4 维护服务集群

Requirement

1. 解释代码逻辑和自己使用的方法思路。
2. 分析问题中算法的复杂度。

在机器学习集群中，用户向服务厂商提供机器学习任务的数据，代码和需要的服务器数量，

这些内容我们称为一个任务(job)。

对每个任务，服务厂商为其分配合适的服务器在云端执行。

注意：我们假设任务之间不共享服务器。

服务器的更换（必做）

机器学习服务器的异常是集群运维最常见的问题，

例如GPU算力下降，GPU过热，PCIe连接老化等，异常的服务器计算和处理速度都会下降。

异常的服务器和正常服务器一同执行机器学习任务时将会拖慢任务的计算。

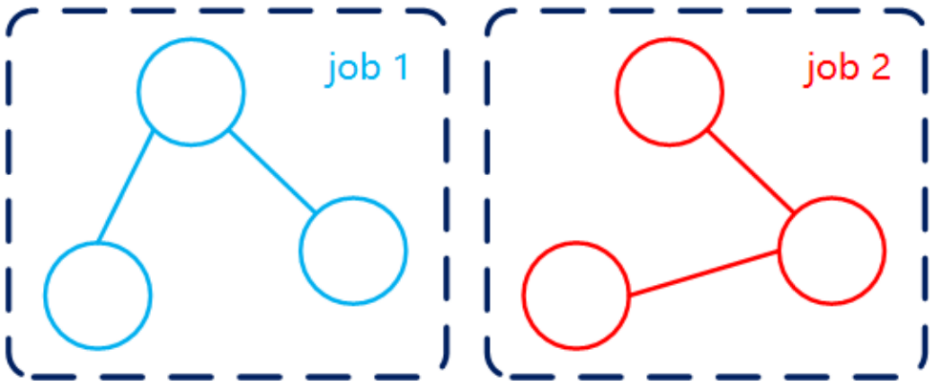
服务厂商在必要时会选择替换异常的服务器来保持部分任务的计算速度。

给出一个由 n 个节点组成的服务器网络，用 n × n 个邻接矩阵图 graph ，包括节点和边集E表示。

我们可以认为这些服务器之间本身不存在连接，而当每个任务到来后，厂商为这些服务器分配连接。

在服务器网络中，当 E[i] [j] = 1 时，表示节点 i 能够直接连接到另一个节点 j。

这也表示节点 i和节点 j属于同一个任务。例如：



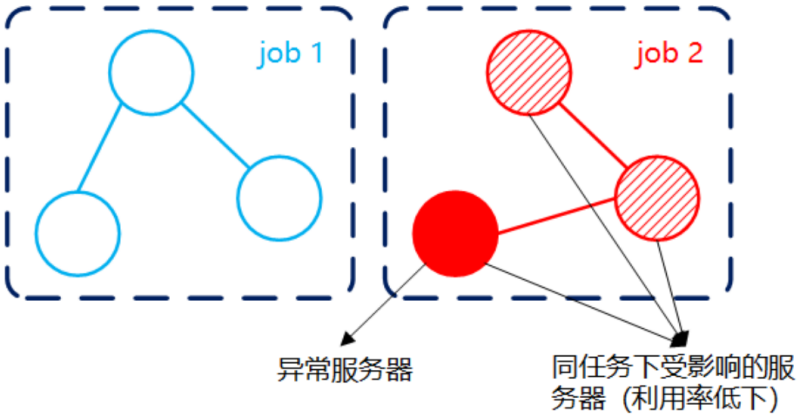
在一段时间的运行后，已知一些服务器 S 出现了异常的情况。

一旦出现服务器异常的情况，使用该服务器的机器学习任务立刻会被拖慢，

同属于这些机器学习任务的正常服务器都会受到影响（低利用率(under utilization)）。

我们用 M(S) 表示整个网络中受到服务器异常影响的最终节点数。

给出了一异常的样例：



维护这个存在服务器异常的集群，假设你具有k个(k<|S|)可供你替换的正常服务器，

请你写出程序找出**能够最小化** M(S)的替换节点编号。

如果有多个节点满足条件，就返回**索引最小**的节点。

输入输出

输入分多行，

第一行为总节点数V，和总连边数|E|。

第二行到第|E|+1行为连边，以i j e的形式指定E[i][j]。

第|E|+2行为异常服务器或异常连接的索引序列，例如0 1 2 3代表前四个服务器异常。

第|E|+3行为K。

例如：

第一小问

4 2

1 2 1

3 4 1

3

1

输出以一行打印选择的异常连接或服务器的索引

1. 代码逻辑和思路

（1）定义常量和函数声明

（2）深度优先搜索

dfs在图中从给定节点开始深度优先搜索，找到节点所在联通分量后

把联通分量的节点存储在component中

用visited数组记录已访问过的节点，防止重复访问

（3）查找联通分量

findComponents遍历图所有节点，用DFS查找每个节点所属联通分量

因为DFS可以有效寻找节点所在联通分量，所以可以将所有节点划分到不同的联通分量中

找到联通分量就将其存储在components中，记录联通分量大小到sizes数组中

（4）检查联通分量是否受异常节点影响：isAffected函数检查

包含异常返回tree，否则返回false

（5）最小化影响的节点替换minimizeImpact

先找到所有包含异常节点的联通分量，在把这些分量中所有节点作为候选替换节点

后按照贪心算法选影响最大的k个节点替换，优先选联通分量最大的节点

贪心算法的策略是遍历所有候选节点，优先选择联通分量最大的节点替换，从而最小化收到影响的节点数

（6）主函数

从输入读图的节点和边数

初始化图的邻接矩阵

根据输入边的信息更新邻接矩阵

读取异常节点信息

读取可替换的节点数k

调用minimizeImpact函数计算需要替换的节点

输出需要替换的节点

算法复杂度分析

1. **DFS (Depth-First Search)**:找到连通分量，并计算该连通分量的最小带宽和节点数。由于DFS遍历每个节点和每条边一次，所以对于一个图来说，时间复杂度是O(V + E)，其中V是节点数，E是边数。
2. **calculateImpact 函数**:函数中包含一次DFS调用，所以其复杂度是O(V + E)。还包含一些基本的常数时间操作（如带宽恢复等），但不会显著影响总体复杂度。
3. **minimizeNetworkImpact 函数**:主要步骤是对每个异常边调用一次 `calculateImpact` 函数，所以总共有 `nSize` 次 `calculateImpact` 调用。调用 `calculateImpact` 的总时间复杂度是O(nSize · (V + E))。后有一个排序过程，该部分的复杂度是O(nSize log nSize)。

综合这两部分的复杂度，最终算法的总体时间复杂度为：

O(nSize·(V+E)+nSizelognSize)

通常情况下，nSize 会远小于 V 和 E，所以如果 V 和 E 很大时，nSize 对复杂度的贡献相对较小。最终复杂度主要受 nSize·(V+E)n影响。因此，这个算法的时间复杂度为：

O(nSize·(V+E))

其他测试数据

测试数据1：简单情况

输入：

4 2

1 2 1

3 4 1

1 2

1

预期输出：

Nodes to replace: 1

测试数据2：所有节点连通

输入：

5 4

1 2 1

2 3 1

3 4 1

4 5 1

1 2

2

预期输出：

Nodes to replace: 1 2

网络设备的更换（选做）

网络连接的异常或故障也会影响多服务器任务的执行速度。

一个任务的执行速度仅与其涉及的服务器中**最慢的网络连接**相关（即部分设备异常不会影响整体执行）。【关键路径】

如一个任务中服务器最低连接带宽为50，

则若该任务中一条带宽为100的链路异常，带宽降低为70，对该任务执行没有影响。

用前一题中的参数，给出一个由 n 个节点组成的服务器网络，用 n × n 个邻接矩阵图 graph 表示。

我们可以认为这些服务器之间本身不存在连接，而当每个任务到来后，厂商为这些服务器分配连接。

记所有的连边为E，E[i][j] = 50 时，表示节点 i 能够直接连接到另一个节点 j，且两个节点之间的网络带宽为50。

在一段时间的运行后，已知一些网络设备 N 出现了异常的情况，

且对于每一个网络连接其异常后的带宽为e = D(e)。

为了评估这种网络影响带来的损失，我们用 L(N) 表示整个网络中受到网络连接异常的影响，其计算方法为：

$$L(N) = \sum^{jobs} 任务降低的带宽 * 该任务的节点数$$

维护这个存在网络连接异常的集群，假设你具有k个(k<|N|)可供你替换的正常网络设备，请你写出程序找出**能够最小化** L(S)的替换节点编号。如果有多个连边满足条件，就返回在N中**索引最小**的连边。

第二小问

4 2

1 2 1

3 4 1

0 //指连边中的第一个即1节点和2节点的连边异常

1

1 -(1)- 2

3 -(1)- 4

为了处理网络设备故障的情况，我们需要考虑网络带宽的降低对任务的影响，并找出替换这些设备的最优方案。

这里假设每个任务的执行速度仅与其涉及的服务器中最慢的网络连接相关。

代码思路：

（1）读取节点数、边数、异常边的信息，并构建邻接矩阵。

（2）找到连通分量，并通过DFS计算最小带宽和节点数。

（3）对每个异常边，计算其对整个网络的影响。

（4）计算完所有异常边的影响后，选择影响最小的 k个异常边进行替换。

代码逻辑：

(1) `dfs` 函数使用深度优先搜索（DFS）遍历图，找到连通分量，并计算该连通分量的最小带宽和节点数。

a.标记当前节点已访问

b.用计数器记录当前连通分量中的节点数

c.遍历节点

d.如果节点i未访问，且与当前节点相连

e.更新最小带宽

f.用dfs递归访问相邻节点

（2） `calculateImpact` 函数计算单个异常边对网络的总影响。

a.保存原始带宽

b.暂时应用异常带宽

c.计算连通分量的最小带宽和节点数

d.恢复原始带宽

e.如果最小带宽是原始带宽，那么异常后的最小带宽将是faultyBandwidth

f.带宽减少量

g.计算影响：带宽减少量 * 节点数

h.sum统计影响

`minimizeNetworkImpact` 函数找到能够最小化网络影响的异常边替换策略。

a.计算每个异常边的影响

b.保存原始索引

c.按影响排序并选择最小的k个

d.将选择的k个最小影响的边索引存入result数组

`main` 函数读取输入数据，初始化图的邻接矩阵，调用核心函数计算最小化影响的替换边，并输出结果。

a.初始化无向图的邻接矩阵

b.读取边的连接信息，初始化带宽为50

c.读取异常边的信息

d.起始和结束节点索引，最后计算异常后的带宽

算法复杂度分析

1. **DFS (Depth-First Search)**:找到连通分量，并计算该连通分量的最小带宽和节点数。由于DFS遍历每个节点和每条边一次，所以对于一个图来说，时间复杂度是O(V + E)，其中V是节点数，E是边数。
2. **calculateImpact 函数**:函数中包含一次DFS调用，所以其复杂度是O(V + E)。还包含一些基本的常数时间操作（如带宽恢复等），但不会显著影响总体复杂度。

3. **minimizeNetworkImpact 函数**:主要步骤是对每个异常边调用一次 `calculateImpact` 函数，所以总共有 `nSize` 次 `calculateImpact` 调用。调用 `calculateImpact` 的总时间复杂度是O(nSize · (V + E))。后有一个排序过程，该部分的复杂度是O(nSize log nSize)。

综合这两部分的复杂度，最终算法的总体时间复杂度为：

O(nSize·(V+E)+nSizelognSize)

通常情况下，nSize 会远小于 V 和 E，所以如果 V 和 E 很大时，nSize 对复杂度的贡献相对较小。最终复杂度主要受 nSize·(V+E)n影响。因此，这个算法的时间复杂度为：

O(nSize·(V+E))

设备替换和成本分析（选做）

假设运维部门总共的资金为P，购买服务器和网络设备的成本分别为R(s)和R(e)。在服务集群出同时出现服务器异常和网络异常的情况下，请你合理运用设备资金，替换异常设备来使得如下影响最小：（alpha和beta都是已知因数）

第三小问可分两行