管理定量分析与软件应用
Chapter2 社会网络的基础知识

潘炤，副教授
华中科技大学管理学院
victola.pz@gmail.com

Chapter2 社会网络的基本概念

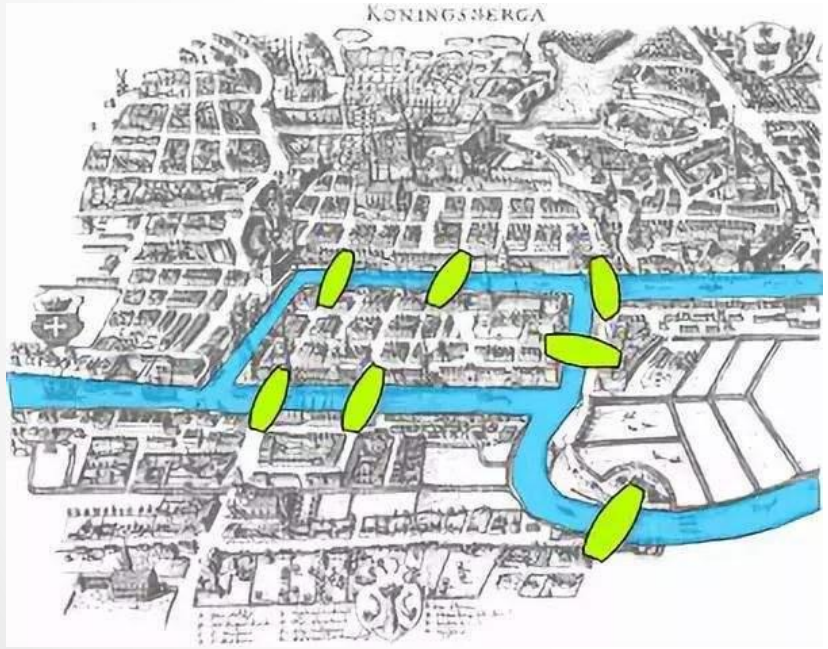
- 社会网络的发展历程
- 社会网络的基本概念
- 社会网络的存储与表示
- 社会网络的可视化

图论的起源与发展历程

- 图论是研究点与线组成的“图形”问题的一门科学，是组合数学的一个分支，也是近几十年来最活跃的数学分支之一。
  - 萌芽阶段（18世纪中叶到19世纪中叶）
  - 发展阶段（19世纪中叶到20世纪中叶）
  - 应用阶段（20世纪中叶以后）
- 存在一类问题：只与事物间联系有关，与大小、位置、距离无关（图论、拓扑学）。

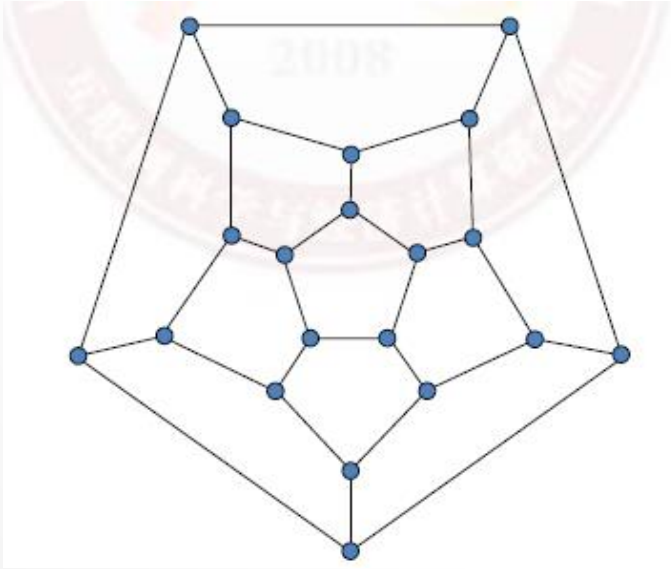
哥尼斯堡七桥问题（1736）

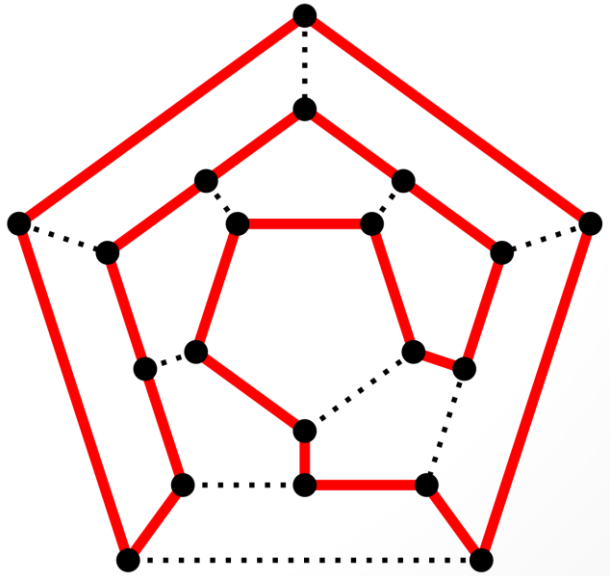
- Königsberg七桥位于前苏联的加里宁格勒，历史上曾是德国东普鲁士省的省会。普雷格尔河流经哥尼斯堡小城, 河中有两个小岛, 在四块陆地之间修建了七座小桥连接岛与河岸及岛与岛。



环游世界问题/Hamilton问题（1856）

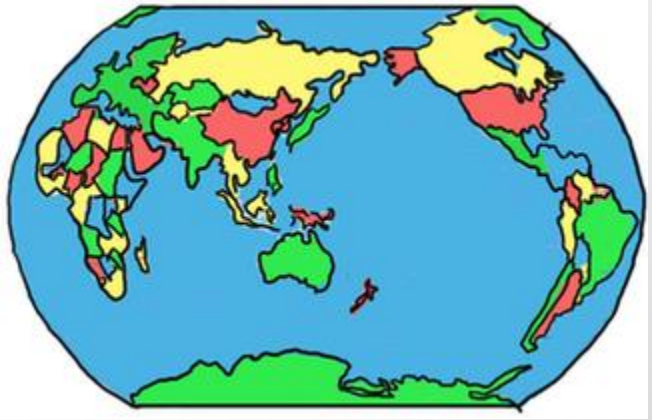
- 1856年，当时英国数学家Hamilton设计了一种名为周游世界的游戏。
  - 他在一个实心的正十二面体的十二个顶点上标以世界上著名的二十座城市的名字。
  - 要求游戏者沿十二面体的棱从一个城市出发，经过每座城市恰好一次，然后返回到出发点，即“绕行世界”。
- Hamilton最早对这类问题进行了研究，此类问题也称为Hamilton问题。





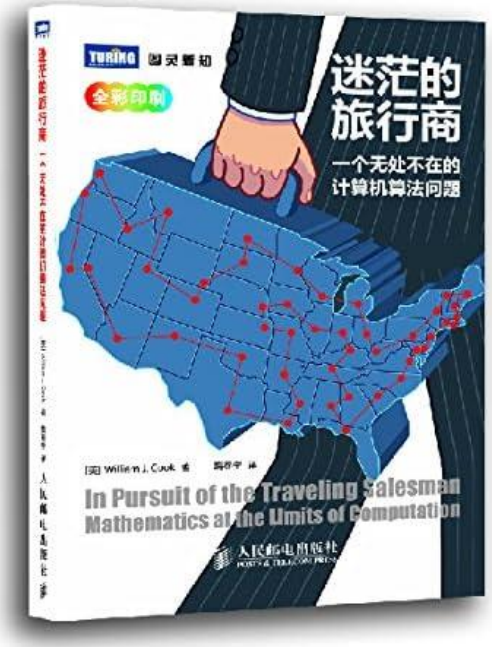
地图着色（四色猜想）（19世纪中）

- 为了能够迅速地区分一个平面地图或球面地图上的各个国家（假设这些国家在地图上都是连通的），需要用若干种颜色对这些国家着色，使得具有公共边界的两个国家涂染不同的颜色。要保证每张地图都能如此着色，最少需要多少种颜色？这个问题是1850年被一名刚毕业的大学生Francis Guthrie首先提出的。
- 直到1976年，四色问题被美国Illinois大学的K.Appel和W.Haken用计算机证明是正确的。这个证明令数学界震惊，它用了1200多小时，作出100亿个独立的逻辑判断。
- 尽管有了这个机器证明，但它仍然是数学上未解决的问题之一 。



旅行售货员问题（1934）

- 给出城市之间的距离，要求一位推销员从某一城市出发，周游每个城市一次，然后回到出发的城市，并且选的路径最短。
- 这是一个图论优化问题，最早由美国数学家威特涅于1934年在普林斯顿一次讨论班上提出。
- 1954年几位美国数学家用线性方程的方法解决了49个城市的旅行售货员问题。

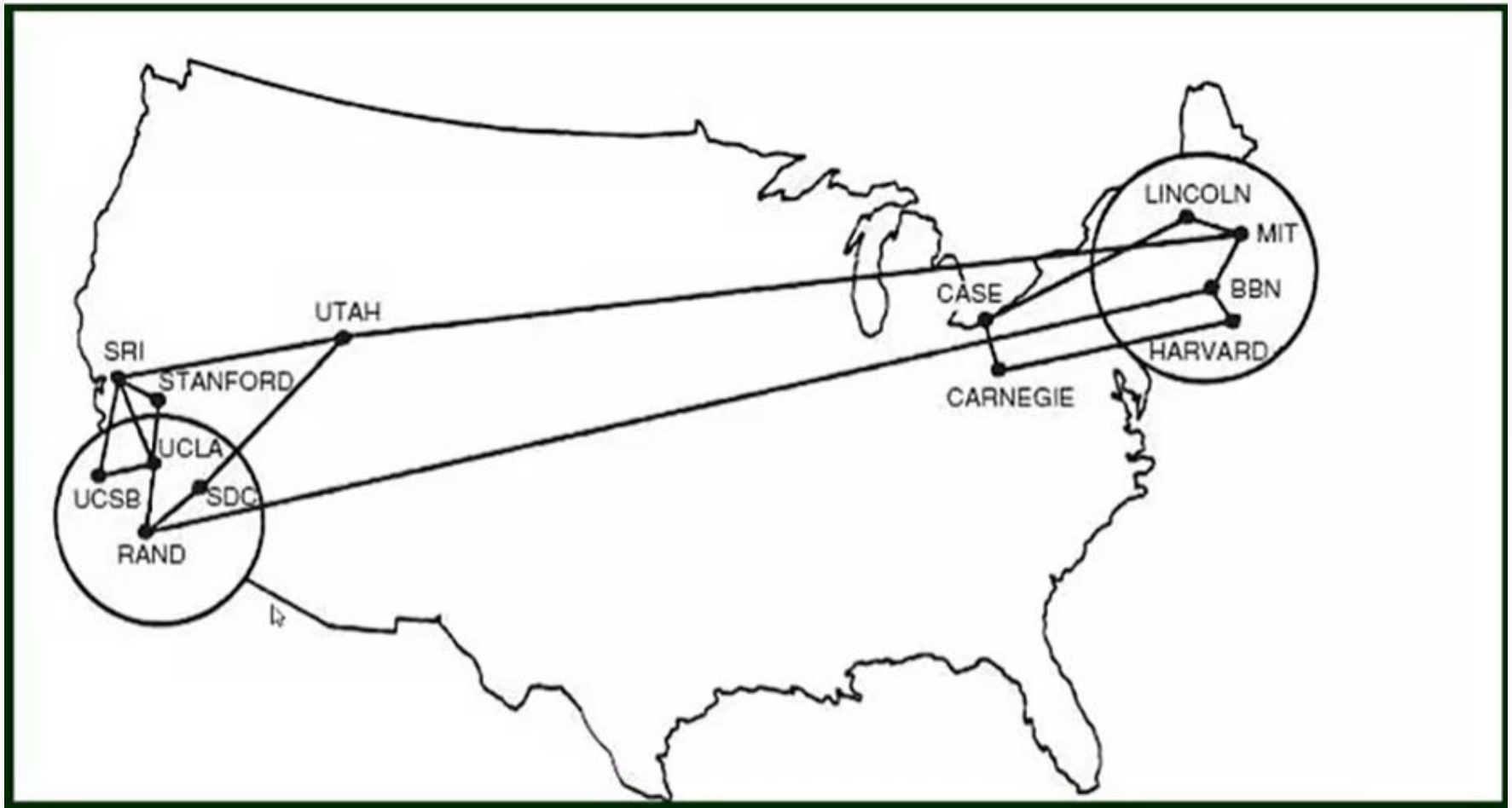


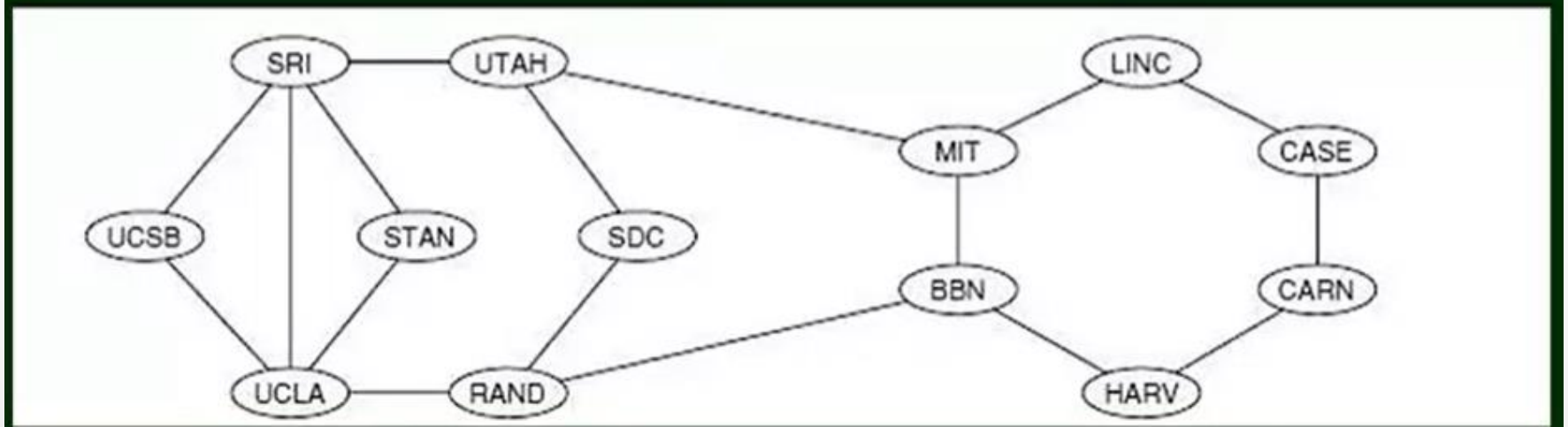
社会网络分析的发展历程

- 20世纪30年代，早期发展主要围绕三条主线：
  - 社会心理学计量学学派的图论研究；
  - 哈佛学派人际模式和团伙研究；
  - 曼彻斯特人类学派部落和“共同”体关系结构研究；
- 20世纪70年代后，社会网络分析成熟起来：
  - “新哈佛学派”出现
  - 怀特的结构分析
  - 格兰诺维特的强弱联系研究
- 20世纪90年代后，社会网络的新发展：
  - 伯特的结构洞理论
  - 林南的社会资本理论

Chapter2 社会网络的基本概念

- 社会网络计算的发展历程
- 社会网络的基本概念
- 社会网络的存储与表示
- 社会网络的可视化


LINCOLN
MIT
BBN
HARVARD
CASE
CARNEGIE
UTAH
SRI
STANFORD
UCLA
UCSB
SDC
RAND


SRI
UTAH
LINC
MIT
CASE
UCSB
STAN
SDC
BBN
CARN
UCLA
RAND
HARV

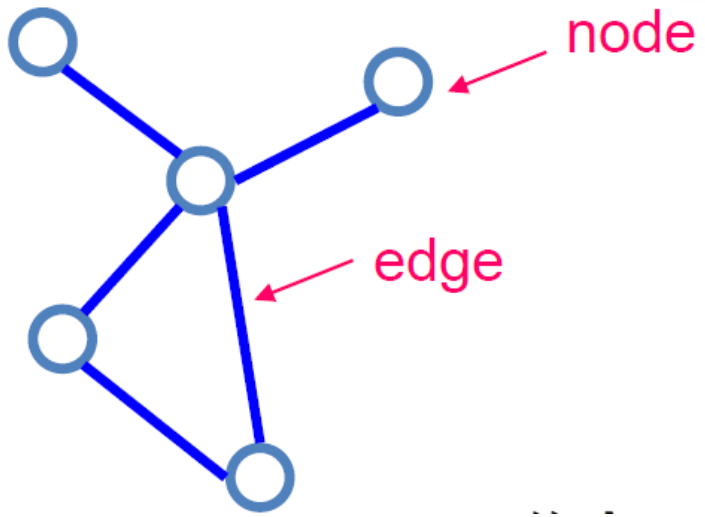
图（Graph）

- 图（graph）是一个三元有序组$\boldsymbol{G}=(\boldsymbol{G}(\boldsymbol{V}),\boldsymbol{G}(\boldsymbol{E}))$。其中，

- $G(V)$是一个非空的**顶点集合**，简称为点集，简记为V；
  - 其元素称为顶点或节点（vertex）。用$|V|$表示顶点数。
- $G(E)$是一个与V不交的**边集合**，由V中的点组成的无序对构成的集合，简称为边集，简记为E；
  - 其元素称为边（edge）。且同一点对在E中可以重复出现多次。用$|E|$表示边数。
- 一个图也可以简记为$G=(V,E)$。

网络（Network）

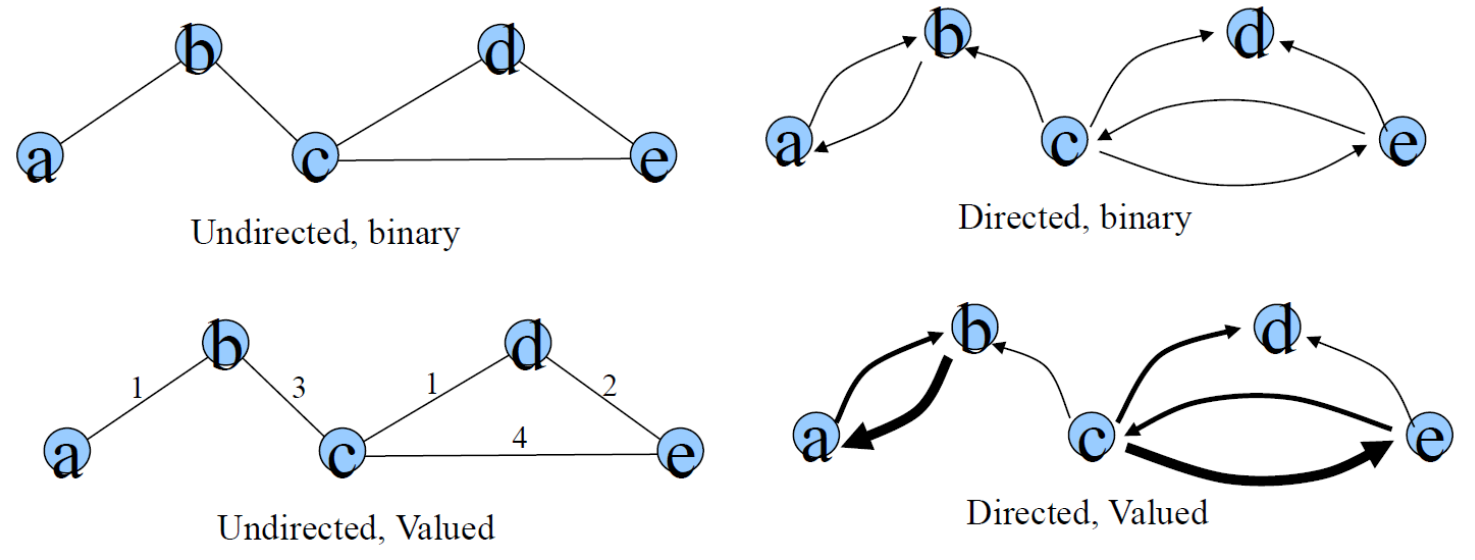
- 网络是节点和边的组合
- “事物” +“联系”
- 网络科学：跨科学



节点	边	学科
Vertices	Edges, arcs	Math
Nodes	Links	Computer Science
Sites	Bonds	Physics
Actors	Ties, Relations	Sociology

有向/无向图，加权和不加权

- 网络中的边可能**有向**也可能无向，也无各自表达不同含义
  - 无向边：或双向边，表示对称关系，如朋友或合作关系
  - 有向边：表示非对称关系，如通讯和关注关系
  - 有序时，图*G*称为**有向图（directed graph）**，否则称*G*为**无向图（undirected graph）**。
- **边的权重**：可以表示关系强度、交流频率、物理距离等



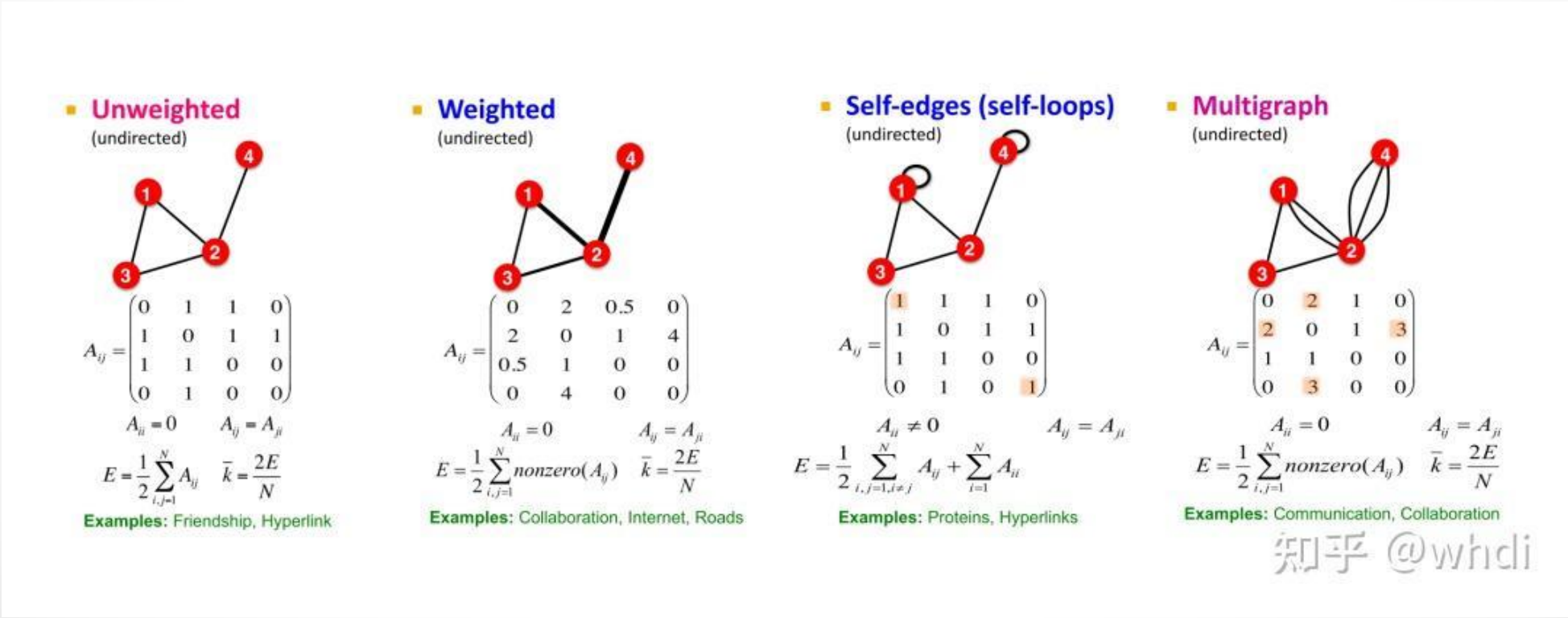
社会网络的基本元素

- 社交网络由许多节点构成的一种社会结构。节点通常是指个人或组织，而社交网络代表着各种社会关系。

- **行动者（Actor）**
  - 也称行为人，在社会网络中常常表现为节点，可以是个人、团队、组织或国家，构成关系的联结点。
  - 属性：区分节点的特征，如性别、年龄、个性等。
- **联结/联系/纽带（Tie）**
  - 关系的内容可能是亲属、友谊、借贷或是沟通，其关系可以是单向或双方，且存在关系强度的差异，关系不同构成不同的网络。
- **边界（Boundary）**
  - 这些行动者和边需要确定一个界限，比如社区内的用户，比如政府单位内，比如同性恋群体。

自环和多重边

- 完全图（complete graph），任意两点都有边相连
- 自环图（Self-edges (self-loops)），自己与自己相连。邻居矩阵的对角线不为0.
- 多重图 （ Multigraph ），存在两点之间大于一条边。

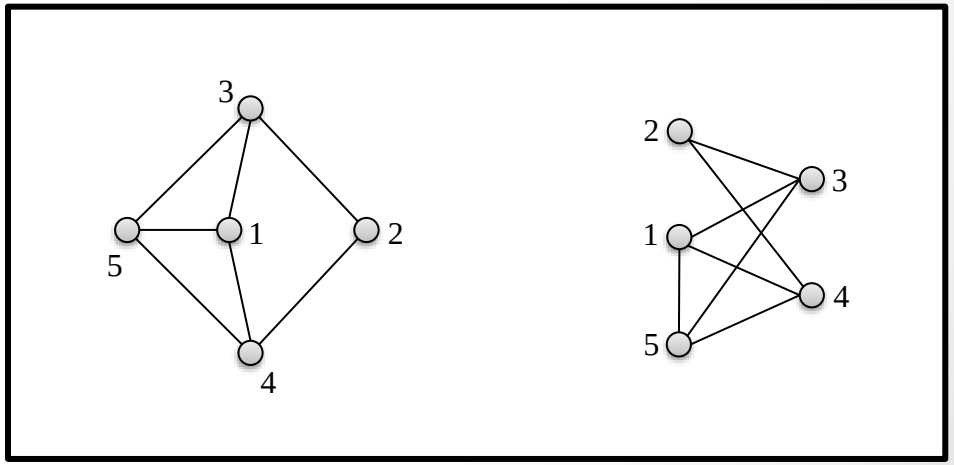


同构（Isomorphism）

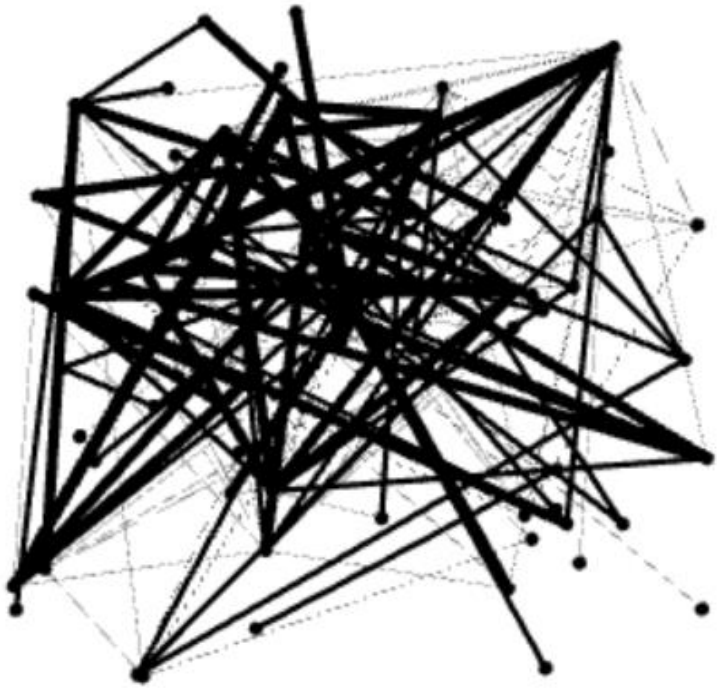
- 设有两个图$G_1 = (V_1, E_1)$和$G_2 = (V_2, E_2)$，若在其顶点集合间存在双映射，使得边之间存在如下关系：设$u_1 \leftrightarrow u_2$，$v_1 \leftrightarrow v_2$，$u_1, v_1 \epsilon V_1$，$u_2, v_2 \epsilon V_2$，$u_1 v_1 \epsilon E_1$，当且仅当$u_2 v_2 \epsilon E_2$，且$u_1 v_1$与$u_2 v_2$的重数相同。称G1与G2同构，记为：$G_1 \cong G_2$。

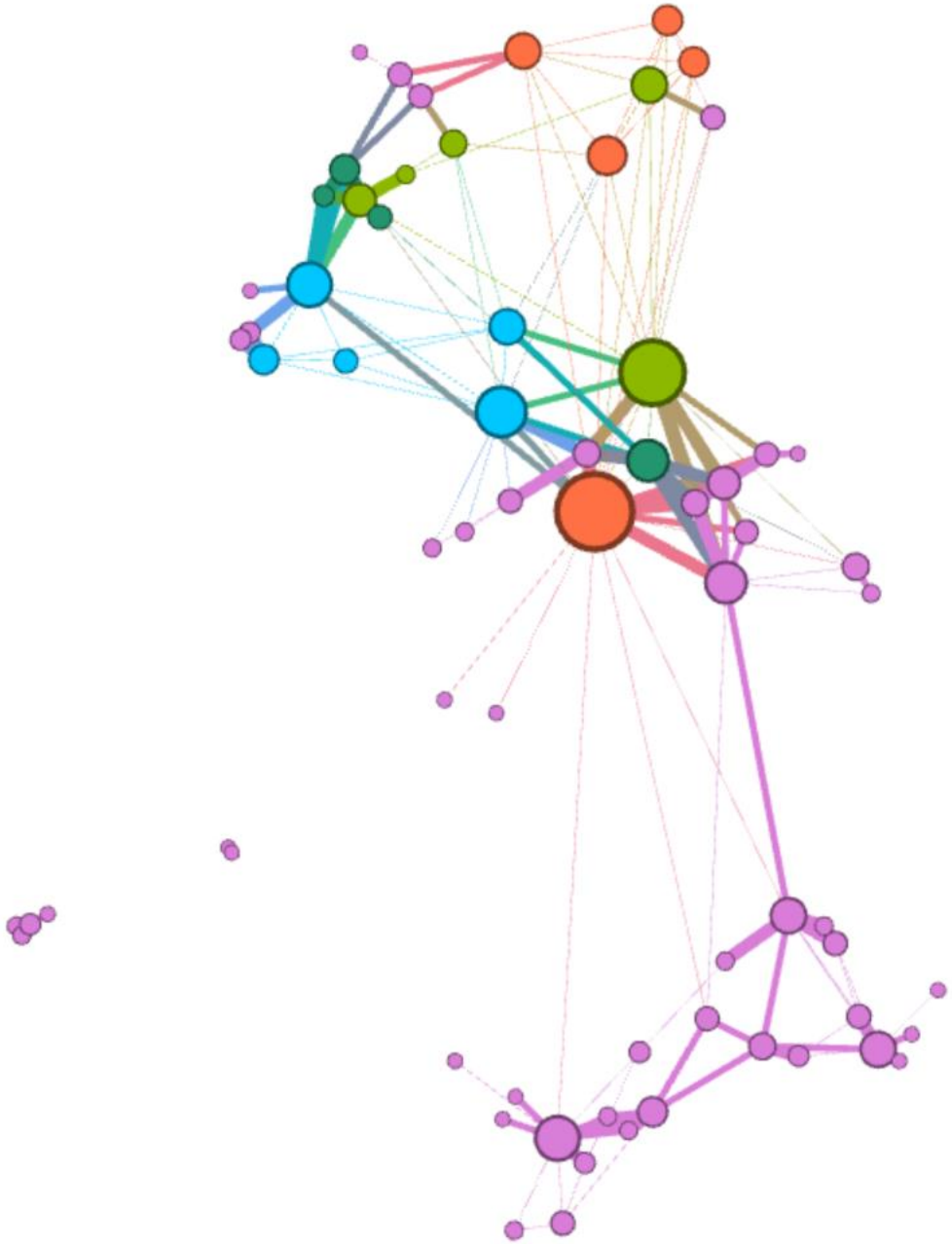
- 由定义可以得到图同构的几个必要条件：
- （1）顶点数相同；
- （2）边数相同；
- （3）关联边数相同的顶点个数相同。

- **画法不同，但本质上（结构上）相同。**



社交网络可视化





邻居和度（Degree）

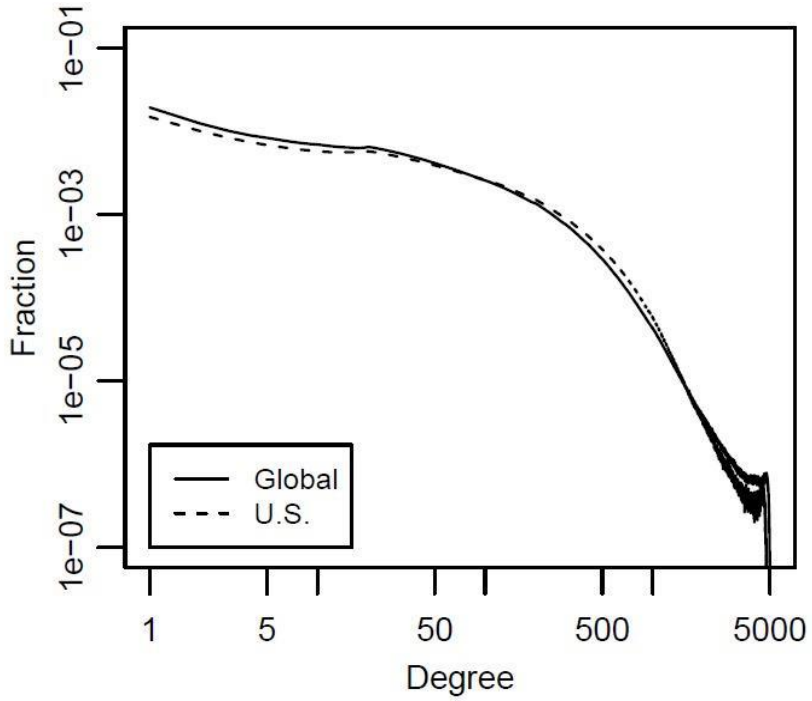
- 对于任意节点而言，与其直接相连的被称作“邻居”（Neighbor）
  - 对节点 v 而言，往采用 N(v) 表示其邻居集合
  - 在有向网络中，入边邻居和出边邻居应加以区分

- 同时，与顶点v相邻的顶点的数目称为v的**度（degree）**
  - 对节点v而言，往往采用d(v)表示其度数。
  - 在有向图G中，以顶点v为终点的边的条数称为v的**入度（in-degree）**，记为$d^-(v)$。
  - 以v为起点的边的条数称为v 的**出度（out-degree）**，记为$d^+(v)$。
  - 有向图中，顶点v的度数$d(v) = d^-(v) + d^+(v)$。

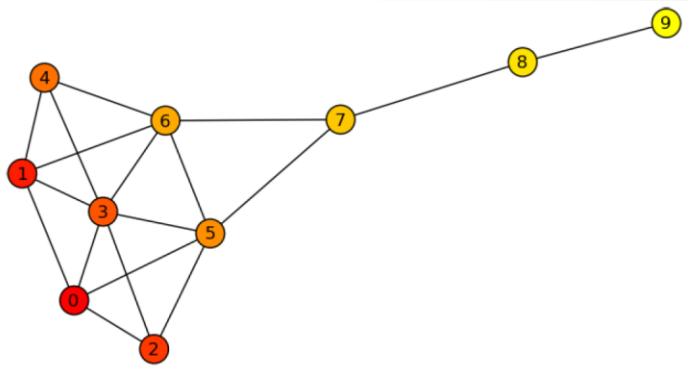
- 度为零的顶点称为**孤立顶点（isolated vertex）**。

邻居和度（Degree）

- 一般而言，真实网络中的节点度数往符合幂律（Power-law）分布
  - 少数节点拥有大多的边
  - 这些少数节点即形成了所谓“影响力节点”，也就是俗称的“大 V”



途径、轨迹、路径、回路



- **途径/通道（Walk）：**
  - 图G中一个顶点与边交错出现的**非空有限序列**$W=v_0e_1v_1e_2\cdots v_{k-1}e_kv_k$，使得对 $1\le i\le k$，边e_i以v_{i-1}，v_i为端点。则称W为从v_0到v_k的一条途径，v_0和v_k称为途径W的**起点**和**终点**，k称为**途径W的长**。若起点与终点相同时称为闭途径。
  - 在简单图中，途径W可以表示为顶点序列$W=v_0v_1\cdots v_{k-1}v_k$

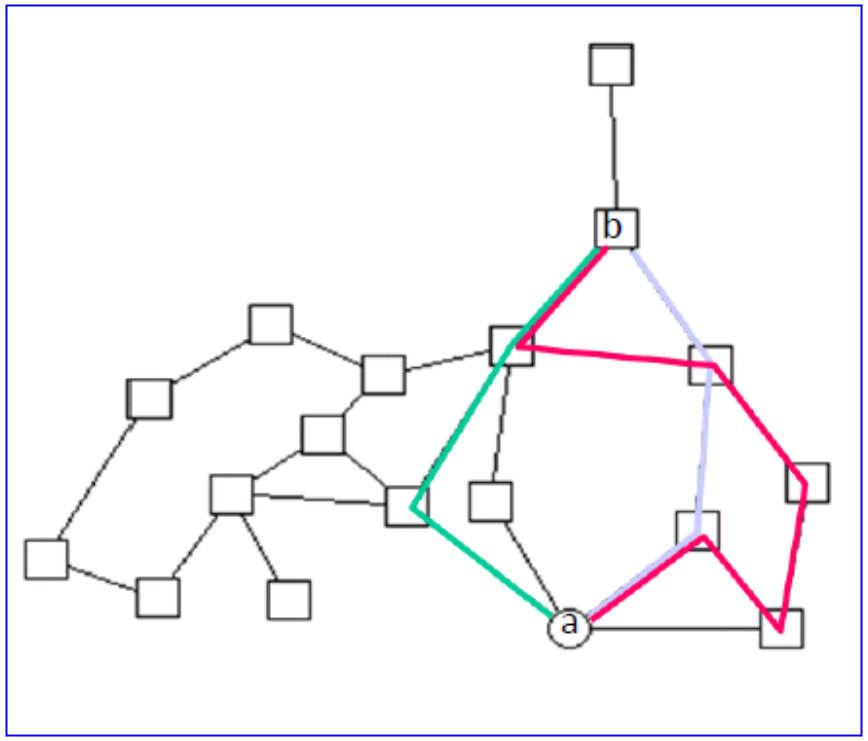
- **轨迹（Trail）：**
  - 若**途径W**的边$e_1, e_2, \cdots, e_k$**互不相同**，则W称为**轨迹/迹**。当$v_0 = v_k$时称为**闭迹**。

- **路径（Path）：**
  - 若**途径W**的**顶点**$v_1, v_2, \cdots, v_k$**也互不相同**，则W称为**路径/路**。 路所含的边数称为**路的长度**。

- **回路/圈（Cycle）：**
  - 一条内部顶点不重复出现的闭路称为**回路/圈**。长度为奇数的圈称奇圈，否则称偶圈。

最短路径（Shortest Path）

- 两个点之间可以选择的不同路径，最短路径尤为重要。
- 距离（Distance）：两点之间的最短路径，。
- 在赋权图中求某点到另一点的权值最小的路径。

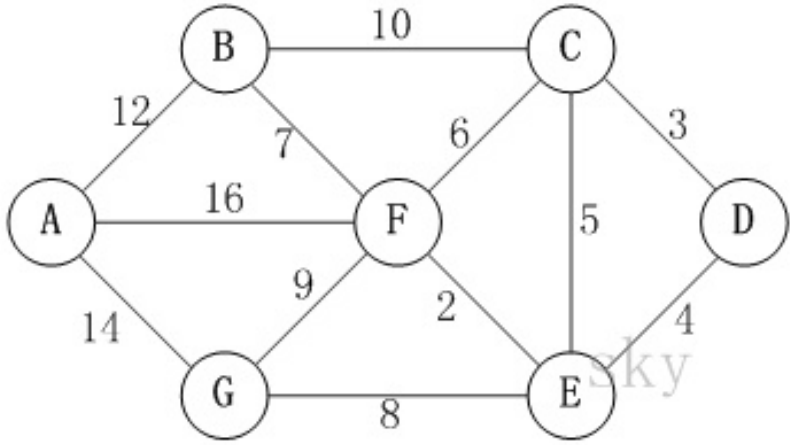


- a和b之间的最短路径:
- a和b之间的距离：3

最短路径算法（Dijkstra算法）

- 迪杰斯特拉于1959年提出的图搜索算法（Dijkstra's algorithm）：对于一个给定的顶点，可以计算出它到所有其他顶点的最低成本（即边的权重）路径。

1. 初始时，S只包含起点s；U包含除s外的其他顶点，且U中顶点的距离为”起点s到该顶点的距离”[例如，U中顶点v的距离为(s,v)的长度，然后s和v不相邻，则v的距离为∞]。
2. 从U中选出”距离最短的顶点k”，并将顶点k加入到S中；同时，从U中移除顶点k。
3. 更新U中各个顶点到起点s的距离。之所以更新U中顶点的距离，是由于上一步中确定了k是求出最短路径的顶点，从而可以利用k来更新其它顶点的距离；例如，(s,v)的距离可能大于(s,k)+(k,v)的距离。
4. 重复步骤(2)和(3)，直到遍历完所有顶点。



- **第1步：**初始化距离，其实指与D直接连接的点的距离。设置集合S用来表示已经找到的最短路径。此时，S={D}，U包含其他节点。 dis[c]代表D到C点的最短距离，因而初始dis[C]=3，dis[E]=4，其余为无穷大。现在得到D到各点距离U={D（0），C（3），E（4），F（*），G（*），B（*），A(*)}，其中*代表未知数也可以说是无穷大，括号里面的数值代表D点到该点的最短距离。

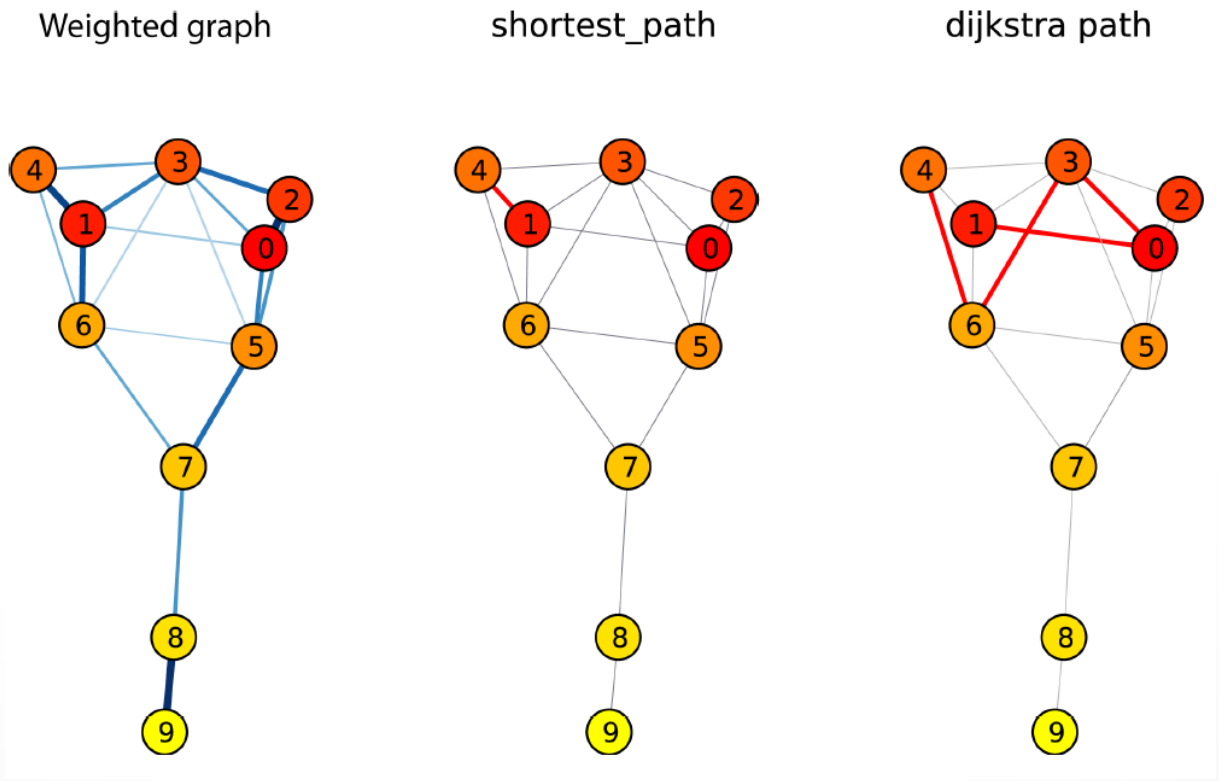
- **第2步：**不考虑集合S中的值，因为dis[C]=3，是当中距离最短的，所以此时更新S，S={D,C}。接着我们看与C连接的点，分别有B，E，F，已经在集合S中的不看，dis[C-B]=10，因而dis[B]=dis[C]+10=13，dis[F]=dis[C]+dis[C-F]=9，dis[E]=dis[C]+dis[C-E]=3+5=8>4(初始化时的dis[E]=4)不更新。此时{D(0)，C(3)，E（4），F（9），G（*），B（13），A(*)}。

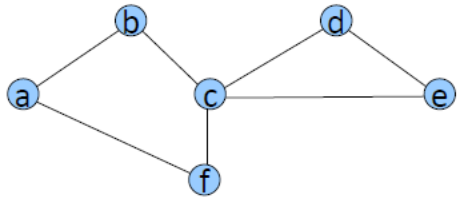
- **第3步：**在第2步中，E点的值4最小，更新S={D，C，E}，此时看与E点直接连接的点，分别有F，G。dis[F]=dis[E]+dis[E-F]=4+2=6（比原来的值小，得到更新），dis[G]=dis[E]+dis[E-G]=4+8=12（更新）。此时{D(0)，C(3)，E（4），F（6），G（12），B（13），A(*)}。

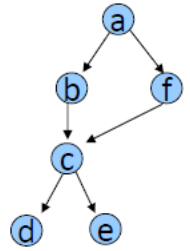
- **第6步：**最后只剩下A值，直接进入集合S={D，C，E，F，G，B，A}，此时所有的点都已经遍历结束，得到最终结果{D(0)，C(3)，E（4），F（6），G（12），B（13），A(22)}.

直径

- 直径（Diameter）：指G的任意两个顶点之间的最短距离最大值。
- 平均距离：所有两点之间距离加总，除以所有两点的组合数。
- 计算时忽略不相连的两点间距离，无穷大。







连通性（Connectivity）

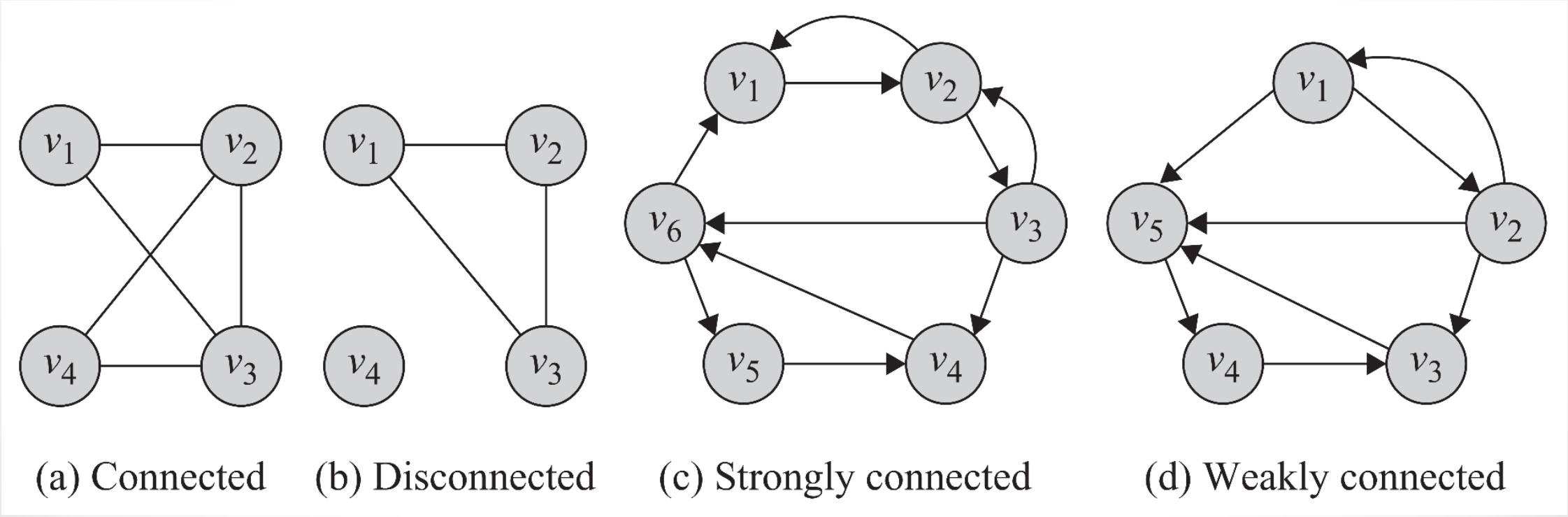
- 节点连通：G的两个顶点u和v称为连通的，如果G中存在(u, v)路径。
- 连通图：一个图是连通图（Component），当且仅当，如果G中的每一对顶点有至少一条路径连接

- 无向图的连通性
  - 若图G中任两个不同顶点都连通，则称此无向图是连通的（Connected），否则称为非连通图或者分离图（Disconnect）。

- 有向图的连通性，有强/弱联通的区别
  - 强连通（Strongly Connected）：任意两个节点之间存在双向的联通路径，即为强连通。
  - 弱连通（Weakly Connected）：忽略方向的前提下任意两个节点之间存在一条路径，即为弱连通。

连通性（Connectivity）

- 图的连通性实例



子图和连通分量

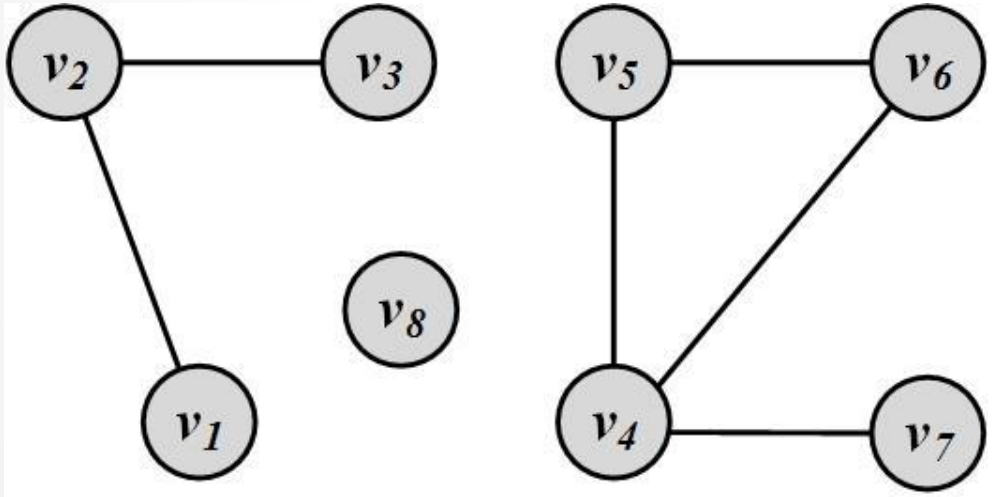
- **子图（Subgraph）**：设有两个图G =（V，E）和G’=（V’，E’），若V’是V的子集，E’是E的子集，则称G’为G的子图。

- **连通分支/分量（Connected Components）**：图G可分为几个不相连通的子图，每一子图本身都是连通的。称这几个子图为G的**连通分量**。
  - 换言之，该子图的任意两个节点之间都是连通的
  - 该子图本身不包含在任何其他连通子图中

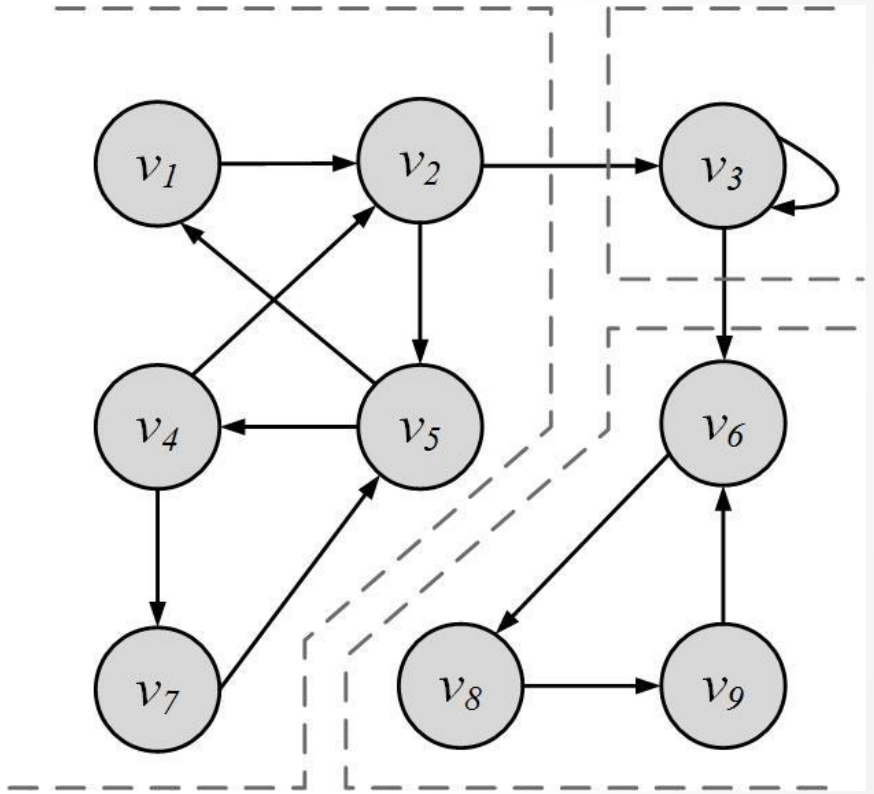
- 在有向图中，我们将**强连通分量**定义为该有向图的一个强连通子图
  - 即任意两个节点之间存在双向连通路径
  - 相应地，弱连通子图对应着弱连通

子图和连通分量

- 连通分量的实例



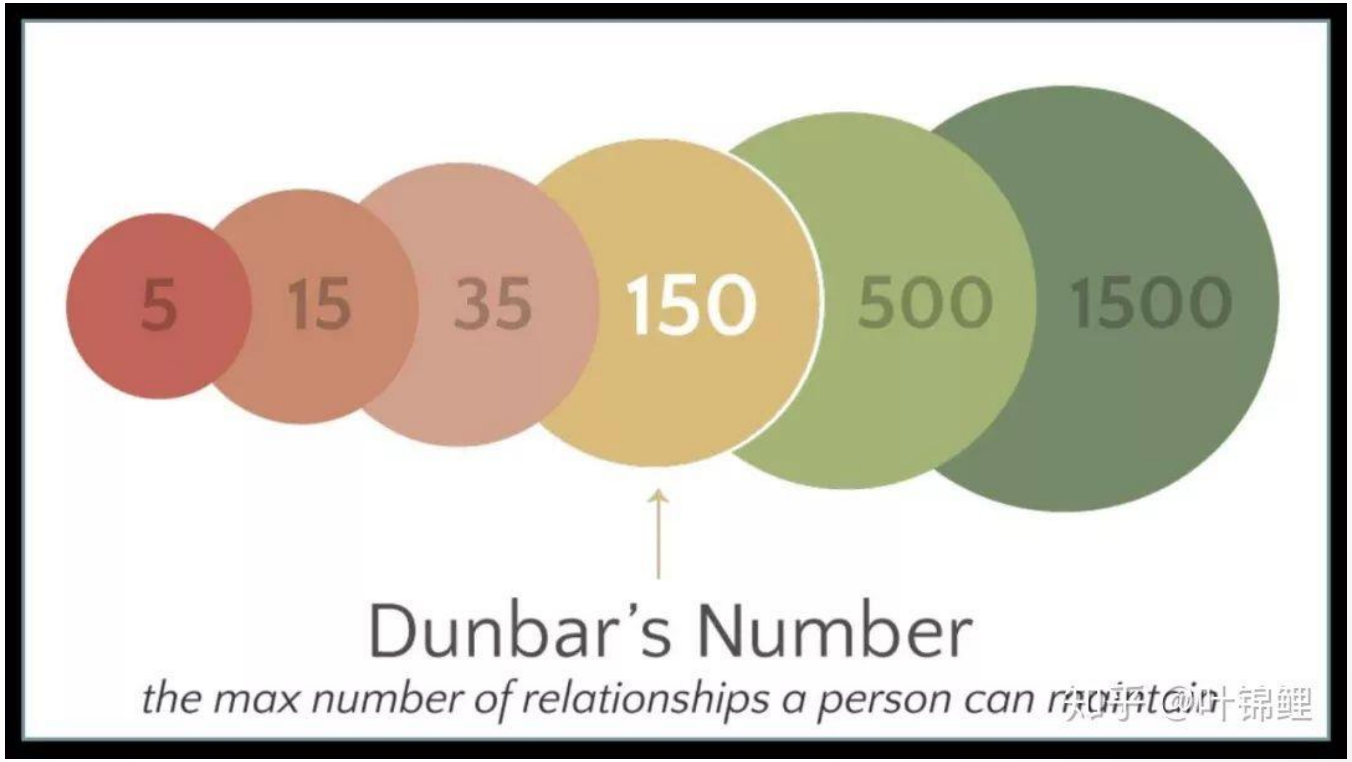
3个连通分量



3个强连通分量

连通分量的扩展：邓巴数字

- **罗宾·邓巴**，牛津大学进化心理学教授，社会遗传学在20世纪90年代提出
- **邓巴数字**（Dunbar Number），即**150法则**（Rule of 150）：个体所能够认识、信任，并在感情上**依赖**的人数不会超过150。



连通分量的扩展：邓巴数字

- **氏族群体的平均规模：**153
- **赫特兄弟会（Huntterites）和阿米什教派（Amish）：**
  - 一旦其共同生活的社区规模超过150人，他们就主动分开。
- **商业组织理论中常用的一个经验法则：**
  - 少于150人的企业可以在良好的人际关系基础上高效运作；一旦企业规模超过这个数字，就需要启动某种等级制度。
  - 150-200是一个关键的临界值，稍大一些的公司，员工的旷工和病假就会大幅增加。
- **商业中的格尔特斯（GoreTex）：**
  - 每个工厂的规模始终在150以下；
  - 在公司业务增长需要增加产能时，就坚持新建工厂，而不肯在原工厂增加人手。
- **军事中的基本作战单位：**
  - 其独立单位为连，其规模在130-150之间。

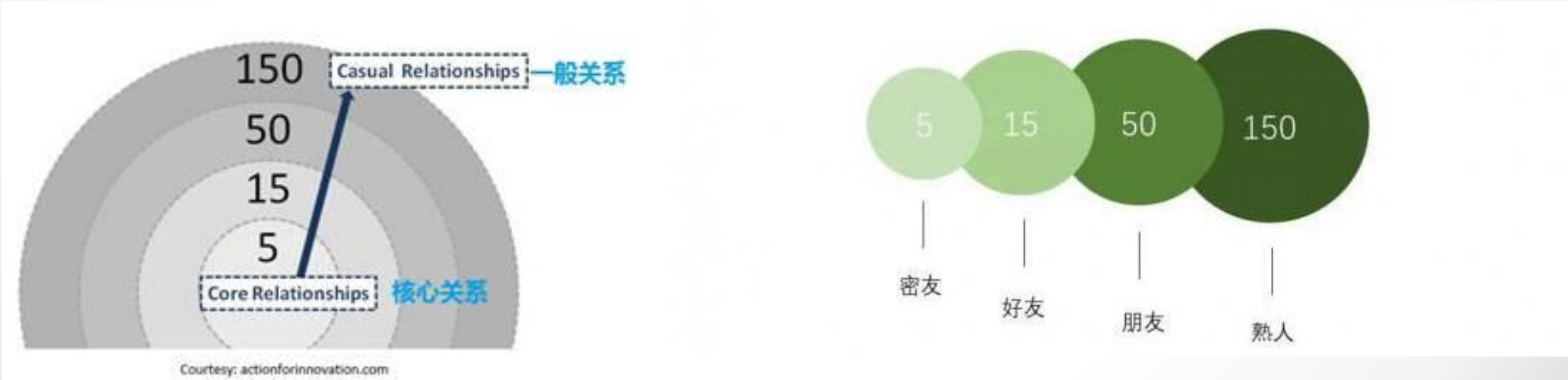
邓巴数字的成因

- **社交智力理论-马基雅维利智力假说**（Machiavellian intelligence hypothesis）
  - 灵长类动物与其他所有动物的最主要分别就是它们生活在复杂的社会关系中。
  - 对于各种非人类灵长类动物物种而言，其群体的规模，即社交世界的复杂性，与这些动物的新皮层（即大脑的外表层，主要负责意识思维）的相对大小密切相关。
  - **群体规模代表着一个动物所能维持的具有一定复杂性的社交关系的上限。**

- 150原则：人类智力所允许拥有稳定社交网络的规模大约是150人。
  - 不仅仅要知道谁是谁，或者x和y有何关系，二者与自己有什么关系，更要了解如何能够利用自己对于相关个体的认知，在需要的时候利用那些社交关系。
  - 人们需要通过合作来发挥潜能，但过大规模的网络将导致沟通效率的下降，最终导致团队的分裂。
  - 这个过程，无疑是连通分量拆分为更多子连通分量的过程。

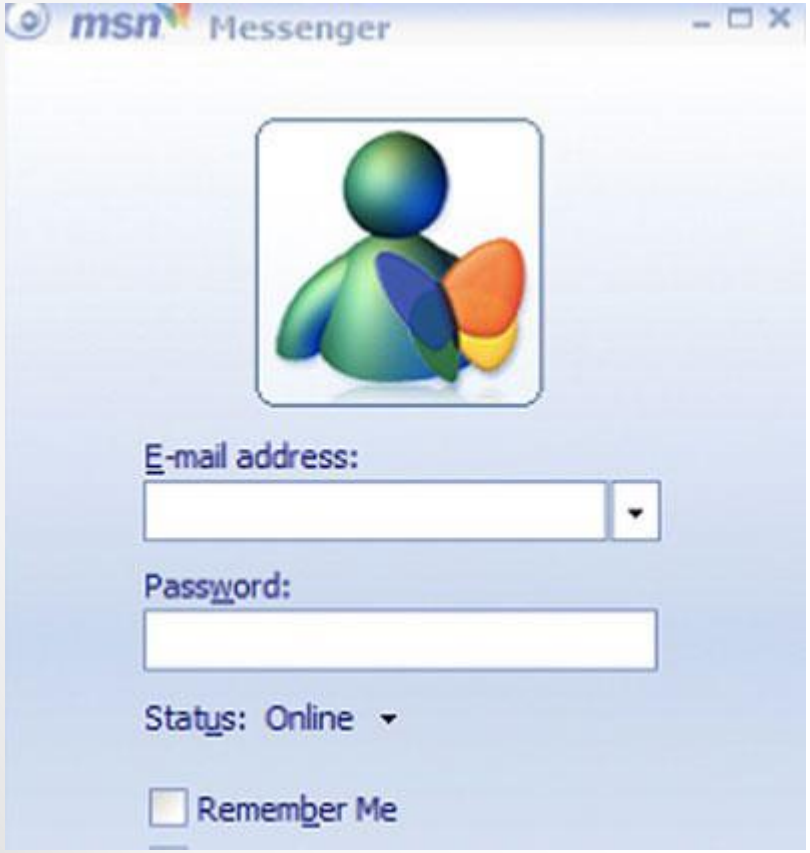
邓巴数字的成因

- 人们根据自己的感觉来区分朋友和熟人。在自己构成的社交圈的150人中，观察各种关系的模式，就会发现存在好几个亲密程度不同的圈子。
- 每个同心圆似乎都能够非常精准地映射出社会关系的两个方面：**联系的频率和亲密感**。
- 在一定的亲疏层次上，行为者能够与之相处的人数似乎是有限的。



邓巴数字：应用

- 中国移动的“动感地带”：SIM卡只能保存150个手机号
- 微软推出的聊天工具“MSN”：只能是一个MSN账号对应150个联系人



邓巴数字的应用

- 在线社交网站的出现对人类的社交视野与能力有什么影响?
- 在社交网站中，拥有200个好友和拥有2000个好友并没有实质的差别。

邓巴数字的应用

- 一个微信群限制最多为500人
- 一个QQ群限制最多5000人

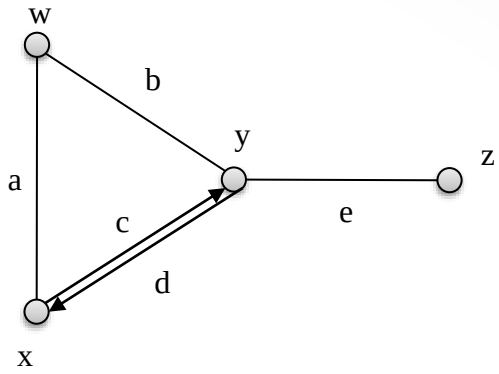
Chapter2 社会网络的基本概念

- 社会网络计算的发展历程
- 社会网络的基本概念
- **社会网络的存储与表示**
- 社会网络的可视化

网络的存储与表示

1. 矩阵（Matrix）表示
   - 邻接矩阵（Adjacency Matrix）表示
   - 关联矩阵（Incidence Matrix）表示
2. 邻接表（Adjacency List）表示
   - 边列表（Edge-List）表示
   - 邻接列表（Adjacency List）表示

矩阵表示

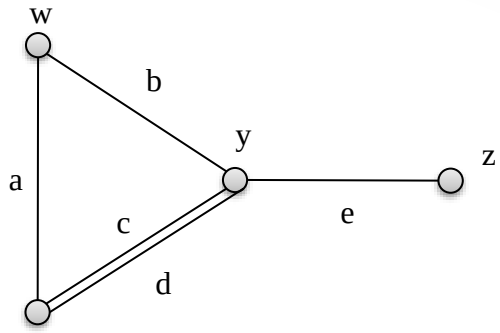


- 设图G=(V, E), | V | = n, | E | = m。
- 图G的**邻接矩阵**（adjacency matrix）是一个n×n矩阵，A(G) = (a_{ij}) n×n，其中a_{ij}是连接顶点v_i与v_j的边的数目。
- 图G的**关联矩阵**（incidence matrix）是一个n×m矩阵。

$$\begin{array}{c} \\ w \\ x \\ y \\ z \end{array}\begin{array}{c} \begin{array}{cccc} w & x & y & z \end{array} \\ \begin{pmatrix} 0 & a & b & 0 \\ a & 0 & c & 0 \\ b & d & 0 & e \\ 0 & 0 & e & 0 \end{pmatrix} \end{array}$$

$$\begin{array}{c} \\ w \\ x \\ y \\ z \end{array}\begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

邻接表表示



- 在真实的社交网络中，有很多“零”细胞——事实上，90%以上的“零”。非零单元与零单元的比率称为密度。大多数在线社交网络的密度只有0.1%或更低；一个社交网络越大，密度就越低。

- 边列表（Edge-List）

from	to	value
x	w	a
x	y	c
x	y	c
y	w	b
r	z	e

- 邻接列表（Adjacency List）

from	to
x	w, y, y
y	x, x, w, z
z	y
w	x, y

网络的存储格式：.net格式

```
*Vertices 3
1 "Node1" 0.0 0.0 0.0 ic Green bc Brown
2 "Node2" 0.0 0.0 0.0 ic Green bc Brown
3 "Node3" 0.0 0.0 0.0 ic Green bc Brown
*Arcs
1 2 3 c Green
2 3 5 c Black
*Edges
1 3 4 c Green
```

UCINET Software

Pajek

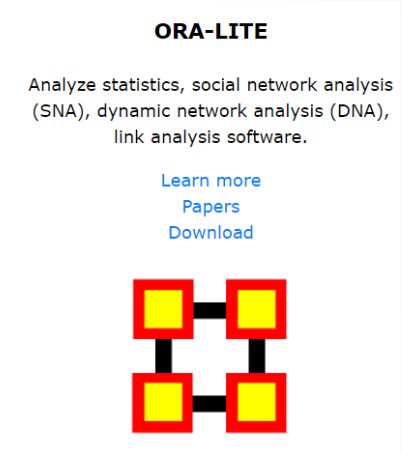
网络的存储格式：GML、GraphML

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml>
  <key id="d0" for="node" attr.name="color" attr.type="string">yellow</key>
  <key id="d1" for="edge" attr.name="weight" attr.type="double"/>
  <graph id="G" edgedefault="undirected">
    <node id="n0">
      <data key="d0">green</data>
    </node>
    <node id="n1"/>
    <node id="n2">
      <data key="d0">blue</data>
    </node>
    <node id="n3">
      <data key="d0">red</data>
    </node>

.... more node definitions....

    <edge id="e0" source="n0" target="n2">
      <data key="d1">1.0</data>
    </edge>
    <edge id="e1" source="n0" target="n1">
      <data key="d1">1.0</data>
    </edge>
    <edge id="e2" source="n1" target="n3"/>

.... more edges ....
  </graph>
</graphml>
```



ORA (Organizational Risk Analyzer)

http://www.casos.cs.cmu.edu/

Gephi

Chapter2 社会网络的基本概念

- 社会网络计算的发展历程
- 社会网络的基本概念
- 社会网络的存储与表示
- 社会网络的可视化

Gephi
makes graphs handy

Download Blog Wiki Forum Support Bug tracker

Home Features Learn Develop Plugins Services Consortium

The Open Graph Viz Platform

Gephi is the leading visualization and exploration software for all kinds of graphs and networks. Gephi is open-source and free.

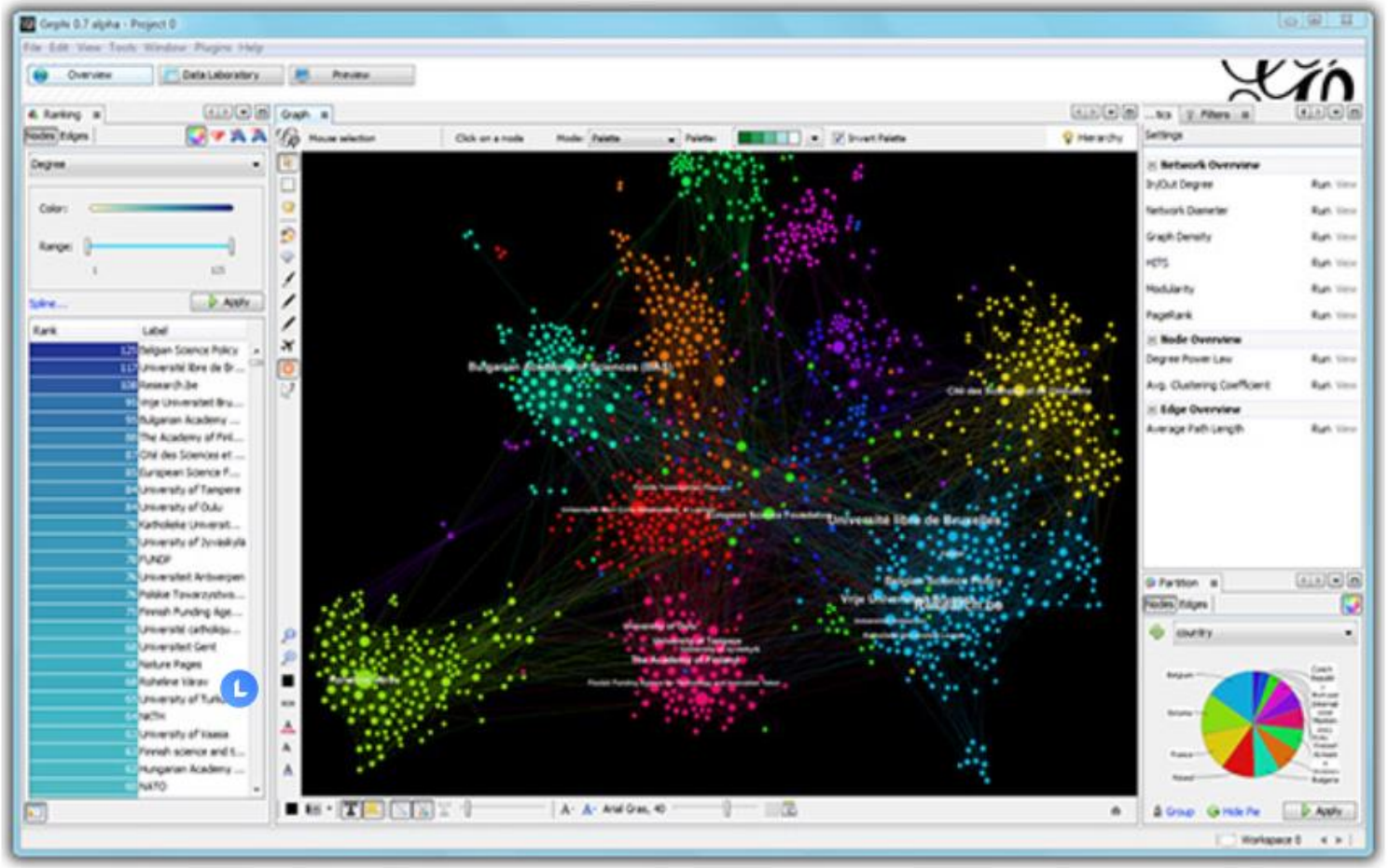
Runs on Windows, Mac OS X and Linux.

Learn More on Gephi Platform »

Download FREE
Gephi 0.9.2

Release Notes | System Requirements

- Features
- Quick start
- Screenshots
- Videos



https://gephi.org/

可视化：Gephi

- 数据表：911 hijackers.gexf
- 导入数据（Import Data）、修改外观（Color、Font）、修改布局（Layout）
- Gephi可以将数据导入后对图进行分割，调整外观和布局，得到可视化的结果。

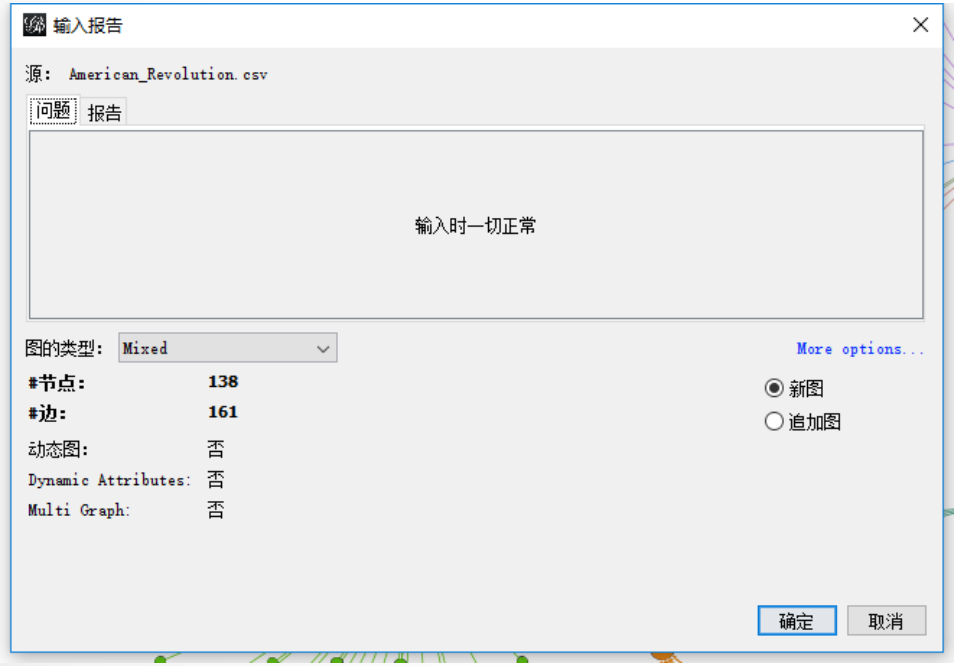


图1 将数据导入Gephi

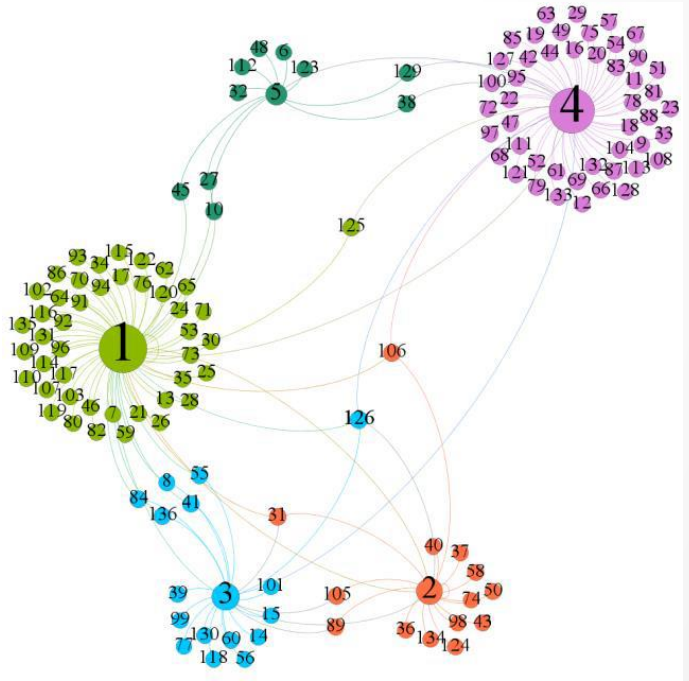


图2 Gephi处理后的可视化结果

Software for Complex Networks

NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

NetworkX provides:

- tools for the study of the structure and dynamics of social, biological, and infrastructure networks;
- a standard programming interface and graph implementation that is suitable for many applications;
- a rapid development environment for collaborative, multidisciplinary projects;
- an interface to existing numerical algorithms and code written in C, C++, and FORTRAN; and
- the ability to painlessly work with large nonstandard data sets.

With NetworkX you can load and store networks in standard and nonstandard data formats, generate many types of random and classic networks, analyze network structure, build network models, design new network algorithms, draw networks, and much more.

https://networkx.org/documentation/stable/

可视化：Networkx基础及绘图语句

- import networkx as nx

- #获得Karate数据
- G = nx.karate_club_graph()
- # 能量布局算法得到的初始图，如图1
- pos = nx.kamada_kawai_layout(G)
- nx.draw(G, pos, with_labels=True, node_size=500, node_color =‘red’)

- # 改变节点的大小和颜色
- nx.draw_networkx_nodes(G, pos=pos, nodelist=r1, node_size=1500, node_color='blue’, node_shape='*’)
- #绘制边，改变颜色和样式
- nx.draw_networkx_edges(G,pos=pos,edge,color='orange',style='dashdot’)
- # 画节点的标签
- nx.draw_networkx_labels(G,pos=pos, font_size=10,font_color='white')

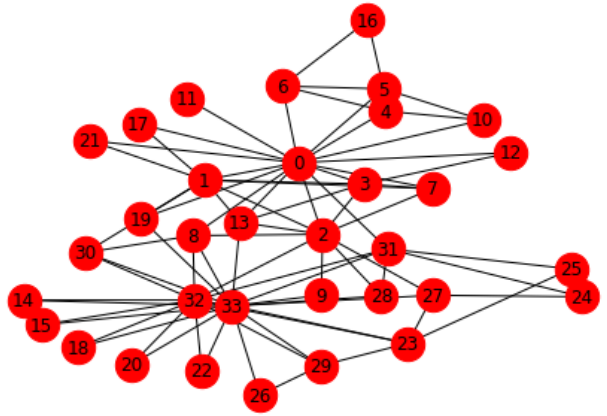


图1 Karate社会网络图

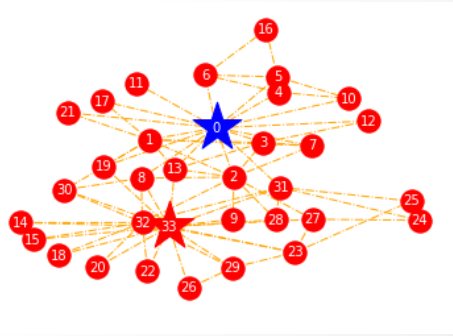


图2 调整后的Karate网络图