外，其他节点不允许重复经过，这种环路称为简单环路，之后描述的环路不做特别说明均指简单环路。经手 14 人的祭品最多能有多少个，则可理解为不同的 14 节点环路最多能有多少个，对于经过节点相同，只是连接顺序不同的环路均算作是 1 个有效环路。

假设 A，B 部落成员节点分别有 M 个和 N 个，对应 CSV 数据表格的行数与列数，为它们统一编号：A 部落成员节点 ID 为 0~（M-1），B 部落成员节点 ID 为 M~（M+N-1）。则根据 CSV 文件可以得到这些节点间的连接情况，进而构建得到一张无向图。问题抽象为根据给定无向图，求其中节点长度为 4,6,8,10,12,14 的环路分别有多少。

1.3 算法原理

根据前述分析，本题抽象为根据给定无向图，求其中节点长度为 4,6,8,10,12,14 的环路分别有多少。

由于寻找的是环路，环路中的每个节点都可以作为起点，为此设环路的起点都从 A 类节点开始，对应节点 ID 为 0~（M-1）。因此依次从编号为 0~（M-1）的节点出发，遍历寻找节点长度为 4,6,8,10,12,14 的环路分别有多少。

直接进行环路的遍历寻找，当搜索的环路较长时，如 12、14 环，会导致搜索的速度较慢，为此采用先搜索单向路径再进行组合拼接的方式进行环路的间接遍历寻找。如 14 节点环路，可以先遍历搜索全部的 8 节点单向路径，如果其中 2 条单向路径起止点相同，且中途没有重合点，那么这 2 条路径可以组合拼接成 1 条 14 节点环路，这样可以从一定程度上提高搜索长节点环路的速度。

假设从 i 号节点出发，完成了全部的 4,6,8,10,12,14 节点环路的遍历寻找，下一轮则从（i+1）号节点出发继续遍历寻找满足长度要求的环路。由于含有 i 号节点的环路在上一轮遍历中已经全部找出，因此从（i+1）号节点出发继续寻找满足长度要求的环路时，应将 i 号节点排除在环路节点的搜索范围之外。

对于经过遍历寻找初步得到的满足长度要求的环路，多条环路可能由相同的节点组成，只是连接顺序不同而已，这些环路按照赛题要求只能算作一条有效环路。因此对于遍历寻找后初步得到的满足长度要求的环路，还需要进一步去重，计算出其中的有效路径数量。

算法步骤如下：

（1）设置最大搜索深度 MAXDEEP 为 8 节点（通过修改 msic.h 文件中的 MAXDEEP 宏定义实现），i=0；

（2）从节点 i 出发，基于图的 DFS 深度优先遍历，在搜索深度不超过 MAXDEEP 条件约束下，不断遍历图中的节点，在遍历的过程中将节点长度为 3,5,6,7,8 的单向路径保存到对应长度的 route**m**_flexarry (**m** 取 3,5,6,7,8)二维路径数组中。route**m**_flexarry 二维路径数组的每一行存储一条长度为 **m** 的单向路径的全部节点，route**m**_flexarry 二维路径数组的行数代表长度为 m 的环路的条数；

（3）i++，如果 i<M 成立，转步骤（2），否则转步骤（4）；

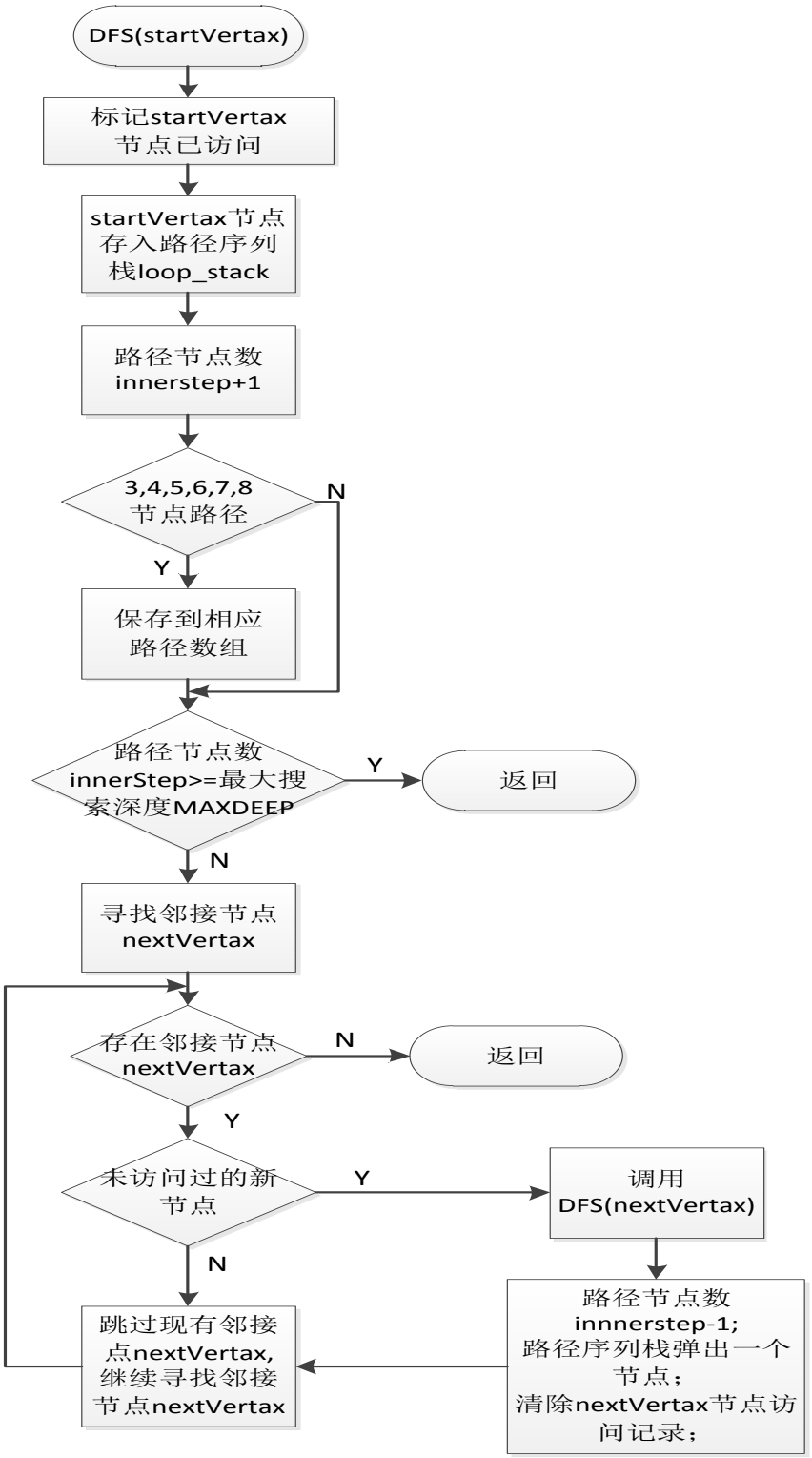
（4）长度为 **m** (**m** 取 3,5,6,7,8)的 route**m** _flexarry 二维路径数组中含有大量相同起止点的单向路径。通过比较 2 条单向路径的起止点，得到 2 条单向路径的大小关系，进而对 route**m** _flexarry 二维路径数组的全部行（一维数组）使用堆排序法，则所有长度为 **m** 的单向路径按照升序排列后重新存入 route**m**_flexarry 二维路径数组。经过本步骤，相同起止点的单向路径将被相邻排列在一起；

（5）遍历长度为 **m** 的 route**m** _flexarry 二维路径数组，进行单向路径的组合拼接。其中起止点相同且中途没有重合点的 2 条长度为 **m** 节点单向路径可以组合拼接为 1 条长度为 **n=2m-2** 的环路。拼接得到的长度为 **n** 的环路保存到对应长度的 cycle**n**_combinedbyroute_flexarry (**n** 取 4,6,8,10,12,14)二维环路数组中。cycle**n**_combinedbyroute_flexarry 二维环路数组的每一行存储一条长度为 **n** 的环路上的全部节点（起点不重复存储），cycle**n**_combinedbyroute_flexarry 二维环路数组的行数代表长度为 **n** 的环路的条数；

（6）将步骤（5）得到的每条环路，使用堆排序法，将节点 ID 升序排列后重新存入 cyclen_combinedbyroute_flexarry 二维环路数组。经过本步骤，相同节点组成只是连接顺序不同的环路将存储为相同的一维数组序列；

（7）长度为 **n** 的 cyclen_combinedbyroute_flexarry 二维环路数组中含有大量相同节点组成的重复环路。通过从高位到低位（大端模式）比较 2 个长度为 **n** 的一维数组，得到这 2 个一维数组间的大小关系，进而对 cyclen_combinedbyroute_flexarry 二维环路数组的全部行（一维数组）使用堆排序法，将所有长度为 **n** 的环路按照升序排列后重新存入 cyclen_combinedbyroute_flexarry 二维环路数组。经过本步骤，相同节点组成只是连接顺序不同的环路将被相邻排列在一起；

（8）对于整体排好序的 cyclen_combinedbyroute_flexarry 二维环路数组，使用 1 个基准指针和 1 个游标指针遍历得到其中的非重复行数量，非重复行数量便是长度为 **n** 的有效环路的数量。对应原始问题中经手 **n** 个人的祭品最多有多少个。



从 startVertax 节点出发搜索单向路径的算法流程图

2 测试环境搭建

2.1 测试工具

本算法对应的源码已通过集成开发工具 Visual Studio 2015 编译与测试 。

2.2 测试步骤

（1） 从中兴捧月算法精英比赛傅里叶赛道下载丰收祭游戏的当日计算数据，下载地址：https://ztechallenge.nowcoder.com/activity/zte2020/5

（2） 将下载得到的 Example.csv 文件复制到与工程文件 cyclefinding_graph.vcxproj 同级的目录下，即 cyclefinding_graph 目录，注意需要保证文件名必须为 Example.csv，否则会导致无法读取数据文件；

（3） 点击工程文件 cyclefinding_graph.vcxproj，使用 Visual Studio 2015 集成开发工具打开整个工程；

（4） 使用 Visual Studio 2015 集成开发工具重新编译整个工程，编译模式为 Release ，平台为 X64；

（5） 编译后可以选择直接在 Visual Studio 2015 中按 Ctrl+F5 开始执行程序；

（6） 控制台显示中间过程提示信息及最终结果，在当前工程目录 cyclefinding_graph 目录下的 result.txt 保存有计算的最终结果和程序运行时间。

（7） 编译后在 cyclefinding_graph 目录下同时生成可执行文件 cyclefinding_graph.exe，该可执行文件可以脱离 Visual Studio 2015 直接运行。如果要直接运行该 exe 文件，需要保证 cyclefinding_graph.exe 的同级目录中具有 Example.csv 文件，否则将导致无法找到数据文件。
经过前述步骤（2）的准备，cyclefinding_graph.exe 同级目录已经存在 Example.csv 文件。

（8） 直接运行 cyclefinding_graph.exe 后，控制台显示中间过程提示信息及最终结果，在同级目录下生成 result.txt 文件，其中保存有计算的最终结果和程序运行时间。

2.3 注意事项

（1）由于笔者现用计算机硬件配置较低（i3-3227U 1.9GHZ + 4G），搜索 7、8 节点单向路径（对应 12、14 节点环路）时，搜索与组合拼接时间较长。对于笔者现有计算机硬件，当配置最大搜索深度 MAXDEEP 为 6 节点时，可计算得到节点长度为 4,6,8,10 的环路数量，用时约 26 秒钟；当配置最大搜索深度 MAXDEEP 为 7 节点时，可计算得到节点长度为 4,6,8,10,12 的环路数量，用时约 23 分钟。最大搜索深度 MAXDEEP 可以修改 msic.h 文件中的 MAXDEEP 宏定义进行配置。

测试时可以根据计算机硬件配置情况，通过修改 msic.h 文件中的 MAXDEEP 宏定义，修改最大搜索深度为 7 或 8，尝试计算得到全部环路数量。

（2）直接运行 cyclefinding_graph.exe 文件时，需要保证 cyclefinding_graph.exe 的同级目录中存在 Example.csv 文件，否则将导致无法找到数据文件。