

16

Connectivity

连通性

描述图中节点的可达性



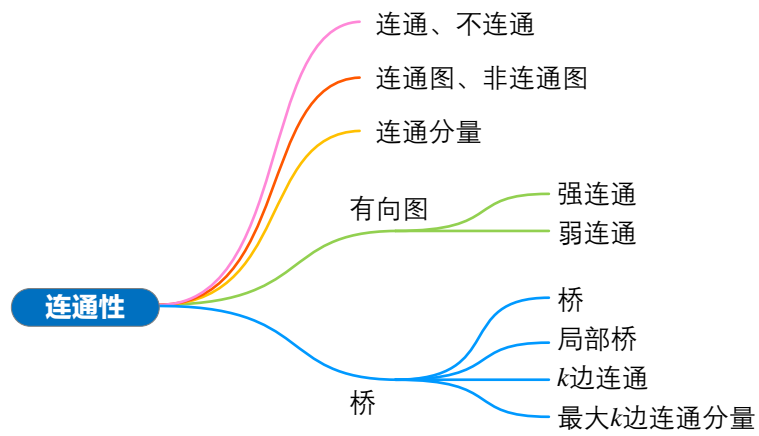
柏拉图我至亲，而真理价更高。

Plato is dear to me, but dearer still is truth.

—— 亚里士多德 (Aristotle) | 古希腊哲学家 | 384 ~ 322 BC



- networkx.bridges() 生成图中所有桥的迭代器
- networkx.complete_graph() 生成一个完全图，图中的每对节点之间都有一条边
- networkx.connected_components() 生成图中所有连通分量的节点集
- networkx.gnp_random_graph() 生成一个具有 n 个节点和边概率 p 的随机图
- networkx.has_bridges() 检查图中是否存在桥
- networkx.has_path() 检查图中是否存在从一个节点到另一个节点的路径
- networkx.is_connected() 判断一个图是否连通
- networkx.is_k_edge_connected() 判断图是否是 k 边连通的
- networkx.is_strongly_connected() 判断有向图是否强连通
- networkx.is_weakly_connected() 判断有向图是否弱连通
- networkx.k_edge_components() 识别图中的最大 k 边连通分量
- networkx.local_bridges() 生成图中所有局部桥的迭代器
- networkx.number_connected_components() 返回图中连通分量的数量
- networkx.shortest_path() 寻找两个节点之间的最短路径



16.1 连通性

连通性 (connectivity) 在图论中是一个重要概念，它描述了图中节点之间的连接关系。本节介绍有关连通性的常用概念。

连通、不连通

在一个无向图 G 中，如果图 G 包含从节点 u 到节点 v 的路径，那么节点 u 和 v 被称为**连通** (connected)；否则，它们被称为**不连通** (disconnected)。节点与节点相连也叫可达 (reachable)，节点与节点不相连也叫不可达 (unreachable)。

除了节点与节点之间的连通，我们还会提到图的连通性。图的连通性考虑的是一幅图中任意一对节点是否可达。一个**连通图** (connected graph) 是指图中的每一对节点都是可达的。图的连通性考虑的是图的整体结构，强调图中没有被孤立的部分，即“孤岛”。这是本章后续要介绍的内容。

如图 8 所示，节点 a 和节点 i 显然相连。通过图中黄色高亮的路径，我们可以从节点 a 走到节点 i ，或者从节点 i 走到节点 a 。这条路径可以记作边的序列 $w = (ac, cg, gi)$ ，也可以记作点的序列 (a, c, g) 。

如果图中两个节点还通过长度为 1 的路径相连，那么这两个节点为**邻接** (adjacent)。图 8 中，和节点 a 邻接的节点有 b 、 c 。也就是说，节点 a 可以走一步便到达节点 b 或节点 c 。

此外，节点 a 和节点 e 、 k 、 j 都不相连；也就是说，没有路径可以到达。

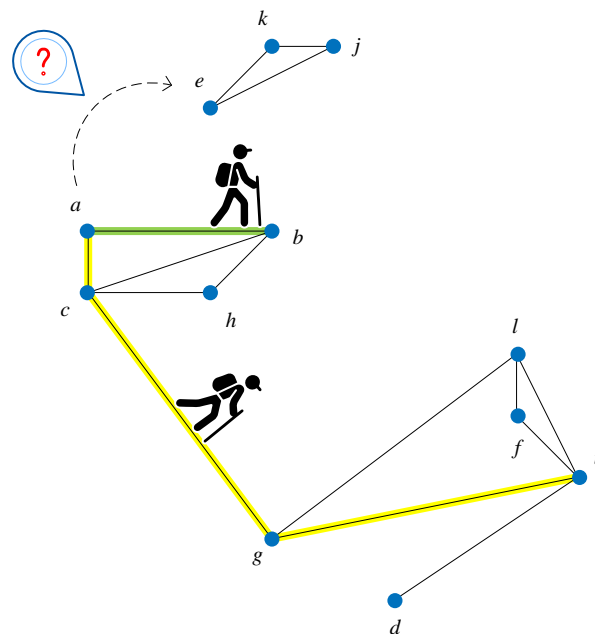


图 1. 节点之间的连接

上一章介绍过，在无向图中，不考虑权重时，最短路径是指连接图中两个节点的路径中，具有最小边数的路径。路径的长度通常用边的数量来度量。最短路径可能有多条，但它们的长度相同，都是连接两个节点的最短路径。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

注意，如果图中边有权重，则计算最短路径时，要计算路径的权重之和。而且有向图中，计算最短路径时还要考虑方向。

图 2 (c) 给出的路径虽然绕了很多远路，但是我们发现这条路径穿越了所有节点。

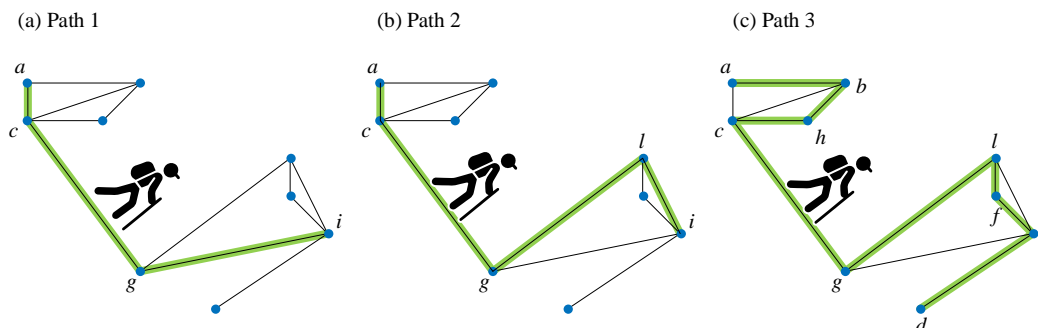


图 2. 最短路径

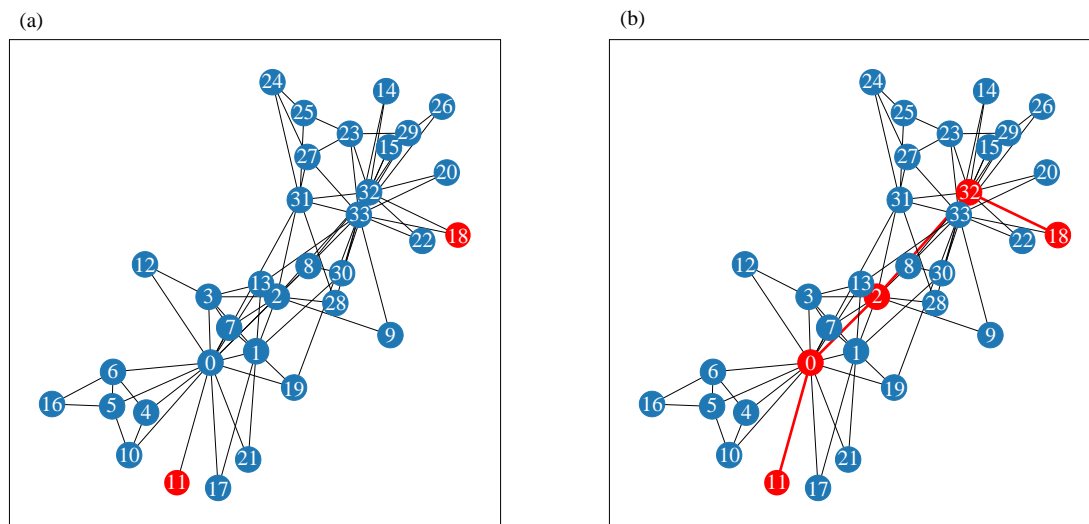


图 3. 空手道俱乐部会员关系图，节点 11、节点 18 的连通性

但是，当我们将节点 11、0 之间的边删除之后，节点 11 和节点 18 将不再连通。

代码 1 绘制图 3，下面聊聊其中关键语句。

- a** `networkx.karate_club_graph()` 导入空手道俱乐部会员关系图。
- b** 用 `networkx.has_path()` 检查在图 `G` 中是否存在从节点 11 到节点 18 的路径。
- c** 用 `networkx.shortest_path()` 找出在图 `G` 中从节点 11 到节点 18 的最短路径。这里的“最短”通常指的是路径上边的数量最少，但如果图中的边有权重，它也可以指的是路径的总权重最低。函数将返回一个列表，列表中包含了从起点到终点的最短路径上的所有节点，包括起始节点和目标节点。

d 为自定义函数，接受一个节点列表 `node_list` 作为输入，并通过列表生成式生成一个新列表 `list_edges`，其中包含相邻节点对形成的边。然后，函数返回边的列表，实现了将一系列节点转换为它们之间相连的边的序列。**e** 调用自定义函数。

f 用 `remove_edge()` 方法删除一条边。

```
import networkx as nx
import matplotlib.pyplot as plt

a G = nx.karate_club_graph()
# 空手道俱乐部图
pos = nx.spring_layout(G, seed=2)

plt.figure(figsize = (6,6))

nx.draw_networkx(G, pos)
nx.draw_networkx_nodes(G, pos,
                        nodelist = [11,18],
                        node_color = 'r')
plt.savefig('空手道俱乐部图.svg')

b nx.has_path(G, 11, 18)
# 检查两个节点是否连通

c path_nodes = nx.shortest_path(G, 11, 18)
# 最短路径

# 自定义函数将节点序列转化为边序列
d def nodes_2_edges(node_list):
    # 使用列表生成式创建边的列表
    list_edges = [(node_list[i], node_list[i+1])
                  for i in range(len(node_list)-1)]

    return list_edges


e path_edges = nodes_2_edges(path_nodes)
# 将节点序列转化为边序列

plt.figure(figsize = (6,6))

nx.draw_networkx(G, pos)
nx.draw_networkx_nodes(G, pos,
                        nodelist = path_nodes,
                        node_color = 'r')
nx.draw_networkx_edges(G, pos,
                        edgelist = path_edges,
                        edge_color = 'r')
plt.savefig('空手道俱乐部图, 节点11、18最短路径.svg')

# 删除一条边
f G.remove_edge(11,0)

nx.has_path(G, 11, 8)
# 再次检查两个节点是否连通
```

代码 1. 检查节点之间的连通性 |  Bk6_Ch16_01.ipynb

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

连通图、非连通图

如果一个图叫**连通图** (connected graph)，则图中的任意一对节点 u 和 v ，都存在一条 u 到 v 的路径。换句话说，从图中的任意一个节点出发，都可以到达图中的任意其他节点；即图中的任意两个节点之间都是可达的。如果一个图是连通图，那么它只有一个分量。图 5 给出了几个连通图的例子。

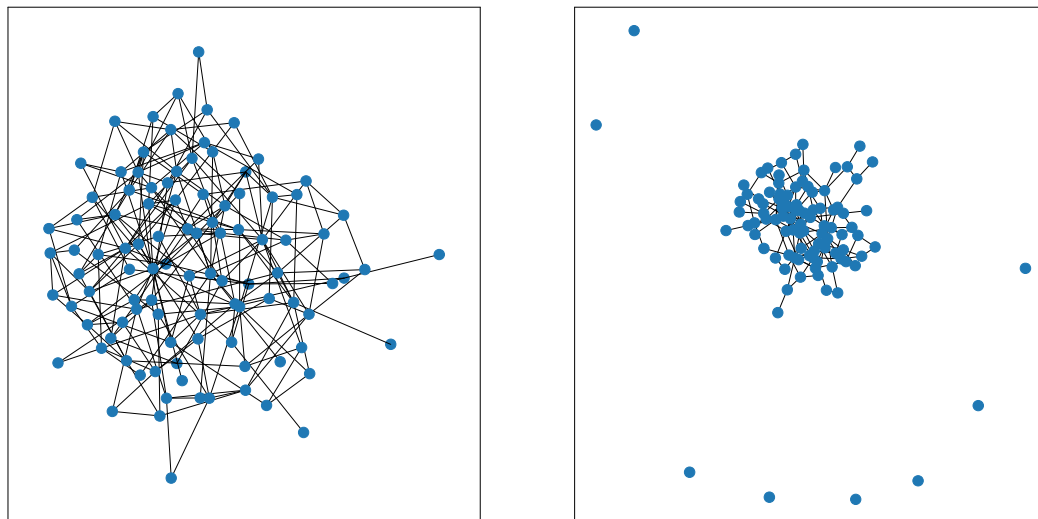


图 4. 比较连通图、非连通图

相反，一个图被称为**非连通图** (disconnected graph)，如果存在至少一对节点，它们之间没有路径相连。一个非连通图可能由多个分量组成，每个分量本身都是一个连通图，但分量之间没有直接的路径。图 6 给出了几个非连通图的例子。

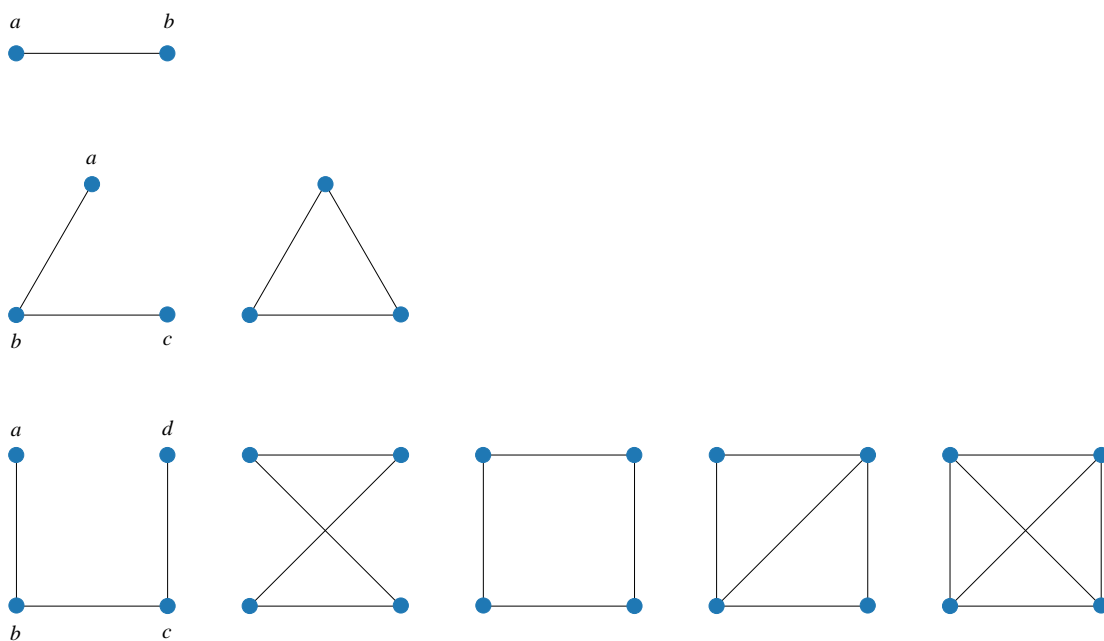


图 5. 连通图

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

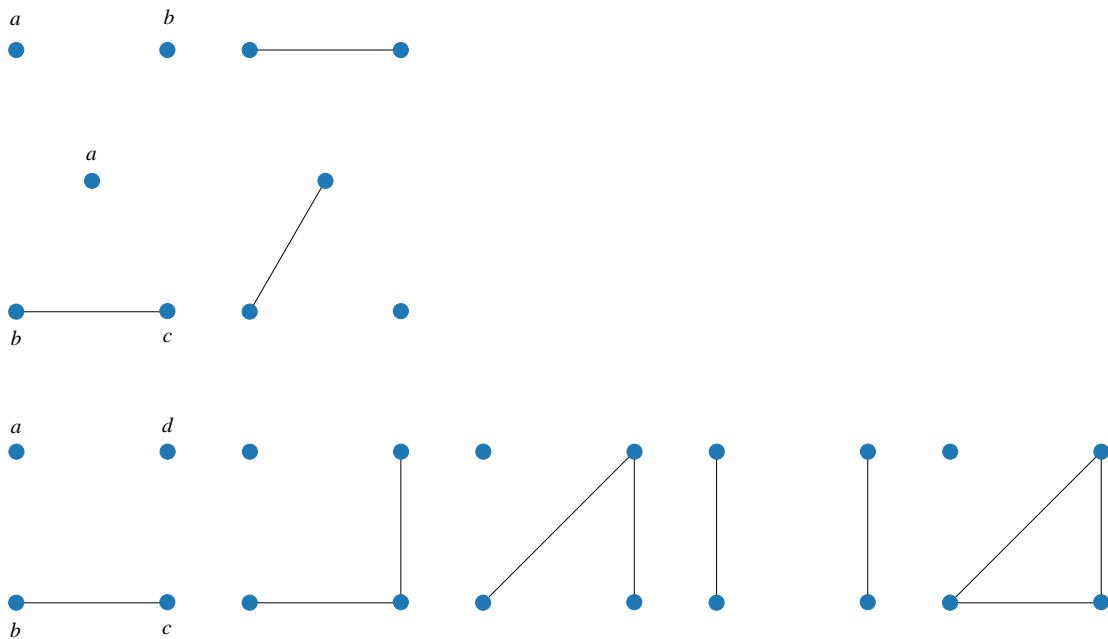


图 6. 非连通图

图 7 可视化具有最多 6 个节点的所有连通图。这个例子来自 NetworkX，请大家自行学习。

https://networkx.org/documentation/stable/auto_examples/graphviz_layout/plo_t_atlas.html

连通图是一个基本的图论概念，它强调了图中节点之间的连接性。在一些应用中，特别是网络和通信领域，连通图的概念非常重要，因为它表示着信息或者流量可以在图中的任意两个节点之间自由传递。

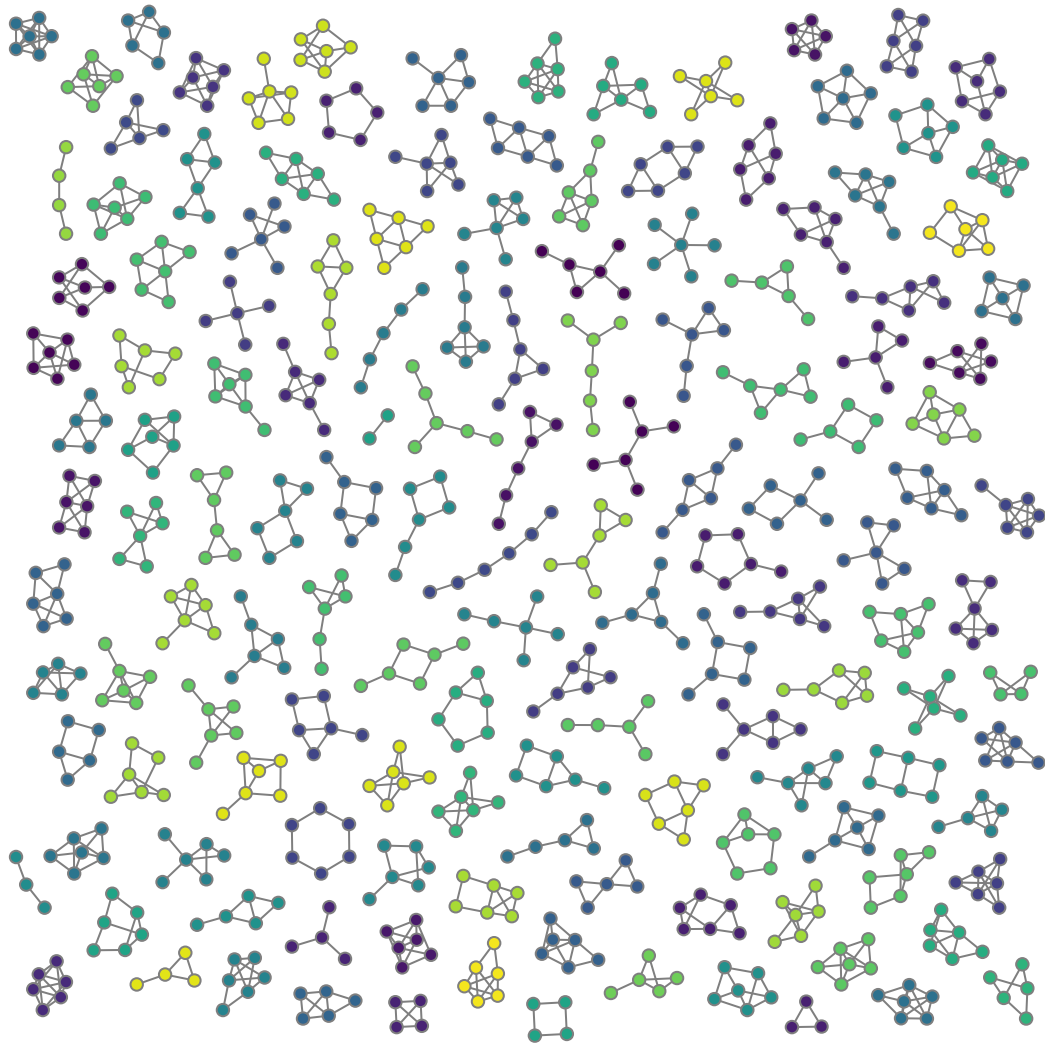


图 7. 最多 6 个节点的所有连通图

16.2 连通分量

进一步观察，我们可以发现节点 e 、 k 、 j 三个节点之间都相互连接；图 8 中剩余其他 9 个节点也都相互连接，哪怕有些节点之间路径可能稍远。这种情况下，这幅图包含了两个**连通分量** (connected components)，简称**分量** (components)，具体如图 8 所示。有些参考书把连通分量叫做连通组件。

简单来说，无向图中，连通分量是一个无向子图，在分量中的任何两个节点都可以经由该图上的边相互抵达；但是，一个分量没有任何一边可以连到其他分量中的任何节点。图 8 的图可以看成由两个分量组成，节点 e 、 k 、 j 构成的分量像是一个孤岛。

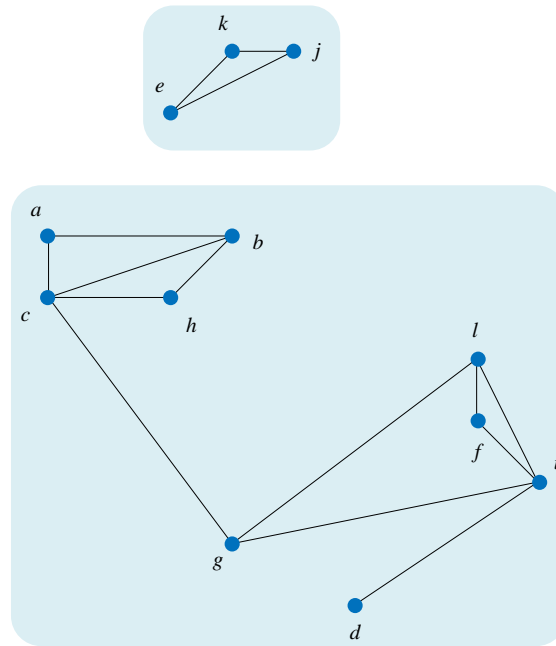


图 8. 图中有两个分量

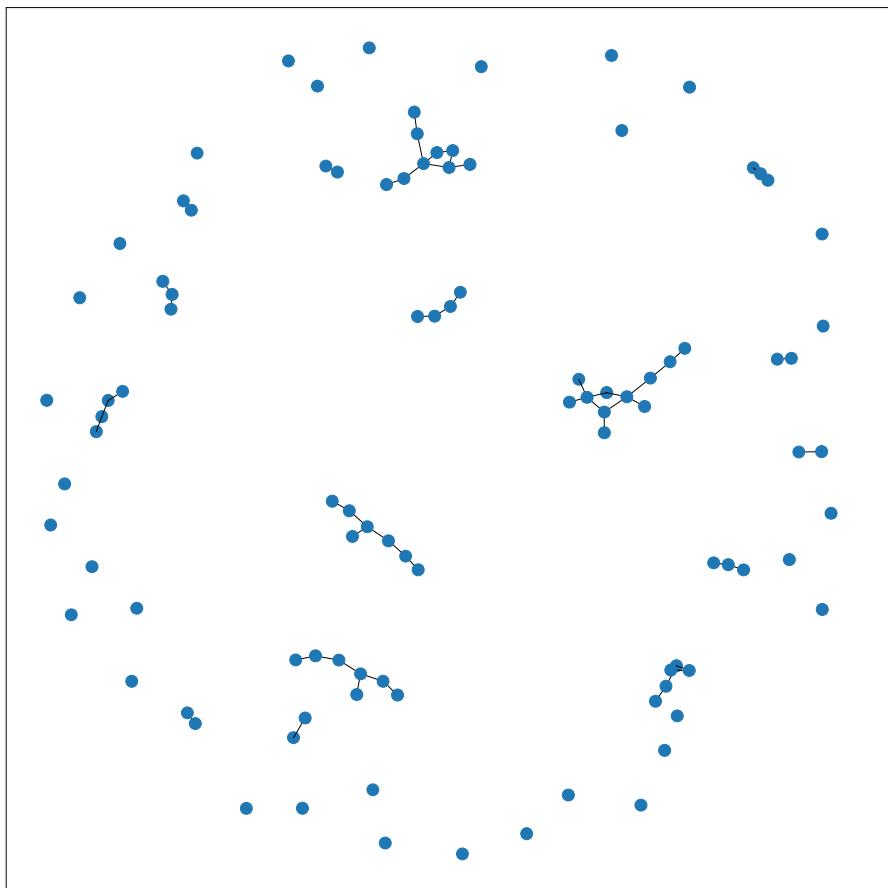


图 9. 含有若干连通分量的无向图

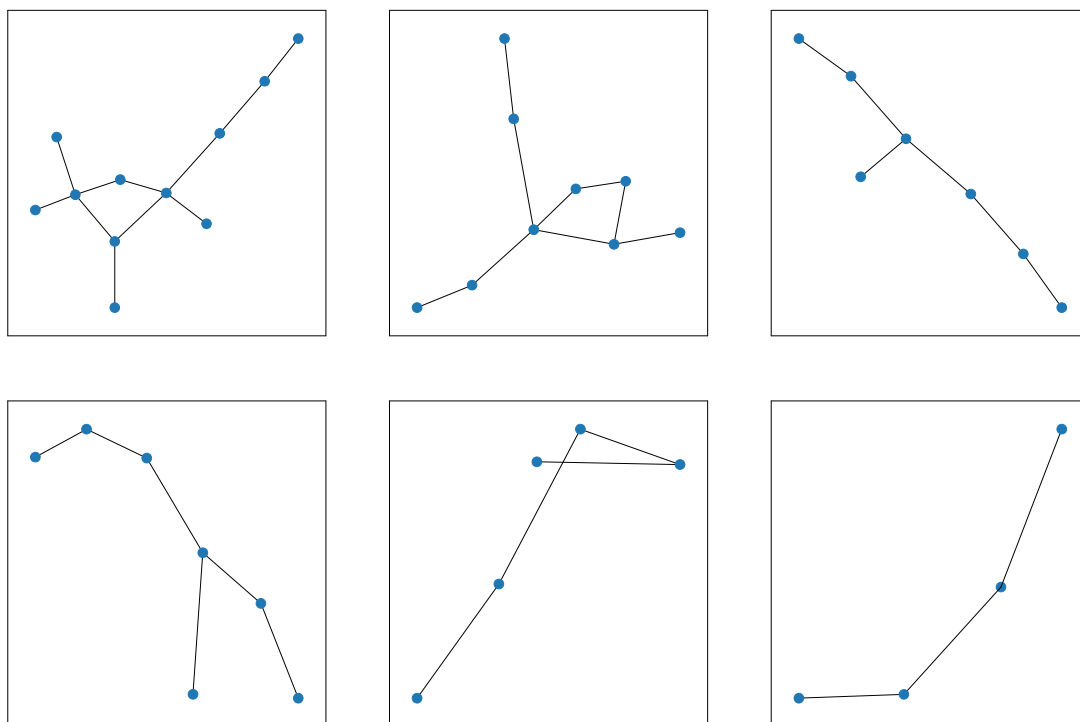


图 10. 节点数量上排名前 6 大连通分量

代码 2 绘制图 9、图 10，下面其中关键语句。

- a 用 `networkx.is_connected(G)` 检查无向图是否连通。回顾一下，在图论中，一个无向图被认为是连通的，如果图中每一对节点之间都存在至少一条路径相连，即从图中的任何一个节点都可以到达任何其他节点。
- b 用 `networkx.number_connected_components(G)` 找出无向图 `G` 连通分量的数量。
- c 用 `networkx.connected_components(G)` 找出无向图 `G` 中的所有连通分量。
- d 用 `sorted()` 按照每个连通分量的大小（即其中包含的节点数量）进行排序。参数 `key=len` 指定排序的依据，这里是每个连通分量的长度，即其中节点的数量。参数 `reverse=True` 指定排序应该是降序的。也就是说，最大的连通分量会排在列表的最前面。
- e 用 `subgraph()` 方法用于根据给定的节点集合创建原图的一个子图。
- f 从 `pos`（存储图中每个节点的位置信息）取出连通分量子图节点位置。
- g 用 `networkx.draw_networkx()` 绘制连通分量子图。

```

import networkx as nx
import numpy as np
import matplotlib.pyplot as plt

G = nx.gnp_random_graph(100, 0.01, seed=8)
# 创建随机图

plt.figure(figsize = (6,6))
pos = nx.spring_layout(G, seed = 8)
nx.draw_networkx(G, pos,
                  with_labels = False,
                  node_size=20)
plt.savefig('全图.svg')

# 检查无向图是否连通
a nx.is_connected(G)

# 连通分量的数量
b nx.number_connected_components(G)

# 连通分量
c list_cc = nx.connected_components(G)

# 根据节点数从大到小排列连通分量
d list_cc = sorted(list_cc, key=len, reverse=True)

# 可视化前6大连通分量
fig, axes = plt.subplots(2,3,figsize = (9,6))
axes = axes.flatten()

e for idx in range(6):
    Gcc_idx = G.subgraph(list_cc[idx])

f     pos_Gcc_idx = {k: pos[k] for k in list(Gcc_idx.nodes())}

# 可视化连通分量

g     nx.draw_networkx(Gcc_idx,
                       pos_Gcc_idx,
                       ax = axes[idx],
                       with_labels = False,
                       node_size=20)

plt.savefig('前6大连通分量.svg')

```

代码 2. 无向图中的连通分量 | Bk6_Ch15_04.ipynb

16.3 强连通、弱连通：有向图

在图论中，强连通 (strongly connected) 和弱连通 (weakly connected) 是用来描述有向图连通性质的两个不同概念。这些概念帮助我们理解和分析有向图中节点之间的可达性。

一个有向图被认为是强连通的，如果图中的每一对节点都是互相可达的。换句话说，对于图中的任意两个节点 u 和 v ，都存在从 u 到 v 的有向路径，同时也存在从 v 到 u 的有向路径。如果一个有向图的整个图是强连通的，则称这个图为强连通图。

在一个不完全强连通的有向图中，可以通过找出最大的强连通子图来识别强连通分量。每个强连通分量都是图中的一个最大子图，在这个子图中，任意两点都是相互可达的。

与强连通相对，一个有向图被认为是弱连通的，如果忽略掉边的方向之后，图中的任意两个节点都是连通的。也就是说，如果将有向图中的所有有向边替换为无向边，那么这个无向图应该是连通的。弱连通更容易满足，因为它不要求节点间的双向可达性。

在有向图中，弱连通分量是图的一个最大子图，其中的节点即使忽略边的方向，也是相互连通的。

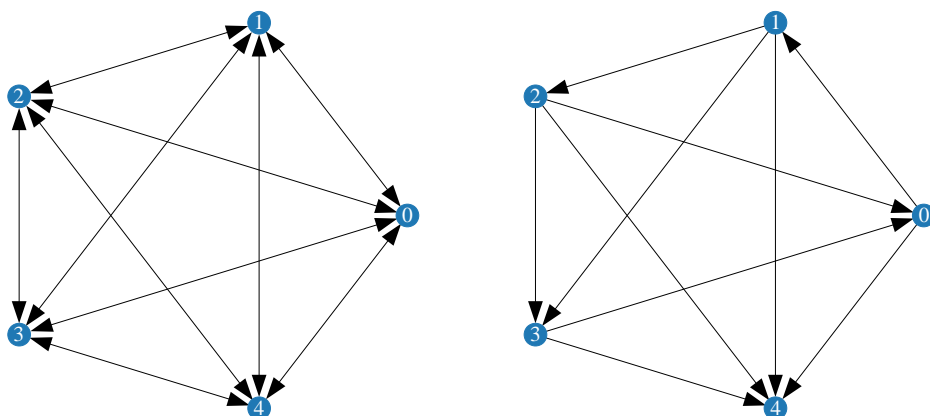


图 11. 强连通，弱连通

16.4 桥

桥

在图论中，**桥** (bridge) 是指连接图中两个不同连通分量的边。移除一个桥可能导致整个图分裂成两个或更多个不再连通的部分。因此，桥在图的连通性中具有特殊的作用。

具体来说，一条边是桥的条件是：如果将这条边移除，图的连通性会减弱，也就是说，原本通过这条边连接的两个连通分量会变得不再连通。

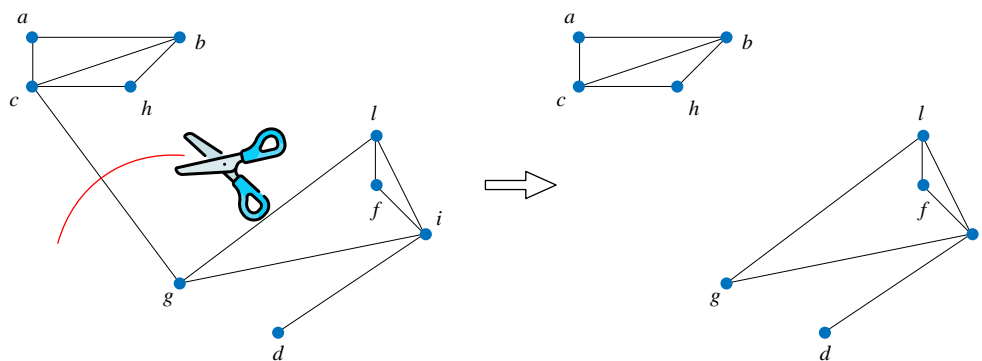


图 12. 拆桥

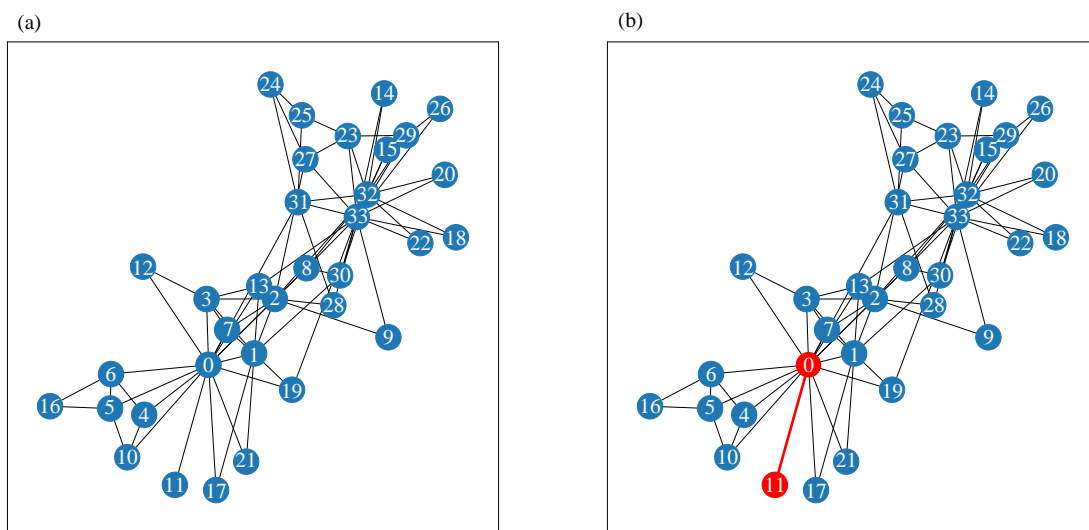


图 13. 空手道俱乐部会员关系图中的桥

局部桥

局部桥 (local bridge) 是指连接两个具有很高相似度的节点的边。具体来说，如果两个节点之间只有一条边，而这两个节点有许多共同的邻居，那么这条边就被称为局部桥。局部桥反映了节点之间在局部邻域内的相对独立性。局部桥主要关注节点之间的相似性，而不是整个图的连通性。

所有的桥都是局部桥。如果一条边是桥，那么它连接的两个节点之间没有其他的替代路径，即这两个节点只能通过这条边相互连接。因此，从局部的角度看，这条边就是连接了两个相对独立的节点，可以被称为局部桥。

但是局部桥未必是桥。局部桥是指连接两个相似性较高的节点的边，它们可能有许多共同的邻居。这种连接方式强调的是节点之间的相似性。然而，即使存在许多共同的邻居，这条边也可能不是图的桥，因为可能存在其他路径连接这两个节点。

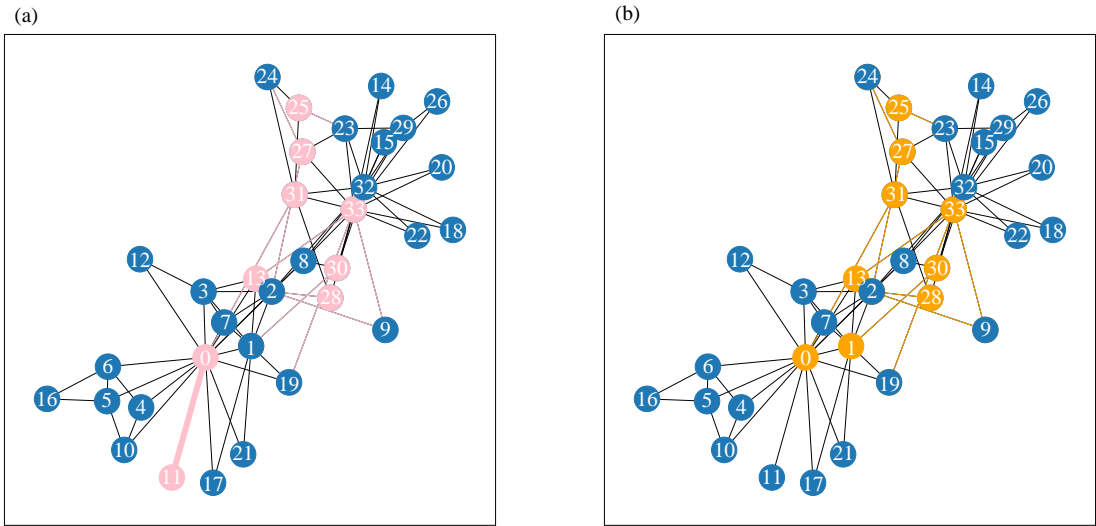


图 14. 空手道俱乐部会员关系图中的局部桥，局部桥中的非桥

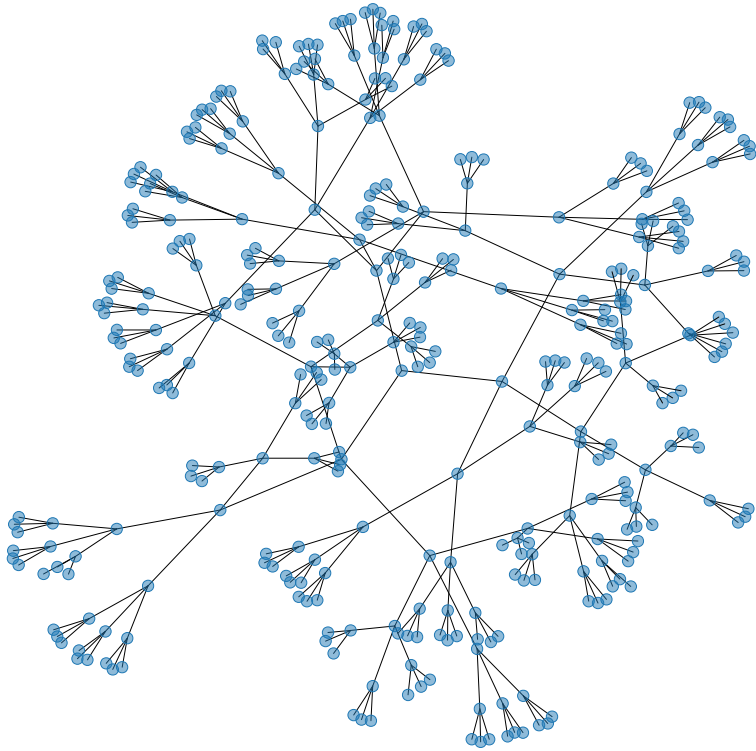


图 15. 树中每条边都是桥，也都是局部桥

k 边连通

在图论中， k 边连通是用来衡量无向图连通性强度的一个概念。一个无向图被认为是 k 边连通的，如果它至少需要移除 k 条边才能变成非连通图。换句话说，即使图中任意少于 k 条边被移除，图仍然能保持连通。这个属性是图的连通性和鲁棒性的一个重要指标，特别是在网络设计和网络分析领域。

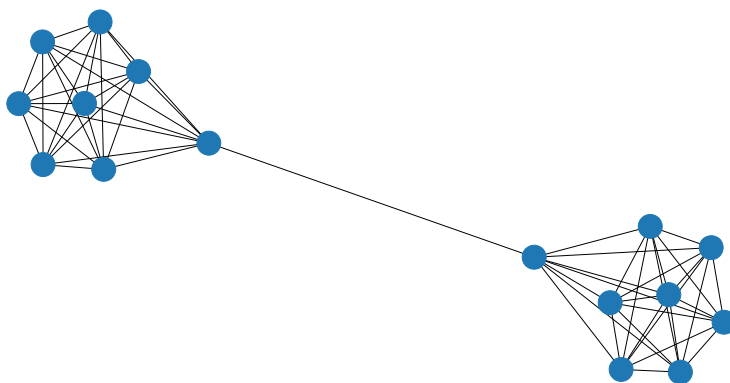


图 16. 哑铃图, 1 边连通

最大 k 边连通分量

最大 k 边连通分量 (maximal k -edge-connected component) 是图中的一个子图，其中任何两个节点至少可以通过 k 条边互相独立的路径相连。这意味着在这个组件内部，即使删除了最多 $k - 1$ 条边，任意两个节点之间仍然至少存在一条路径，使它们保持连通。

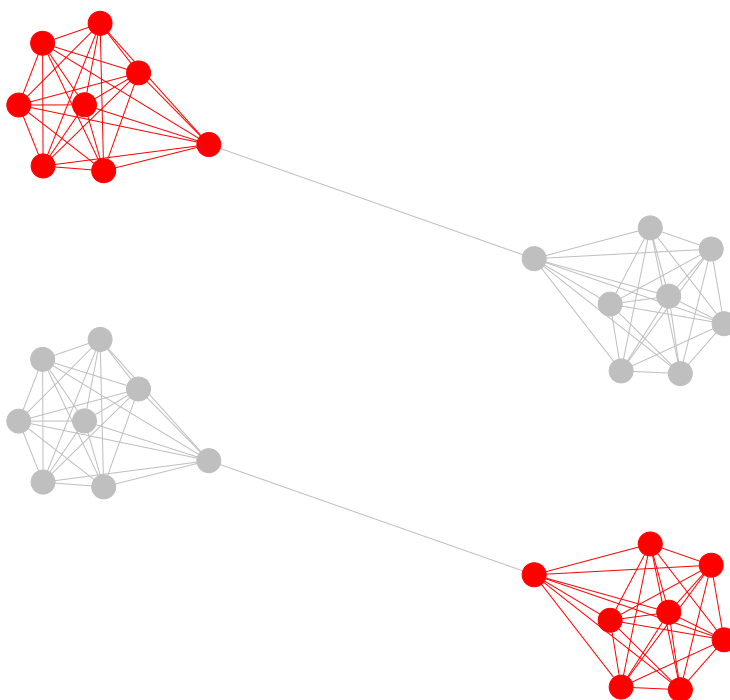
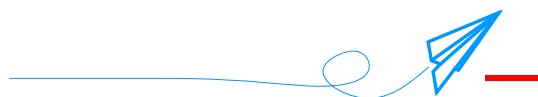


图 17. 哑铃图, 两个最大 2 边连通分量



连通性是图论中描述图的节点如何通过边相连的概念。连通图中任意两点间都有路径相连；非连通图存在至少一对节点间无路径相连。连通分量是无向图中最大的连通子图。有向图的强连通意味着任意

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

两点间存在双向路径，弱连通则至少通过将所有边视为无向边可实现连通。桥是图中移除后会增加连通分量数量的边，局部桥是除非两端直接相连外，不属于任何其他更小的循环的边。