第 5 章

无标度网络

5.1 幂律分布及二八定律

复杂网络中的无标度网络是一类网络，其度分布（节点的连接数分布）遵循幂律分布。这意味着在无标度网络中，存在少数节点具有极高的度，而大多数节点具有较低的度，如图 5-1 所示。这种分布方式与传统的随机网络（如 ER 网络）的度分布有明显的不同，随机网络的度分布通常呈正态分布或泊松分布，其中大多数节点的度相对接近。

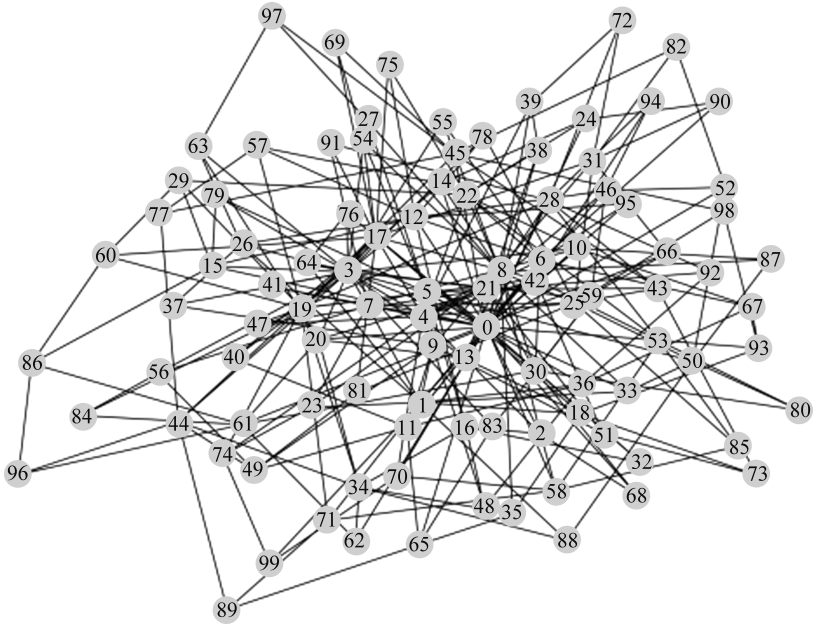

图 5-1　常见无标度网络

5.1.1 幂律分布

幂律分布是一种特殊的概率分布，其概率密度函数（Probability Density Function，PDF）与变量 x 的幂次关联。在无标度网络中，度分布 $P(k)$ 通常遵循幂律分布，表示为

$$P(k) \propto k^{-\gamma} \tag{5-1}$$

其中，$P(k)$是具有度 k 的节点的概率，γ 是幂指数，通常在 2～3。这个分布的重要特征是，当 γ 小于 2 时，尾部的长尾非常重，意味着存在少数节点具有极高的度，这些节点被称为“超级节点”或“中心节点”。

无标度网络的幂律分布导致了网络的脆弱性和鲁棒性之间的平衡。虽然网络中存在一些高度连接的节点，它们容易成为攻击目标，但与此同时，网络对随机节点的攻击具有较强的鲁棒性，因为大多数节点的度相对较低。

5.1.2　二八定律

与无标度网络相关的另一个概念是“二八定律”(80/20 Rule)，也称帕累托原理(Pareto Principle)，这是一种粗略的概括，指在无标度网络中，少数高度连接的节点(20%)贡献了网络中大多数的连接关系(80%)。这个规则与幂律分布密切相关，因为幂律分布导致了这种不平衡的连接分布。

二八定律的含义是，虽然网络中有大量的节点，但只有少数几个节点才具有极高的影响力。这些高度连接的节点在信息传播、网络鲁棒性、疾病传播等方面具有重要作用。这种不平衡的分布使得网络对特定节点的攻击或故障非常敏感，因为破坏其中一个关键节点可能会导致对整个网络的重大破坏。

总之，复杂网络中的无标度网络具有度分布遵循幂律分布的特点，导致网络中的不平衡连接分布。这种不平衡性在二八定律中得到了体现，强调了网络中少数节点的重要性，这些节点在信息传播、网络鲁棒性和许多其他复杂网络过程中发挥关键作用。同时，这也使网络更加脆弱，容易受到攻击或随机故障的影响。因此，研究无标度网络的结构和动力学对理解复杂系统的行为至关重要。

5.2　幂律分布的数据拟合

幂指数通常用 α 表示。在度分布的幂律形式中，概率密度函数 $P(X)$可以写成 $P(X=x)\propto x^{-\alpha}$。估计幂指数是评估幂律分布形状的关键步骤。

5.2.1　数据分箱

数据分箱是一种数据预处理技术，是用来处理幂律分布数据的方法。通过将原始数据进行分组，再对每一组内的数据进行平滑处理，使得在不同的度数范围内可以更准确地观察分布情况。常见的分箱的方式主要有等深分箱(每组数据一样多)、等宽分箱(每组区间长度一样)、用户自定义、最小熵(各分组内的数据具有最小熵)；平滑的方式主要有均值平滑(用组内均值代替组内每个元素)、中间值平滑(用组内中间值代替组内每个元素)、边界平滑(用组内离得较近的边界值代替组内元素)。

对于幂律分布，在做直线拟合时，采用对数分箱更能准确地估计幂指数。对数分箱(Log Binning)是一种处理幂律分布数据的常见方法。通过在对数尺度上对数据进行分箱，有助于平滑尾部的长尾分布，并提高对幂指数的估计的准确性。以下是使用对数分箱估计幂指数的一般步骤。

(1) 对数分箱。

① 将数据点 x 按对数尺度分箱。在对数分箱中,每个箱的宽度在对数尺度上是相等的。

② 选择对数分箱的基数,通常是以 2 为底或以 10 为底。例如,以 2 为底的对数分箱中,每个箱的范围为$[2^i, 2^{i+1})$。

(2) 统计每个箱中的数据点数量:计算每个对数箱中包含的数据点数量。

(3) 计算概率密度:计算每个对数箱的概率密度,即每个箱中数据点的数量除以总数据点的数量。

(4) 拟合幂指数:使用拟合方法,如最小二乘法,对概率密度和对数尺度下的箱中心值进行拟合。幂指数可以通过对拟合得到的直线的斜率进行估计。

下面使用对数分箱估计幂指数。

```
#coding=utf-8

import numpy as np
import matplotlib.pyplot as plt

#生成一个幂律分布的示例数据
data = np.random.pareto(2, size=1000)

#对数据进行对数分箱
logbins = np.logspace(np.log10(min(data)), np.log10(max(data)), num=20,
endpoint=True, base=10.0)

#统计每个箱中的数据点数量
hist, edges = np.histogram(data, bins=logbins)

#计算概率密度
pdf = hist / sum(hist)

#计算对数尺度下的箱中心值
bin_centers = (edges[:-1] + edges[1:]) / 2

#进行线性拟合(最小二乘法)
coefficients = np.polyfit(np.log10(bin_centers), np.log10(pdf), 1)

#提取斜率,即幂指数
alpha_estimate = -coefficients[0]

#绘制对数尺度下的度数分布和拟合直线
plt.scatter(bin_centers, pdf, label='数据')
plt.plot(bin_centers, 10**(np.polyval(coefficients, np.log10(bin_centers))),
label=f'拟合, α ≈ {alpha_estimate:.2f}', color='red')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('度数 (对数尺度)')
plt.ylabel('概率密度 (对数尺度)')
plt.legend()
plt.show()

print(f"估计的幂律指数(α): {alpha_estimate:.2f}")
```

输出结果为:估计的幂律指数(α):−0.16,如图 5-2 所示。

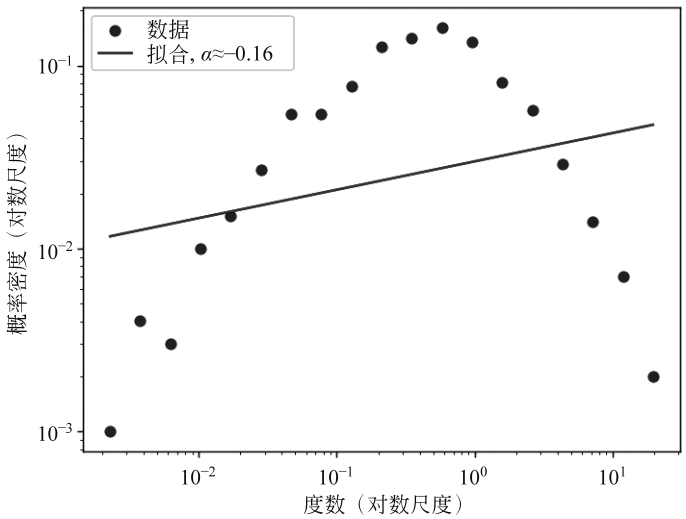

图 5-2　数据分箱拟合法

5.2.2　最小二乘法估计

最小二乘法(Least Squares Method)是一种常用的参数估计方法,其主要目标是通过最小化观测数据与模型预测值之间的残差平方和找到最优参数。

考虑一个幂律分布的形式为

$$P(X)=C\cdot x^{-\alpha} \tag{5-2}$$

其中,$P(X)$是随机变量 X 的概率密度函数,C 是归一化常数,α 是幂律指数。为使用最小二乘法拟合参数 α,可以执行以下步骤。

(1) 选择模型:确定使用的幂律模型形式,如 $P(X)=C\cdot x^{-\alpha}$。

(2) 取对数:为了线性化幂律模型,对概率密度函数取对数,即

$$\log(P(X))=-\alpha\cdot\log(x)+\log(C) \tag{5-3}$$

(3) 建立线性方程:将上述方程转换为线性形式 $y=mx+b$,其中 $y=\log(P(X))$,$x=\log(x)$,$m=-\alpha$,$b=\log(C)$。因此,可以得知

$$y=-\alpha\cdot x+\log(C) \tag{5-4}$$

(4) 最小二乘拟合:使用最小二乘法拟合线性方程,最小化残差平方和,即

$$\sum_{i=1}^{n}((-\alpha\cdot x_i+\log(C))-y_i)^2 \tag{5-5}$$

其中,(x_i,y_i)是观测样本数据点。

(5) 参数估计:从最小二乘拟合中提取参数,得到幂律指数 α 和归一化常数 C。

通过最小二乘法,可以估计幂律分布中的拟合参数 α,在最小化残差平方和的过程中,使得模型与观测数据的拟合误差最小化。需要注意的是,该方法在处理大量数据时可能会受到极端值的影响,因此在一些情况下可能需要使用其他拟合方法或考虑正则化技术。

下面使用最小二乘法估计幂指数。

```
#coding=utf-8

import numpy as np
from scipy.optimize import minimize
import matplotlib.pyplot as plt

#生成幂律分布的数据
np.random.seed(42)
data = np.random.pareto(a=2.5, size=1000)

#定义拟合函数(幂律分布)
def power_law(x, alpha):
    return x**(-alpha)

#定义对数最小二乘法的目标函数
def log_objective_function(alpha, data):
    return np.sum((np.log(data) + alpha * np.log(data.min()) - alpha * np.log
(data))**2)

#使用对数最小二乘法进行参数估计
result = minimize(log_objective_function, x0=[2], args=(data,), method='Powell')

#获取估计的参数值
alpha_estimate = result.x[0]

#绘制原始数据和拟合曲线
plt.hist(data, bins=50, density=True, alpha=0.7, label='观测数据')
x_values = np.linspace(min(data), max(data), 100)
plt.plot(x_values, power_law(x_values, alpha_estimate), 'r-', label=f'幂律函数
(alpha={alpha_estimate:.2f})')
plt.xlabel('X')
plt.ylabel('概率密度')
plt.legend()
plt.show()
```

其输出结果如图 5-3 所示。

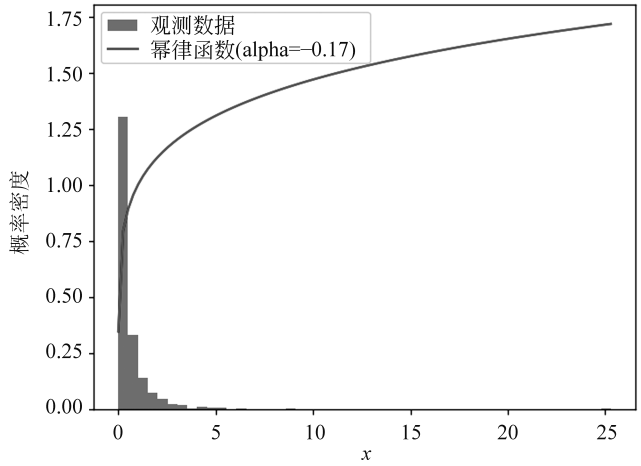

图 5-3 最小二乘法

5.2.3　极大似然估计

极大似然估计(Maximum Likelihood Estimation,MLE)也是一种常用于估计概率分布参数的方法,它的核心思想是寻找能够最大化给定观测数据出现的可能性的模型参数值。在幂律分布的情境下,极大似然估计可用于估计幂指数 α。

假设有一组观测数据 $X=\{X_1,X_2,X_3,\cdots,X_n\}$,这些数据是从一个幂律分布中独立地抽取的,希望找到其参数值,使得观测到的这组数据的似然最大。对于一个幂律分布,其概率密度函数为 $P(X=x)\propto x^{-\alpha}$。我们的目标是通过这组观测数据估计 α。

(1) 定义似然函数:对于一个样本 x_i,其概率密度函数为 $P(X=x_i)\propto x_i^{-\alpha}$。由于样本是独立同分布的,整体的似然函数可以表示为样本的联合概率密度函数的乘积,即

$$L(\alpha,C\mid X)=\prod_{i=1}^{n}P\times(X=x_i)=\prod_{i=1}^{n}C\cdot x_i^{-\alpha} \tag{5-6}$$

(2) 对数似然函数:为了方便计算,通常取对数似然函数(Log-Likelihood),即

$$l(\alpha,C\mid X)=\ln L(\alpha,C\mid X)=\sum_{i=1}^{n}[\log(C)-\alpha\cdot\log(x_i)] \tag{5-7}$$

(3) 对函数两边求导数:对对数似然函数关于参数 α 和 $\log(C)$ 分别求偏导数,令其等于零,即

$$\begin{cases}\dfrac{\partial l}{\partial\alpha}=-\displaystyle\sum_{i=1}^{n}\frac{\log(x_i)}{n}+\frac{\alpha}{n}\sum_{i=1}^{n}\log(x_i)=0\\[2ex]\dfrac{\partial l}{\partial\log(C)}=n-\dfrac{\alpha}{n}\displaystyle\sum_{i=1}^{n}\log(x_i)=0\end{cases} \tag{5-8}$$

(4) 解方程求估计值:通过解方程,得到使对数似然函数最大化的参数值,即 α 和 $\log(C)$ 的估计值。

(5) 参数估计:从最大似然估计中提取参数,得到幂律指数 α 和归一化常数 C。

其中值得注意的是,在取对数似然函数时,对数中的 C 变成 $\log(C)$,因此在最大化似然估计中,通常估计出的是 $\log(C)$ 的值而不是 C 的值。因此在实际应用中,为得到 C,可以通过对估计的 $\log(C)$ 进行反对数运算而得到 C 的值。

在实践中,通常使用统计软件工具(如 Python 中的 scipy.stats.powerlaw.fit 或 R 中的 powerlaw 包)进行极大似然估计。以下是该方法的 Python 代码。

```
import numpy as np
from scipy.stats import powerlaw
import matplotlib.pyplot as plt

#生成幂律分布的随机样本
alpha_true = 2.5
data = powerlaw.rvs(alpha_true, size=1000)

#进行极大似然估计
fit_alpha, fit_loc, fit_scale = powerlaw.fit(data, floc=0)

#绘制拟合结果与原始数据的对比图
plt.figure(figsize=(10, 6))
#绘制原始数据的直方图
```

```
plt.hist(data, bins=50, density=True, alpha=0.7, color='blue', label='原始数据')
x = np.linspace(min(data), max(data), 100)
y_fit = powerlaw.pdf(x, fit_alpha, loc=fit_loc, scale=fit_scale)
plt.plot(x, y_fit, 'r--', linewidth=2, label=f'拟合: alpha={fit_alpha:.2f}')

#输出真实的 alpha 值和拟合的 alpha 值
print(f"真实的 Alpha: {alpha_true}")
print(f"极大似然估计的 Alpha: {fit_alpha}")

plt.xlabel('数值')
plt.ylabel('概率分布')
plt.legend()
plt.show()
```

其输出结果如下(见图 5-4)。

```
真实的 alpha: 2.5
极大似然估计的 alpha: 2.5739469787580123
```

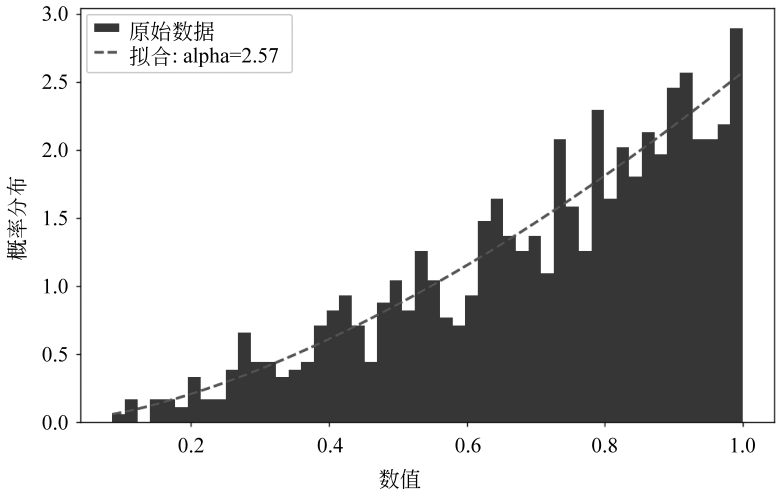

图 5-4 极大似然估计法

极大似然估计提供了一种基于数据拟合参数的强大方法,但需要注意的是,样本的大小对估计的稳定性和准确性有一定影响,特别是在样本较小的情况下。因此,在使用极大似然估计进行估计时,应注意样本大小和可能的偏差。

5.2.4 累计度分布

累计度分布是指网络中度数不小于某一给定值的节点数在总节点数中所占的比例。在幂律分布的情境下,累计度分布通常可以用 $P(X \geqslant x) \propto x - \alpha + 1$ 表示。

(1) 数据准备:首先需要获取网络的度分布数据,可以通过度数序列(Degree Sequence)或度分布直方图得到。

(2) 排序:将度数按照从大到小的顺序排列,形成有序序列。

(3) 计算累计概率:计算每个度数对应的累计概率,即大于或等于该度数的节点在总节点数中所占的比例。

(4) 拟合幂律分布:对于幂律分布 $P(X)$,先通过积分得到其累计度分布,再通过对累计度分布取对数可得其幂律指数 α。其累计度分布和取对数后公式为

$$\begin{cases} P(X \geqslant x) = \left(\dfrac{x}{x_{\min}}\right)^{1-\alpha} \\ \log P(X \geqslant x) = (1-\alpha) \cdot \log x + C \end{cases} \tag{5-9}$$

其中 $x_{\min}$ 是数据的最小值,用于归一化。在双对数坐标下,累计度分布应该呈现一条直线,直线的斜率为 $1-\alpha$,通过线性回归拟合这条直线即可得到幂律指数 α。

以下是一个简单的 Python 代码示例,使用网络 NetworkX 库生成随机网络,并拟合幂律函数。

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

#生成幂律分布的随机样本
alpha_true = 2.5
data = np.random.pareto(a=alpha_true, size=1000)

#计算累计度分布
sorted_data = np.sort(data)
cdf = 1.0 - np.arange(1, len(sorted_data) + 1) / len(sorted_data)

#定义幂律分布的累计度分布函数
def cumulative_power_law(x, alpha):
    return (x ** (-alpha))

#进行拟合
params, covariance = curve_fit(cumulative_power_law, sorted_data, cdf)

#获取估计的幂指数 alpha
alpha_estimate = params[0]

#绘制原始数据的累计度分布和拟合曲线
plt.figure(figsize=(10, 6))
plt.scatter(sorted_data, cdf, s=10, color='blue', label='经验累计分布')
x = np.linspace(min(data), max(data), 100)
plt.plot(x, cumulative_power_law(x, alpha_estimate), 'r-', label=f'拟合: α=
{alpha_estimate:.2f}')
plt.xscale('log')
plt.yscale('log')
plt.xlabel('数值 (对数尺度)')
plt.ylabel('累计概率 (对数尺度)')
plt.legend()
plt.show()

print(f"真实的 alpha: {alpha_true}")
print(f"拟合的 alpha: {alpha_estimate}")
```

输出结果如下(见图 5-5)。

```
真实的 alpha: 2.5
拟合的 alpha: -0.07943084648515915
```

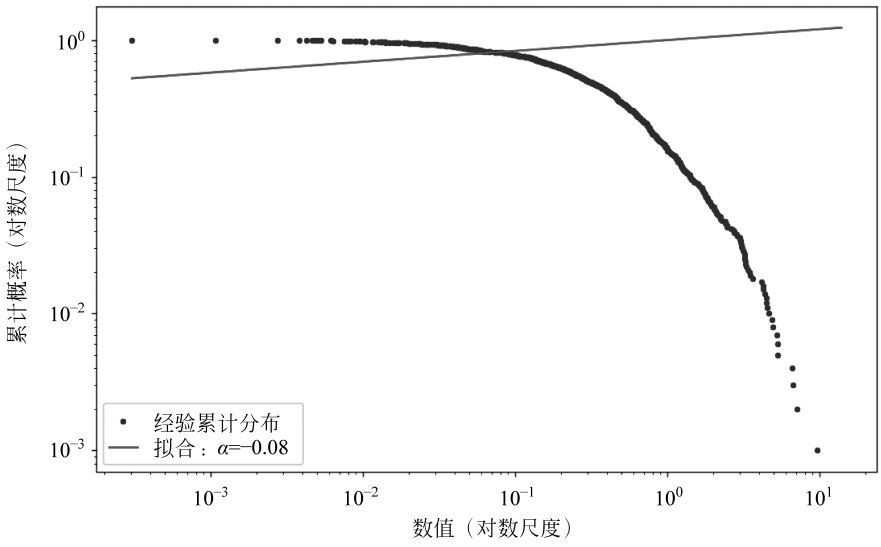

图 5-5 累计度分布

5.3 幂律分布网络的结构

幂律分布网络的结构是近年来复杂网络研究领域的一个重要分支。这种网络结构的特点在于网络中的大多数节点拥有较少的连接数，而少数节点则拥有非常多的连接数。这种网络结构并没有一个明显的“中心”，也就是说，没有一个或几个特定的节点可以占据网络的绝对主导地位。这种分布形态在现实生活中也有很多应用场景，比如互联网上的网页链接、社交网络中的用户关系等。以下是幂律分布网络的结构特点。

（1）无标度性：幂律分布最显著的特点是它的无标度性。在幂律分布中，无论将横坐标（即事件的规模）缩小多少，或者将纵坐标（即事件发生的概率）扩大多少，幂律分布的形状都不会发生改变。这种无标度性使得幂律分布在许多自然和社会现象中都广泛存在。

（2）长尾现象：由于尾部概率分布非常缓慢，因此在双对数坐标系中，幂律分布的图形呈现斜坡形态，类似一个长尾巴。

幂律分布网络的尾部概率分布缓慢下降，这意味着在事件的尺度变大时，事件发生的概率虽然会降低，但是降低的速度非常缓慢。这使得幂律分布在描述一些极端事件时非常有用，比如在描述地震、雪崩等自然现象时，由于这些现象的发生往往是由一系列小规模的扰动累积而成的，因此幂律分布能够很好地描述这些现象。

（3）自相似性：对于幂律分布的网络，无论将其观察范围缩小还是扩大，其网络结构都保持一致，即网络在不同尺度上具有相似性。这种自相似性使得幂律分布在描述一些具有分形结构的现象时非常有用。

（4）发散性：幂律分布的发散是指幂律指数小于或等于某个值时，幂律分布的均值和方差会变得无穷大。一阶原点矩发散是指幂律指数小于或等于 1 时，幂律分布的一阶原点矩（即均值）发散。这意味着，随着幂律分布的变量取值增大，其对应的概率密度将趋于 0，而对应的累积概率则将趋于 1。二阶中心矩发散是指当幂律指数小于或等于 2 时，幂律分

布的二阶中心矩(即方差)发散。K 阶矩发散是指当幂律指数小于或等于 K 时,幂律分布的 K 阶矩发散。

(5) 稳健性:即使在网络中加入或删除一些节点和连接,幂律分布的网络结构仍能保持相对稳定的状态。这种自组织性在网络演化过程中起到关键作用,使得网络能够在不断变化的环境中保持相对稳定。

总之,幂律分布的网络结构是一种非常特殊的网络结构,它具有无标度、长尾和自组织等特点。这些特点使得幂律分布的网络结构在现实生活中具有广泛的应用前景,比如在社交网络分析、网络流量控制等领域都有重要的应用价值。

5.4　BA 无标度网络模型

大多数现实网络的分布并不像随机网络那样呈现泊松分布,特别是大尺度的网络,如 Internet 及一些新陈代谢网络等,它们都具有幂指数形式的度分布,即 $P(k)\propto k^{-r}$。其泊松分布曲线下降缓慢。具有这种度分布形式的网络称为无标度网络,因此幂律分布也称为无标度分布。

BA(Barabási-Albert)网络是一种常见的无标度网络模型,是 1999 年由美国圣母大学物理系的阿尔伯特-拉斯洛·巴拉巴西(Albert-László Barabási)教授和其博士生雷卡·阿尔伯特(Réka Albert)在 *Science* 杂志上发表的一篇题为《随机网络中的标度的涌现》的论文上提出的,该论文揭示了复杂网络的无标度特性,因此建立了无标度网络模型。这个模型能够生成具有幂律分布度数的网络,其中一些节点拥有大量连接,而大多数节点只有较少的连接。

无标度网络在许多真实网络上都有出现,如互联网、万维网、科学合作网络等。例如,在因特网网页超链接中,一些网页(节点)被大量其他网页链接(连接度很高),而大多数网页只被相对较少的其他网页链接,且新建立的网站明显更倾向链接其他高知名度的网站。一些热门网页,如搜索引擎的主页,社交媒体页面等,具有大量的入站链接,形成网络中连接度较高的节点;在生物系统中的蛋白质相互作用网络中,一些关键的蛋白质(节点)在细胞内相互作用频繁,而大多数蛋白质只与少数其他蛋白质发生相互作用;在演员合作网络中,演员(节点)通过合作参与电影或戏剧项目,一些著名的演员可能与大量其他演员一起工作,形成网络中连接度较高的节点,而大多数演员可能只有有限的合作关系;表现最明显的是在线社区的社交网络,一些用户可能有大量的关注者或朋友,而大多数用户可能只有一小部分的社交连接,如 Facebook 或 Twitter,用户(节点)之间的连接关系形成一个无标度网络。

5.4.1　BA 无标度网络的构建

在 BA 模型中,新节点更倾向与那些已经拥有较多连接的节点相连。这种偏好依附机制是导致幂律分布的关键。每个已有节点的连接概率是它当前连接数的函数,即 $P(A \text{ connects to } B)\propto \text{degree of } B$。其网络模型的构建过程如下。

增长(Growth)阶段:选择 m_0 个节点作为初始节点,每个时间步新增一个带有 m 条边的节点,并且与已经存在的节点随机建立连接,其中满足条件 $m\leqslant m_0$。

优先连接(Preferential Attachment)阶段:在网络的发展过程中,新加入的节点更有可能连接到已经具有较高连接度的节点。一个新节点与一个已经存在的节点 v_i 相连接的概