

# ネットワーク分析の基礎と実践

津川 翔

2021年10月4日

# 目次

第 1 章	はじめに	3
1.1	様々なネットワーク	3
1.2	ネットワーク分析の活用例	6
1.3	本稿のねらいと構成	8
1.4	ネットワーク分析のためのツール	8
第 2 章	ネットワークの基礎	11
2.1	基本的な用語	11
2.2	ネットワークの表現方法	13
2.3	隣接行列とエッジリスト	15
2.4	次数	16
2.5	経路	16
2.6	連結	16
2.7	より複雑なネットワークの表現方法	17
2.8	本稿の扱うネットワーク	19
第 3 章	ネットワークの特徴量	22
3.1	特徴量を計算する前に：ネットワークの可視化	22
3.2	次数分布、平均次数	24
3.3	平均最短経路長	26
3.4	クラスタリング係数	27
3.5	実ネットワークの特徴量の計算	28
第 4 章	中心性解析	34
4.1	導入	34
4.2	次数中心性	35
4.3	近接中心性	35
4.4	媒介中心性	36
4.5	固有ベクトル中心性	38
4.6	PageRank	38

4.7	コアネス ( $k$ -コア指標) . . . . .	39
4.8	実ネットワークの中心性指標の計算 . . . . .	40
4.9	さらなる学習のために . . . . .	43
第 5 章	コミュニティ抽出 . . . . .	44
5.1	コミュニティとは? . . . . .	44
5.2	Girvan-Newman 法 . . . . .	47
5.3	モジュラリティ . . . . .	48
5.4	Newman 法 . . . . .	50
5.5	Louvain 法 . . . . .	51
5.6	実ネットワークのコミュニティ抽出 . . . . .	52
5.7	コミュニティ抽出に関する議論 . . . . .	58
5.8	さらなる学習のために . . . . .	59
第 6 章	ネットワーク分析の実践 1：ネットワークのロバスト性 . . . . .	60
6.1	導入 . . . . .	60
6.2	ネットワークのロバスト性評価のためのプログラム . . . . .	61
6.3	インターネットのアキレス腱 . . . . .	64
第 7 章	ネットワーク分析の実践 2：ネットワーク上のウィルス拡散 . . . . .	67
7.1	導入 . . . . .	67
7.2	SIR モデル . . . . .	68
7.3	シミュレーションプログラム . . . . .	69
7.4	ネットワーク構造がウィルス拡散に与える影響 . . . . .	71
第 8 章	ネットワーク分析の実践 3：Twitter のエゴネットワーク解析 . . . . .	76
8.1	導入 . . . . .	76
8.2	Twitter API を用いたソーシャルネットワークの収集 . . . . .	78
8.3	Twitter アカウントのエゴネットワークの特徴分析 . . . . .	80
第 9 章	ネットワーク分析の実践 4：研究者ネットワークの解析 . . . . .	86
9.1	信学会 MIKA 関係研専研究者ネットワークの構築 . . . . .	86
9.2	信学会 MIKA 関係研専研究者ネットワークの分析 . . . . .	90
第 10 章	おわりに . . . . .	97
10.1	最近の課題へのポインタ . . . . .	97
10.2	まとめ . . . . .	98
参考文献		100

# 1

## はじめに

インターネットにおけるルータ間の接続関係、Web ページ間のハイパーアクション関係、企業間の取引関係、社会における人と人との友人関係など、現実世界の様々なモノ同士の関係は「ネットワーク」として表現できる。現実の様々なネットワークの特徴を定量化し、ネットワークから有用な知見を抽出するための手法が、本稿で解説する「ネットワーク分析」である。本稿は、ネットワーク分析の初学者を対象とし、基本的な概念から実践的なネットワーク分析の活用例までを解説することを目指している。本章ではまず、ネットワーク分析とは何か、どのように有用であるのかを解説し、ネットワーク分析を学ぶ動機付けを行う。

### 1.1 様々なネットワーク

「ネットワーク」という単語から想像するものは何であろうか？日常会話において「ネットワーク」という単語を用いる時、多くの場合は通信ネットワーク（インターネット）のことを意味するであろう。本稿の読者の多くにとっても、馴染み深いネットワークは通信ネットワークであると思われる。

本稿における「ネットワーク」という単語は、何らかの対象間の「つながり」をノード（点）とリンク（線）で表現したものを意味する（図 1.1）。インターネットは、ノードをルータ、ルータ間の接続関係をリンクとしたネットワークである。人をノード、人と人の友人関係をリンクとしたソーシャルネットワークも、本稿におけるネットワークの一種である。図 1.2 にいくつかの現実のネットワークを可視化した例を、また表 1.1 にネットワークとそのノードおよびリンクの例を示す。このように、現実の様々な対象が、ネットワークとして表現される。

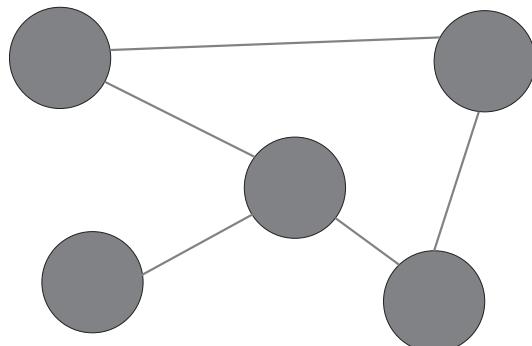
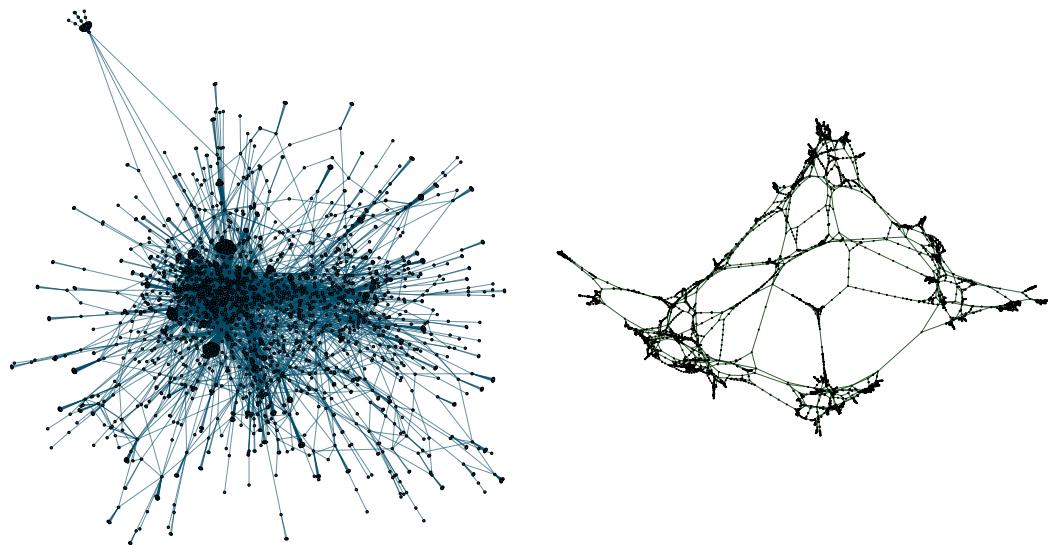


図 1.1: 本稿におけるネットワークの例

表 1.1: 様々なネットワークとそのノードおよびリンクの例

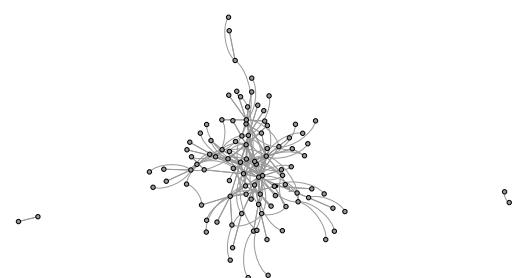
ネットワーク	ノード	リンク
インターネット	ルータ	IP リンク
ソーシャルネットワーク	人	友人・知人関係
企業の取引関係	企業	取引関係
電力網	発電所/変電所	送電線
道路網	都市	道路
タンパク質ネットワーク	タンパク質	相互作用
脳機能ネットワーク	脳領域	脳活動の相関

ネットワークは、現実の複雑な系を「適度に」抽象化するのに適した表現方法である。人の社会やインターネット、脳機能など、ネットワークとして表現される対象は非常に複雑であるが、ノードとリンクで抽象化して表現することで、数理的な手法や計算機を用いた手法が適用可能になる。これによって、複雑な系のモデル化や理解が進むと期待されており、また実際に進んできた。さらには、様々な応用分野でもネットワークという表現方法の強みが示されている。ネットワークという表現の有用性を説明するための一例として、人の社会でなんらかの情報が広がるダイナミクスを考えよう。それは非常に複雑であり、情報が何かということにも依存するし、それぞれの人の特性、その時々の状況にも依存する。このような複雑な現象を複雑なまま扱うのは困難であるが、人をノード、人ととの関係をリンクで表現し、情報の拡散を確率的な事象として抽象化して表現してやると、ネットワーク分析の手法で解析が可能となる。このような解析によって、例えば、少數の「インフルエンサー」の存在によって情報の拡散規模が大きく影響を受けること [4] や、「コミュニティ」と呼ばれる人が密につながれたクラスターの存在によって、情報拡散がトラップされやすいこと [5]、など、情報の種類や人の特性、その時々の状況にあまり依存しない普遍的な特性が明らかとなる。このような普遍的な特性が明らかになると、ソーシャルメディアにおける広告配信など応用のドメインにおいても有用である。



(a) AS (Autonomous System) [1]

(b) PowerGrid [2]



(c) オンライン掲示板 [3]

図 1.2: 現実のネットワークを可視化した例

ネットワークは、様々な分野で利用可能な有用な表現方法である。

現実に存在する複雑な系をネットワークとして表現し、解析する研究は、様々な分野で取組まれてきたが、それらが融合する形で「ネットワーク科学」と呼ばれる学際的研究分野が形成された。なお、ノードとリンクで表現されるネットワークは、「グラフ理論」の分野でも古くから研究対象となっていた。ネットワーク科学の分野はグラフ理論の道具を使いつつ、現実の何らかの対象を扱うという点に特徴がある。ネットワーク科学分野の歴史的な経緯をごく簡単にであるが、振り返って説明する。古くから、現実のネットワークを対象とする研究は行われていたが、今日のネットワーク科学という分野が形成されるに至ったきっかけとしては、Watts と Strogatz のスモールワールドネットワークの研究 [2] と、Barabási らのスケールフリーネットワーク<sup>\*1</sup>の研究 [7] の影響が大きいであろう。これらの研究は、現実の様々なネットワークに共通して見られる性質を明らかにし、その性質を再現する数理モデルを提案した。これらのモデルは今日においてもよく使われている。また時をほぼ同じくして、初期の Google の Web 検索エンジンを構成するコア技術として PageRank [8] が登場した。このあたりから、計算機科学の応用分野でもネットワークの重要性が認識され始め、情報検索や情報推薦などの分野で研究が活発化した。以降、ネットワークの数理的、理論的側面は主に物理（統計物理）分野を中心とし、計算量やサービスへの応用の側面では計算機科学分野を中心とし、ネットワークとして表現可能な分析対象を持つ分野の研究者も参加しつつ、発展してきた。

本稿では、学際的な研究分野であるネットワーク科学分野で研究されてきたネットワーク分析の手法を解説する。先に歴史的経緯を説明したように、ネットワーク科学という分野自身が、様々な分野が融合する形で形成されている。そのため、重要な成果が様々な分野に分散して存在している。本稿では、様々な分野にまたがる重要な成果の中でも、応用の効きやすい基本的なものに限定し、ネットワーク分析を実際に活用できるようになることを目指す。

## 1.2 ネットワーク分析の活用例

現実の複雑な対象をネットワークとして表現し、解析することでどのようなことが可能となるのであろうか？以下ではいくつかの典型的な例を紹介する。

### 1.2.1 ネットワークの頑健性の解析

通信網や交通網、電力網などはノード同士がつながっていてこそ、ネットワーク全体での機能を発揮する。もしいくつかのノードやリンクが故障してしまったり、攻撃を受けてしまったりすると、ネットワーク全体の接続性はどの程度維持されるであろうか？このような問い合わせる際に、ネットワーク分析の手法が有用である [9]。このトピックについては、本稿の 6 章でも扱う。

<sup>\*1</sup> スケールフリーという用語には色々と議論がある [6] ので、以降はあまり不用意にこの言葉は使わない

### 1.2.2 ウィルス拡散の特性解析

今世界中で流行している COVID-19 などを介して感染の広がるウィルスの拡散特性を調べるのも、ネットワーク分析が有用である [10]。人と人との接触パターンによって、ウィルスの拡散速度や規模は大きな影響を受ける。人と人との接触パターンはネットワークとして表現することができる。ネットワーク上でのウィルス拡散のシミュレーションや数値解析は、ウィルス拡散の対策を考えるのも有用であると期待されている。このトピックについては、本稿の 7 章でも扱う。

### 1.2.3 テロ対策

テロ組織のネットワーク解析もネットワーク分析の応用例としてよく取り上げられる [11]<sup>\*2</sup>。テロ組織の構成員の目撃情報などから、組織のネットワーク構造を推定することで、組織内の隠れた重要人物を炙り出すような応用が期待されている。

### 1.2.4 脳機能の解析

脳の領域をノード、脳活動の相関をリンクとして脳をネットワークとして表現することができる [12]。ネットワークとして表現することで、似た機能を持つ脳領域のモジュール（本稿の言葉ではコミュニティ）を特定したり、ある運動に対して重要な役割を果たす脳領域を推定するなどの解析が行われている。

### 1.2.5 情報検索

コンピュータサイエンス分野でのネットワーク分析の活用例として最も代表的なのは、Web 検索を代表とする情報検索であろう。Google の初期の Web 検索で重要な役割を果たした PageRank [8] は、Web のネットワークを解析することで Web ページの重要度を推定する指標である。その後も、大量の文書の中から重要なものを特定する情報検索の分野ではネットワーク分析が活用されている。

### 1.2.6 情報推薦

情報検索と似ているが少し異なるタスクである情報推薦にもネットワーク分析が活用されている。情報推薦とは、オンラインショッピングサイトで商品を推薦したり、VOD サービスで映画を推薦したりする技術のことである。このような情報推薦を実現するためには、ユーザや推薦対象の商品などの関係をネットワークとして表現し、ネットワークに対する機械学習の枠組みであるグラフニューラルネットワークを組み合わせることで良い推

<sup>\*2</sup> アメリカの先生はけっこう counter terrorism を例に挙げることが多い気がするし、実際そういう予算で研究がされているようだ。日本人的にはちょっとこの例はピンとこないかもしれない。ヤクザのネットワーク分析とかやっている人はいるのだろうか？

薦が実現できるという結果が、ここ最近数多く報告されている [13]。

### 1.2.7 バイラルマーケティング

ソーシャルメディア上での口コミを利用して商品やサービスなどの情報を拡散するバイラルマーケティングと呼ばれる手法が注目されている。バイラルマーケティングにおいて情報を拡散してもらう影響力の強いインフルエンサーを特定するのにも、ネットワーク分析が活用できると期待されている [14, 15]。

このようにネットワーク分析は、通信はもちろん、上位レイヤのサービスや、人の行動や脳の解析など幅広い応用ドメインを持っている。

## 1.3 本稿のねらいと構成

本稿は、ネットワーク分析の初学者を対象とし、基本的な概念から実践的なネットワーク分析の活用例までを解説することを目指している。特にネットワーク分析の実例を多く紹介し、すぐに分析に取りかかることができるよう分析のためのコードも紹介する。

なお、ネットワーク科学の分野では、ネットワークの構造上の特徴やネットワーク上のダイナミクスを数理的な手法で解析するアプローチも様々行われているが、本稿は計算機を用いて実データを分析するための手法に特にフォーカスする。ネットワーク科学の理論的な側面の入門書としては、バラバシ本（原著 [16]、邦訳 [17]）や増田・今野本 [18]などを参照されたい。

本稿の構成は以下の通りである。まず 2 章において、ネットワーク表現の用語および基本的な事項について説明する。3 章では、ネットワークの特徴を定量化する基本的な指標を紹介する。4 章では、ネットワークから重要なノードを抽出する中心性指標を、5 章では、ネットワークからノードのグループを抽出するコミュニティ抽出手法をそれぞれ解説する。6 ~ 9 章ではネットワーク分析の実践的な例を紹介する。ネットワークの故障や攻撃に対するロバスト性の評価手法、ネットワーク上でのウィルス拡散特性の分析手法、Twitter のエゴネットワークの解析手法、研究者ネットワークの解析手法について、実践的に紹介する。最後に 10 章において本稿のまとめを述べる。

## 1.4 ネットワーク分析のためのツール

ネットワーク分析のための便利なツールやパッケージが登場し、ネットワーク分析に取り組むための技術的なハードルは低下している。本稿では、特に Python と Python においてネットワーク分析をするための標準的なライブラリの一つである NetworkX を用いた例を紹介する。なお、本稿で用いたプログラムのソースコードは GitHub <sup>\*3</sup> 上でも公開している。

<sup>\*3</sup> [https://github.com/s-tugawa/MIKA21\\_tutorial](https://github.com/s-tugawa/MIKA21_tutorial)

以下に、Python から利用可能なネットワーク分析のための主要なモジュールを示す。

NetworkX <sup>\*4</sup> 基本的なネットワーク分析手法はだいたい実装されており、使い方も簡単である。大きなネットワークを扱う時には実行速度の面でややつらいというのが欠点ではある。本稿ではこれを用いる。

graph-tool[19] <sup>\*5</sup>Tiago P. Peixoto 氏の開発するモジュール。中身は C++ の Boost Graph Library を使っており高速なのが特徴である。NetworkX と比べると後発であるが、現在ではかなり機能も充実している。Boost に依存しているということもあり、ノード番号は 0 始まりの連続する整数でなければならないなど<sup>\*6</sup>、少し融通の効かないところもある。

python-igraph <sup>\*7</sup> igraph は、R 向けのインターフェースがかなり充実しており、Python が流行る前は、ネットワーク分析と言えば、R で igraph という感じであった。その Python 向けのインターフェースが python-igraph である。Python 版の方はドキュメントの整備があまり進んでおらず、やや取っつきにくい印象である。筆者は R の igraph は今でも使うが、python-igraph は使ったことがなく、実はよく分かっていない。

また、ネットワークの可視化用のソフトウェアとして以下のものが存在する。

Gephi [20] GUI ベースで、操作方法も直感的であり、可視化以外にも、ネットワークの特徴量を計算する機能もついている。Python 等のプログラミングがあまり得意でなければ、まず Gephi を触るところからネットワーク分析を始めてみるのもよいかもしれない。

Cytoscape [21] 生物系の分野で開発された可視化に特化したソフトウェアである。Gephi よりもかっこいい感じの可視化ができる印象である。

本稿の一部では、ネットワークの可視化に Gephi を用いている。ただし、Gephi 自体の使用方法の解説は本稿では割愛する。GUI ベースのツールであり、使用法は比較的容易に修得可能であると考えられる。なお、上に挙げた NetworkX などのライブラリにも、可視化の機能は備わっているが、それほど強力なものではない。

本稿で紹介するプログラムは、Python 3.8.3、NetworkX 2.5 のバージョンで動作確認している。Python3 系、NetworkX のバージョン 2 系であれば、細かなバージョンの違いは気にせず動作するものと思われる。python3 および pip3 がインストールされている環境であれば、以下のように NetworkX をインストールすることができる。

<sup>\*4</sup> <https://networkx.github.io/>

<sup>\*5</sup> <https://graph-tool.skewed.de/>

<sup>\*6</sup> もし古い知識だったらすみません

<sup>\*7</sup> <http://igraph.org/>

```
$ pip3 install networkx
```

Python 環境のセットアップ方法については本稿では割愛するが、Web でも書籍でも豊富な情報が存在する。

# 2

## ネットワークの基礎

本章ではネットワーク分析で用いられる用語を定義し、いくつかの基本的な概念について解説する。

### 2.1 基本的な用語

本稿では、分析対象の系を点と線で表現したものを、「ネットワーク (network)」と呼ぶ。また、ネットワークにおいて、点で表現されるものを「ノード (node)」、ノード間の関係を表す線を「リンク (link)」と呼ぶ。図 2.1 に基本的な用語を示す。ネットワークのことを「グラフ (graph)」、ノードのことを「頂点や節点 (vertex)」、リンクのことを「辺や枝、エッジ (edge)」などと呼ぶ場合もある。これらの様々な用語はしばしば混乱をまねくが、ネットワーク科学やネットワーク分析の文脈では、これらの用語は特に区別なく interchangeable に用いられる。つまり、ネットワークとグラフ、ノードと頂点、リンクと辺、はそれぞれ同じものを指すことがほとんどである。本稿ではネットワーク、ノード、リンクという用語を基本的に用いる。グラフ理論に由来する概念の説明においてグラフ、頂点、辺などの用語を用いる場合もある<sup>\*1</sup>が、ネットワーク、ノード、リンクと同じ意味と理解して間違いない。ちなみに、「ネットワーク、ノード、リンク」の組み合わせはネットワークの言葉で、「グラフ、頂点、辺」の組み合わせはグラフ理論の言葉と筆者は習ったが、実際には、「ネットワーク、ノード、エッジ」のような使い方もよくみかける。なお、意図的に「ネットワーク」と「グラフ」を使い分けて、現実に存在する対象のこと

<sup>\*1</sup> 例えば辺リスト、エッジリストという用語があるが、リンクリストという表記は一般的でないので、エッジリストと呼ぶ。

を「ネットワーク」、現実の対象を点と線で抽象化したものを「グラフ」と呼ぶこともある。たとえば人の社会における人と人との関係を「ソーシャルネットワーク」、ソーシャルネットワークをグラフとして抽象化したものを「ソーシャルグラフ」のように使い分けることもある。

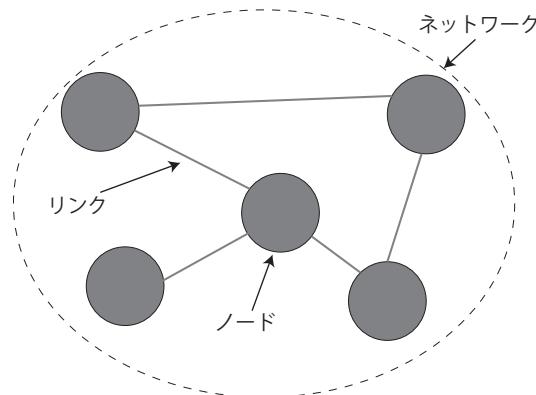


図 2.1: ネットワークの基本的な用語

数学的には、ネットワークにおけるノードの集合を  $V$ 、リンクの集合を  $E$  と表記し、ネットワークをノード集合  $V$  とリンク集合  $E$  の組として  $G = (V, E)^{*2}$  と表記する。リンク  $(u, v) \in E$  はノード  $u$  とノード  $v$  の間にリンクが存在することを表す。ネットワークにおけるノード数を  $N = |V|$ 、リンク数を  $M = |E|$  と表記する。ノード数  $N$  を、ネットワークの大きさやネットワークの規模などと呼ぶこともある。図 2.2 に示すネットワークは、 $N = 5$ 、 $M = 5$ 、であり、

$$V = \{v_1, v_2, v_3, v_4, v_5\} \quad (2.1)$$

$$E = \{(v_1, v_2), (v_1, v_3), (v_2, v_5), (v_3, v_4), (v_3, v_5)\} \quad (2.2)$$

である。

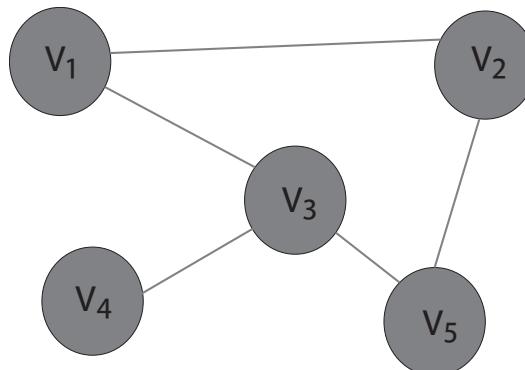


図 2.2: ネットワークの例

---

<sup>\*2</sup> graph の  $G$ 、vertex の  $V$ 、edge の  $E$  である。さっそくネットワーク、ノード、リンクではない言葉づかいが出てきたが、そういう習慣である。

$(u, v) \in E$  である時、ノード  $u$  とノード  $v$  は「隣接している」と言う。ノード  $u$  とのリンクを有するノードのことをノード  $u$  の「隣接ノード」と呼ぶ。

## 2.2 ネットワークの表現方法

本節ではネットワークの表現方法のいくつかのバリエーションを紹介する。

### 2.2.1 有向か無向か

ネットワークには、リンクの向きの有無によって、有向ネットワーク、無向ネットワークの2種類の表現方法が存在する(図2.3)。論文の共著関係や俳優の共演関係のような関係性に向きがない(関係性が対称である)ような場合には、無向ネットワークを用いる。それに対して、WWWにおけるハイパーリンク関係や、電話の発信関係、メールの送受信関係のように、非対称の関係性は有向ネットワークとして表現できる。無向ネットワークにおいては、 $(u, v) = (v, u)$  である(ノード  $u$ 、 $v$  間のリンクを  $(u, v)$ 、 $(v, u)$  のどちらで表記しても構わない)が、有向ネットワークにおいては、 $(u, v) \neq (v, u)$  である。

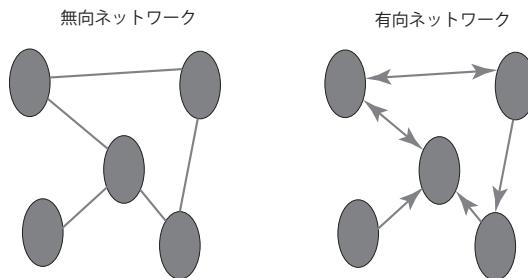


図 2.3: 有向ネットワークと無向ネットワーク

### 2.2.2 重み付きか重みなし

ノード間の関係性に「強さ」が定義できる場合には、関係性の強さをリンクの重みとして表現した重み付きネットワークを用いることができる。例えば電子メールの送受信関係や電話の発信関係を表現する場合、メールの送信回数や電話の発信回数をリンク重みとすることができる。一方で、WWWにおけるハイパーリンク関係には、関係性の強さのようなものは定義できない。そのような場合には、重みなしネットワークを用いる。図2.4に重み付きネットワークと重みなしネットワークの例を示す。

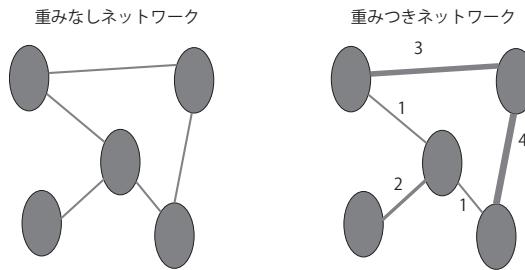


図 2.4: 重み付きネットワークと重みなしネットワーク

リンク  $(u, v)$  のリンク重みは  $w_{u,v}$  と表記する。 $w_{u,v} \geq 0$  と仮定するのが一般的である ( $w_{u,v} = 0$  はリンクが存在しないことと等価) が、リンク重みとして負の値を考えることもできる。例えば、好き/嫌いのような関係性を表現したい場合には、リンク重みとして負の値を考えることもできる。このような重みに負の値を取る重み付きネットワークのことは、符号付きネットワーク (signed network) とも呼ばれる。

### 2.2.3 分析対象と適切なネットワークの表現方法

リンクの向きの有無、およびリンク重みの有無の組合せによって、4通りのネットワークの表現方法を紹介した。分析対象のネットワークとそのネットワークの典型的な表現方法の例を表 2.1 に示す。例えば、電子メールの送受信関係は、メールの送信回数を重みとした重み付きネットワークとして表現できる。また、誰から誰にメールを送信したかという向きが自然に定義できるため、有向ネットワークである。ウェブページのハイパーリンク関係には、向きが定義できる一方、重みに相当する情報はないため、WWW のネットワークは重みなし有向ネットワークとして表現するのが自然である。このように、ネットワーク分析においては、分析対象に応じて、適切なネットワークの表現方法を選択する必要がある。

表 2.1: 分析対象のネットワークとその典型的な表現方法の例

ネットワーク	向き	重み
ルータ	無し	リンク容量
知人関係	無し	無し
メールの送受信関係	送信した人から受信した人へ	送信回数
WWW	リンクしたページからされたページへ	無し

どのようなネットワークの表現方法を用いるべきかは、分析対象によって自然と決まる場合もあるが、目的や用いる分析手法によって、その都度判断する必要がある。関係性に向きが定義できる場合であっても、双方向のリンクだけを抽出したり、単純にリンクの向きを無視するなどして、無向ネットワークとして表現することもできる。関係性に強さが

定義できる場合であっても、ある閾値以上の重みを有するリンクだけを抽出したり、単純に重みを無視することで重みなしネットワークとして表現することもできる。このようにネットワークの表現方法には、ある程度の自由度が存在する。目的に応じて、適切なネットワークの表現方法を選択することが、ネットワーク分析における最初の重要なステップである。なお、重みなしネットワークよりも重みつきネットワークの方が、無向ネットワークよりも有向ネットワークの方が情報量は多いが、必ずしも情報量の多い表現を選べばよいというわけでもない。分析手法との兼ね合いで、重み付きネットワークの重みをそのまま用いるよりも、重みの大きな重要なリンクのみを抽出した重みなしネットワークを用いる方が良い結果が得られるということも、経験的に存在する。このようなネットワークの前処理方法は、分析者それぞれのノウハウによるところが大きく、あまり体系化されていないが、ネットワークの重要な部分を抽出する backboning [22] (ネットワークの骨組を抽出するといった意味) 手法については研究が行われている。

## 2.3 隣接行列とエッジリスト

ネットワークを数理的に取り扱う上での便利な表現方法として、隣接行列が存在する。隣接行列  $A$  は  $N \times N$  の正方行列であり、ノード  $v_i$  ( $1 \leq i \leq N$ ) からノード  $v_j$  ( $1 \leq j \leq N$ ) へのリンクが存在すれば、 $A$  の  $i$  行  $j$  列の要素  $A_{ij}$  が 1、リンクが存在しなければ 0 として定義される。無向ネットワークの場合は、 $A_{ij} = A_{ji}$  であり、 $A$  は対称行列となる。また、重み付きネットワークの場合は、 $A_{ij} = w_{ij}$  となり、 $i$  行  $j$  列の要素は、ノード  $v_i$ 、 $v_j$  間のリンクの重みとして定義される。図 2.2 の例のネットワークの隣接行列は次式のように表せる。

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \quad (2.3)$$

一方で、計算機上でネットワークを表現するのに適した方法として、エッジリスト(辺リスト)が存在する。エッジリストは、リンクを有するノードのペアを列挙した以下のようない形式のものである。

### # エッジリストの例

```
1 2
2 3
3 4
```

リンク重みを表現したい場合には、ノード番号の横にリンク重みを書くこともできる。現実のネットワークは、非常に疎である(リンクのないノードペアが多い)。つまり、隣接

行列で表現した時に 0 の要素が非常に多くなる。素朴に隣接行列をメモリ上に持とうとすると、消費メモリ量は  $N \times N$  のオーダーとなるが、エッジリストであれば、リンク数  $M$  のオーダですむため、計算機上ではエッジリスト表現がよく用いられる。

## 2.4 次数

ノード  $v_i$  の有するリンクの本数  $k_i$  をノード  $v_i$  の「次数」と言う。図 2.2 の例では、 $k_1 = 2$ 、 $k_2 = 2$ 、 $k_3 = 3$ 、 $k_4 = 1$ 、 $k_5 = 2$  である。

有向ネットワークでは、ノード  $v_i$  に入ってくるリンクの本数だけを数える入次数とノード  $v_i$  から出していくリンクの本数だけを数える出次数が定義できる。有向ネットワークで特に向きについて言及せずに次数と言う場合は、出次数と入次数の和を意味すると考えられる \*3。

## 2.5 経路

ネットワーク  $G$  上で、あるノード  $v$  から別のノード  $u$  までリンクを辿って（同じノードを一度しか通らずに）到達できる時、辿ったリンクの列をノード  $v$  からノード  $u$  までの経路 (path) と呼ぶ。図 2.2 におけるノード  $v_1$  から  $v_4$  までの経路としては、 $(v_1, v_3, v_4)$  および  $(v_1, v_2, v_5, v_3, v_4)$  が存在する。

経路に登場するリンクの数を経路長と呼ぶ。ノード  $u$ 、 $v$  間の経路は複数存在する場合があるが、それらの経路のうち経路長が最も短いものを「最短経路」と呼ぶ。最短経路の長さを最短経路長と呼ぶ。ノード  $u$ 、 $v$  間の最短経路長のことを、単にノード  $u$ 、 $v$  間の「距離」と呼ぶ場合もある。図 2.2 の例でノード  $v_1$  から  $v_4$  までの経路は 2 つ存在したが、そのうち、 $(v_1, v_3, v_4)$  が最短経路であり、最短経路長は 2 である。

## 2.6 連結

無向ネットワーク  $G$  上の全てのノード間に 1 つ以上の経路が存在する時、ネットワーク  $G$  は「連結」であると言う。連結でないネットワークのことは、「非連結」であると言う。

非連結なネットワーク  $G$  は、複数の島（部分ネットワーク）で構成されるが、それぞれの島のことをネットワーク  $G$  の連結成分と言う。より正確な説明は以下の通りである。 $V' \subseteq V$  かつ  $E' \subseteq E$  の時  $G' = (V', E')$  を  $G = (V, E)$  の部分ネットワークであると言う。任意の 2 ノード間に経路の存在するような部分ネットワークのうち極大なものが連結成分である。ここで、極大な部分ネットワーク  $G'$  とは、 $G'$  にどのノードを追加しても非連結になってしまうような  $G'$  のことである。

\*3 論文等では誤解のないよう定義すべきである

## 2.7 より複雑なネットワークの表現方法

### 2.7.1 二部ネットワーク

現実のネットワークを表現する方法として、よく用いられるものの一つとして二部ネットワーク<sup>\*4</sup> (bipartite network) が存在する。図 2.5 に示すように、二部ネットワークとは、二種類のノードを持ち、同一の種類のノード間にはリンクの存在しないネットワークのことである。例えば、商品とその商品を購入したユーザの関係や、論文とその論文の著者の関係は二部ネットワークとして表現できる。このような二部ネットワークは、同一のノードへのリンクを持つノード間にリンクを生成することにより、一種類のノードのみで構成される一部ネットワークに変換することができる。このような操作を二部ネットワークの一部ネットワークへの射影 (one-mode network projection) [23] と呼ぶ。例えば、商品と商品を購入したユーザの関係を表現した二部ネットワークは、同じ商品を購入したユーザ間をリンクで結んだユーザのネットワークと、同じユーザに購入された商品をリンクで結んだ商品のネットワークに射影することができる。二部ネットワークを二部ネットワークとして解析する手法も存在するが [24]、ネットワーク分析の手法は一部ネットワークを対象としたものが多いため、二部ネットワークを解析する際には、しばしば一部ネットワークへの射影が行われる。なお、ノードの種類が二種類である二部ネットワークが頻繁に登場するため取り上げたが、ノードの種類が三種類以上の multipartite network という表現方法も存在する。

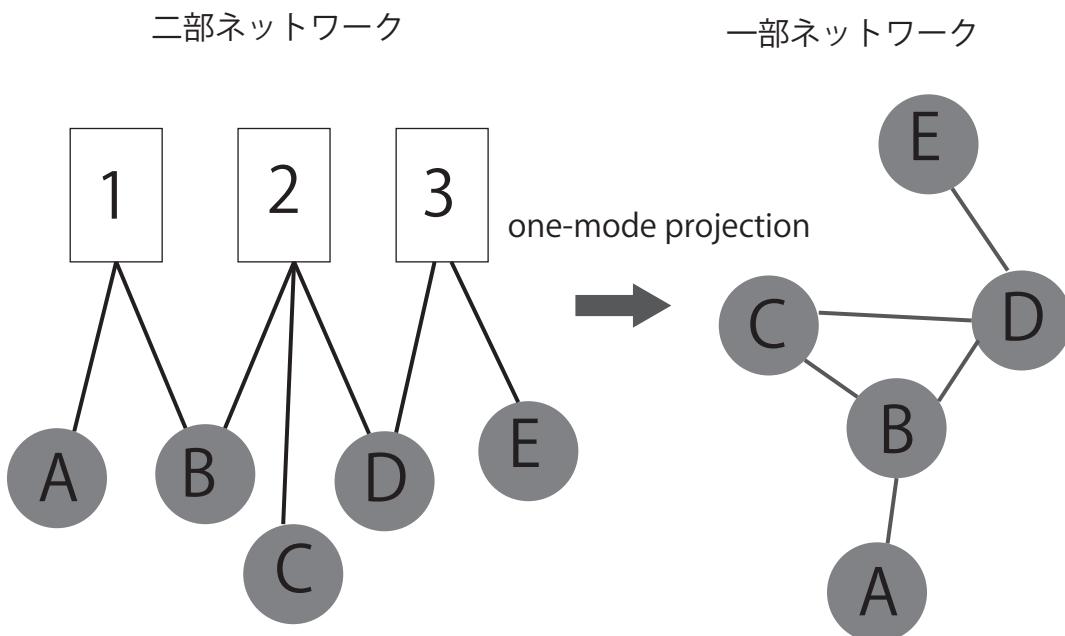


図 2.5: 二部ネットワークと一部ネットワークへの射影

<sup>\*4</sup> 日本語としては二部グラフの方が一般的かもしれないが、ここでは同じものを指す

### 2.7.2 テンポラルネットワーク

時間とともに構造が変化するネットワークの表現方法として、テンポラルネットワーク (temporal network) が存在する (図 2.6)。テンポラルネットワークでは、ノードやリンクにそのノードやリンクの存在する時刻の情報が付与される。例えば、人の間での感染症の伝播の流れを分析する場合には、誰と誰が対面したかということに加えて、「いつ」対面したかということが重要となる [25]。また人と人の間での情報拡散の流れを分析する場合にも、「いつ」情報が伝わったかということが重要となる [26]。このような場合には、対面した時刻や情報の伝わった時刻の情報を有するテンポラルネットワークとして、人と人の間の関係を表現することが有効である。テンポラルネットワークを分析するための最も単純な手法は、ある時間幅 (窓幅) ごとに、その窓幅の中に存在するノードとリンクだけで構成されるネットワークを構築し、テンポラルネットワークを時間情報のないネットワークの系列として表現することである。このようにすることで、テンポラルでないネットワークを対象とする一般的なネットワーク分析の手法を適用することができる。一方で、この方法には、明示的な時刻の情報が失われてしまったり、適切な時間幅を決めるのが難しいといったデメリットも存在する。そこで、テンポラルネットワークの時間の情報をそのまま利用する分析手法についても、活発に研究されている [27, 28, 29]。

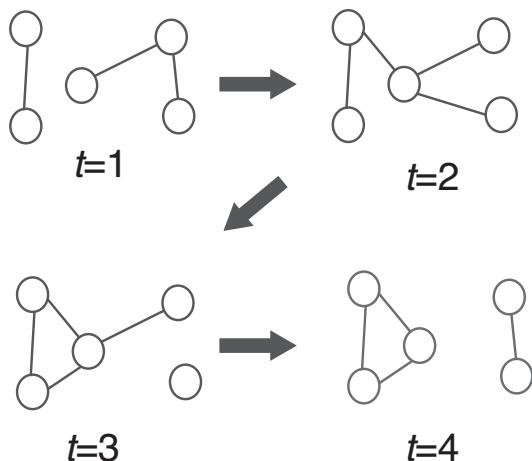


図 2.6: テンポラルネットワークの例

### 2.7.3 多層ネットワーク

複数のネットワークを複数の層 (レイヤ) とし、層間の関係を層間のリンクで表現した多層ネットワーク (multilayer network) と呼ばれるネットワークの表現方法も存在する (図 2.7)。例えば、人の社会のネットワークは、電子メールの送受信関係、電話の発信関係、対面での交流関係、など複数の種類のコミュニケーション手段それぞれを層とした多層ネットワークとして表現できる。また、電力網と通信網など相互に依存するネットワー-

クも多層ネットワークとして表現できる<sup>\*5</sup>。複数の系にまたがる現象を対象とする場合には、このような表現が有用である。例えば、ある変電所の故障が通信設備の停電を引き起こし、通信設備の停電による通信断によって別の変電所に制御信号が届かなくなり、その変電所も停止し…といった連鎖的な故障の解析に多層ネットワーク表現が用いられている[30]。

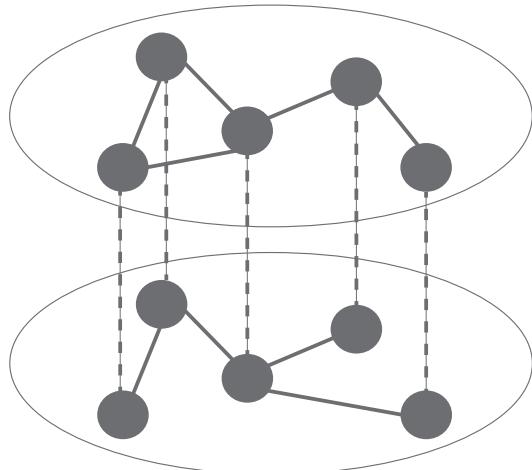


図 2.7: 多層ネットワークの例

#### 2.7.4 属性付きネットワーク

ノードやリンクに対して、そのノードやリンクの属性情報を付与した属性付きネットワーク(attributed network)という表現方法も存在する。通常、ノードやリンクの属性は、ノードやリンクに対応した特徴ベクトルとして表現する。重みつきネットワークは、リンクに対して重みという一次元の特徴だけを付与した属性付きネットワークと捉えることもできる。情報検索や情報推薦などで用いられる知識グラフ(knowledge graph)は、属性付きネットワークの代表的な例である[31]。また、オンラインショッピングにおける商品と商品を購入したユーザ間の関係と、それぞれの商品やユーザの特徴を属性付きネットワークとして表現し、商品の推薦に用いるような試みも行われている[13]。computer science の応用的なドメインでは、属性付きネットワークも頻繁に登場する。

### 2.8 本稿の扱うネットワーク

本稿では、特に明示的に断わらない限り、分析対象のネットワーク  $G = (V, E)$  は重みなし無向ネットワークであるとする。重みなし無向ネットワークが最も単純なネットワークの表現方法であり、説明を簡単化するのが目的である。ただし、ほとんどのネットワー

<sup>\*5</sup> 前者の人のネットワークの例は multiplex network、後者の電力網と通信網の例は interdependent network、など多層ネットワークの中にもさらに分類がある

ク分析手法は、自然と有向ネットワークにも、重み付きネットワークにも拡張可能である。またネットワーク  $G$  は連結であり、多重リンクおよび自己ループ（図 2.8）を含まない単純ネットワーク（単純グラフ）であることを仮定する。

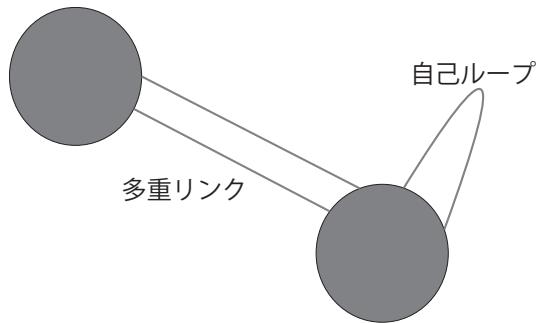


図 2.8: 自己ループと多重リンクの例

2.7 節で紹介したような、複雑なネットワークの表現方法に対する分析手法は現在も研究途上であり、本稿の扱う範囲外である。興味を持たれた読者の方は、サーベイ論文等を参照されたい。テンポラルネットワークについては文献 [27, 28, 29]、多層ネットワークについては文献 [32, 33] などが詳しい。属性付きネットワークについては、あまり分野として体系化されていない印象であるが、例えば文献 [34] などが参考になると思われる。

本章の最後に、本稿で用いる記号の定義をまとめたものを表 2.2 に示す。

表 2.2: 本稿で用いる記号

記号	説明
$G = (V, E)$	ネットワーク
$V$	ノードの集合
$E$	リンクの集合
$v_i \in V$	ネットワーク $G$ に属するノード
$(v_i, v_j) \in E$	ノード $v_i$ と $v_j$ の間のリンク
$k_i$	ノード $v_i$ の次数
$N$	ネットワーク $G$ のノード数
$M$	ネットワーク $G$ のリンク数
$\langle k \rangle$	ネットワーク $G$ の平均次数

**コラム：自分の問題に適切なネットワーク表現は？**

研究のトレンドとしては、temporal, multilayer, heterogeneous, attributed...といった複雑なネットワーク表現の研究が活発である。こういった複雑なネットワークの解析手法は、実応用ドメインでも、例えばアリババのレコメンデーションでの活用が報告されるなど [13]、成果を挙げている。複雑なネットワークの解析手法を研究することも重要である一方、なんでもかんでも複雑な表現方法を使えばいいわけではないとも思う。ネットワークの良いところは、「良い具合の抽象化」にあると考えている。自分の扱う Question や設定した問題に対して、「良い具合」の抽象化のレベルを見極めて、可能な限り抽象化してしまう（簡単なネットワーク表現を使う）のが、ネットワーク科学の研究の最初の難しいところであり、おもしろいところでもあると思っている。最新のテクニカルなトレンドについていけないおじさんの言い訳にならないよう、注意しつつ…

# 3

## ネットワークの特徴量

本章では、与えられたネットワークの構造的な特徴を定量化するための指標のうち、いくつかの基本的かつ代表的なものを紹介する。

### 3.1 特徴量を計算する前に：ネットワークの可視化

ネットワークの構造的な特徴を把握するための方法として、ネットワークの可視化が有用である。図 3.1 にいくつかのネットワークを可視化した例を示す。

1 つ目のネットワークは、Facebook における友人関係のネットワーク<sup>\*1</sup> [35] である。いくつかのノードのグループ（クラスタ、コミュニティ）が存在することが分かる。5 章でも詳しく紹介するが、「コミュニティ構造」と呼ばれる、現実のネットワークでよく見られる特徴である。2 つ目のネットワークはタンパク質間の相互作用の関係<sup>\*2</sup> [36] を表している。中心に密に接続されたノード群（core）と、周辺に次数の小さなノード（periphery）が配置されている。これも、core-periphery 構造と呼ばれる、現実のネットワークでよく見られる特徴である。このように、人が目で見れば、ネットワークの特徴に関する多くの示唆を得ることができる。3 つ目のネットワークは、論文の共著関係のネットワーク<sup>\*3</sup> [14, 37] の一部を拡大して表示したものである。このネットワークは、約 3 万 7 千ノード、約 23 万リンクを有している。このような大規模ネットワークは、ノードやリンクの数が多くて、目で関係を理解するのが難しい。このような目で見て特徴を把握することが難しいネットワークに対しては、本章でこの後に紹介するようなネットワークの特徴量

<sup>\*1</sup> <http://snap.stanford.edu/data/ego-Facebook.html>

<sup>\*2</sup> <http://networksciencebook.com/translations/en/resources/data.html>

<sup>\*3</sup> <http://research.microsoft.com/en-us/people/weic/graphdata.zip>

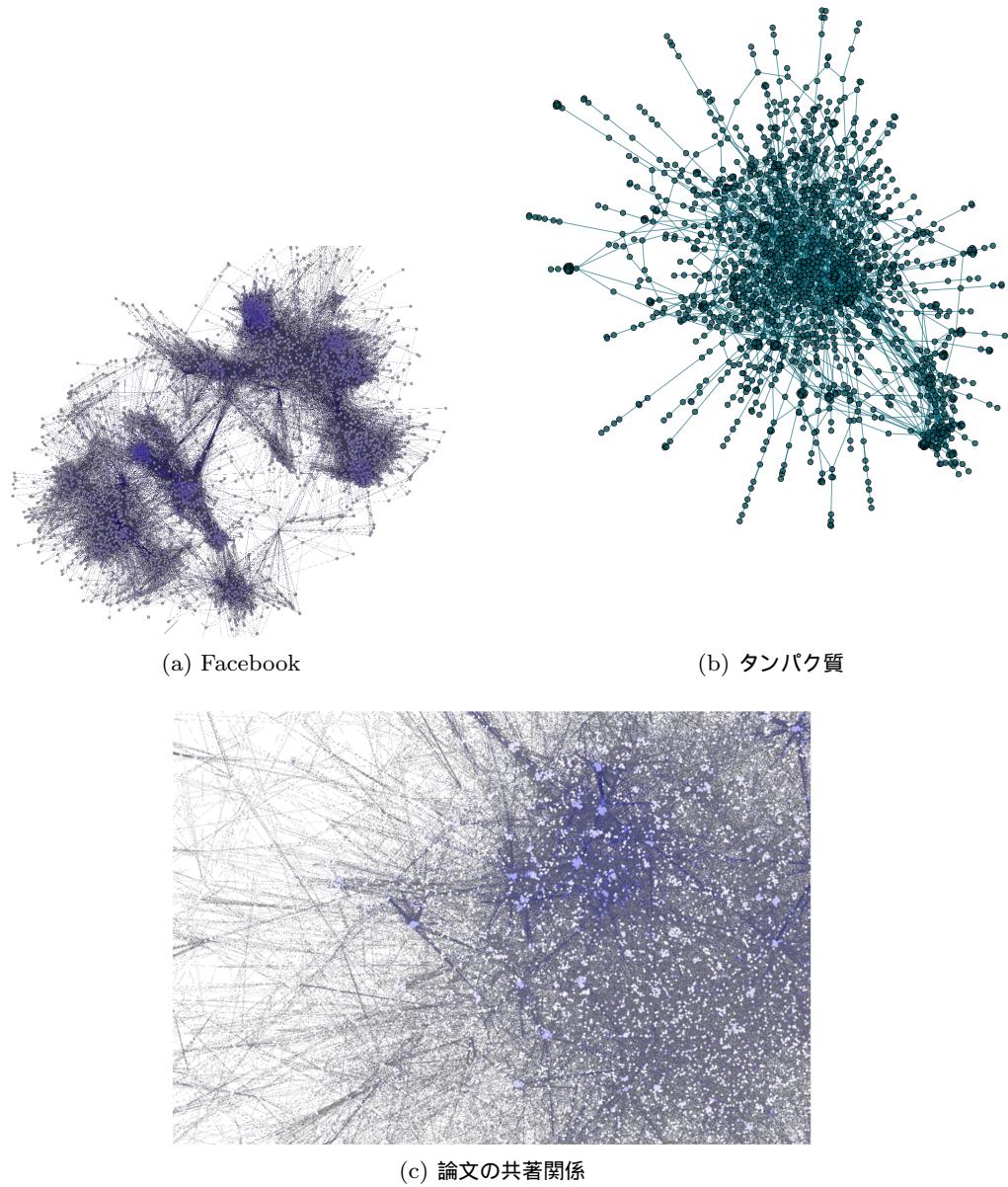


図 3.1: ネットワークを可視化した例

を定量化する指標を計算することが有効である。

整理すると、ネットワークが小規模で目で見られる範囲であれば、特徴量を計算する前に、まず可視化して目で見てみることでそのネットワークの構造的特徴に関する多くの示唆を得ることができる。経験上、リンク数にもよるが、ノード数 5,000 くらいまでは可視化して把握することができるが、10,000 を超えるとかなり難しくなる。大規模なネットワークを扱う時や、対象とするネットワークの数が多い時、複数のネットワークを定量的

に比較したい時、などには、本章でこの後に紹介する指標が有効である。

## 3.2 次数分布、平均次数

2章では、ノードを特徴付ける基本的な概念として、次数を紹介した。次数は、ノードの有するリンクの数で定義されるが、この指標はネットワークにおけるノードのある種の「重要度」と捉えることもできる。例えば人のネットワークにおいて、次数の高いノードというのは、豊富な人脈を持つインフルエンサーと考えることができる。またインターネットにおいて次数の高いノードというのは、多くのルータと接続され、大量のトラヒックを中継する基幹的なルータであると考えられる。

次数は、ノードの特徴を表す指標であったが、これをネットワーク全体の構造の特徴付けに用いる際に「次数分布」がよく用いられる。名前の通り、次数分布とは、ネットワーク内のノードの次数の分布である。ネットワークにおける次数  $k$  のノードが全ノードに占める割合（次数  $k$  のノード数を、全ノード数  $N$  で割った値）を  $p(k)$  と表記すると

$$\{p(k)\} \equiv \{p(0), p(1), p(2), \dots, p(N-1)\} \quad (3.1)$$

を次数分布と呼ぶ。次数の最大値が  $N - 1$  なのは、自己ループと多重リンクが存在しないという前提のためである。

ここで、

$$0 \leq p(k) \leq 1, \sum_{k=0}^{N-1} p(k) = 1 \quad (3.2)$$

を満たしている。したがって、 $p(k)$  を確率分布であると解釈して、次数  $k$  を有するノードが存在する確率が  $p(k)$  である、と考えてもよい。本稿では詳しく扱わないが、ネットワークを数理的に取り扱う際や、ネットワークの生成モデルを考える際には、 $p(k)$  を確率分布と解釈しておく方が都合がよい。

さて、ここで図 3.1 に示したネットワークの次数分布を見てみよう。図 3.2 の上側のグラフは次数分布を線形スケールでプロットしたもの、下側のグラフは両軸を対数スケールでプロットしたものである。線形スケールでプロットしたものは点が下にはりついていてよく分からぬが、対数スケールでプロットしたものは、点がおおよそ直線上に乗っていることがわかる。これは、大多数のノードの次数が小さい一方で、非常に大きな次数を持つノードも少数ながら存在することを表している。現実のネットワークの多くは、このような偏った次数分布を有することが知られている [16]。次数の非常に大きなノードは、「ハブノード」とも呼ばれ、ハブノードの存在は、ネットワーク上での情報拡散などのダイナミクスに大きな影響を与える。詳しくは、本稿の 7 章でも扱う。次数分布の形状は、そのネットワーク上でのダイナミクスについての情報を有していることから、ネットワーク分析において次数分布をプロットすることは、ある種のお作法的によく行われる。

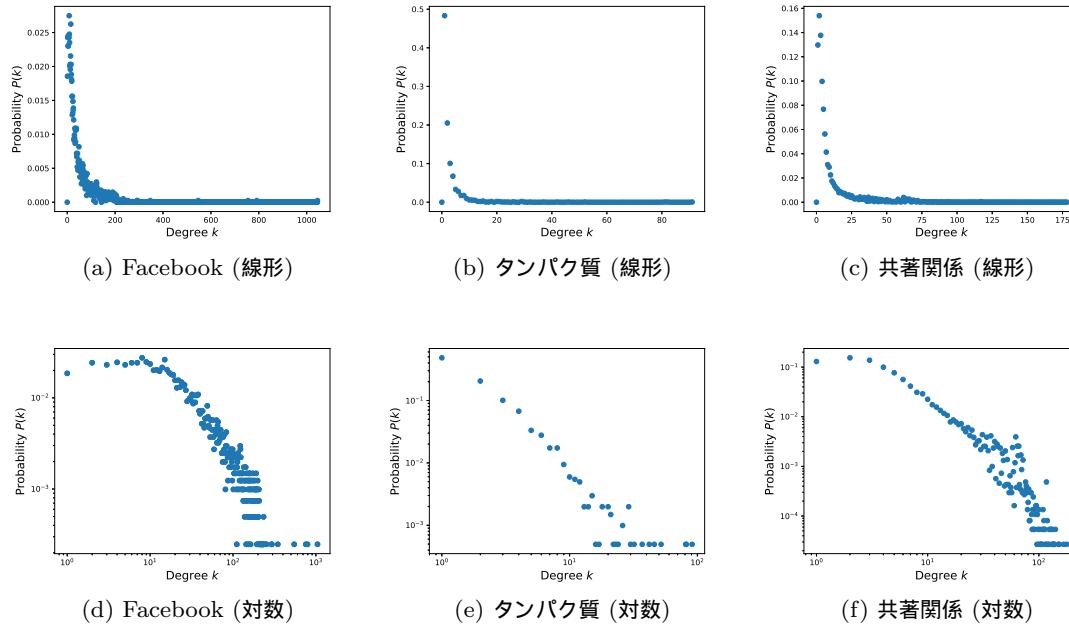


図 3.2: 次数分布の例

図 3.2 のような対数軸で直線上に乗る分布は、べき分布と呼ばれる。数式で書くと  $p(k) \propto k^{-\gamma}$  となるような分布である。次数  $k$  を持つノードの存在確率が  $k^\gamma$  に反比例するという意味である。 $\gamma$  が分布の形状を決定するパラメータであり、べき指数と呼ばれる。 $\gamma$  が小さいほど、大きな次数を持つノードの出現確率が高くなる。べき分布は、自然現象や社会現象などネットワーク以外にも出現することが知られ、古くから研究されている。ジップの法則やパレートの法則といった名前でも知られている<sup>\*4</sup>。

次数分布の要約統計量である平均次数もネットワーク分析で用いられる。名前の通り、次数の平均であり、平均次数  $\langle k \rangle$  は次式で定義される。

$$\langle k \rangle = \frac{1}{N} \sum_{i=1}^N k_i = \frac{2M}{N} \quad (3.3)$$

さきほど定義した  $p(k)$  を用いて

$$\langle k \rangle = \sum_{k=1}^{N-1} kp(k) \quad (3.4)$$

とも書ける。平均次数は、社会ネットワークであれば、社会における人は平均何人友達がいるか、インターネットであれば、ルータが他に平均何台のルータと接続されているかを表す。平均次数は、ネットワーク内にどの程度リンクが存在するかをネットワークの規模に依存せず比較する際に便利な指標である。

<sup>\*4</sup> 組織の 8 割の成果を 2 割の人が生み出しているといった話のこと

ちなみに、平均と同様に次数の分散や標準偏差などを定義することもできる。ただし、上でも紹介したように現実のネットワークの次数分布はべき分布という正規分布とは大きく異なる分布であり、あまり分散や標準偏差を使ってネットワークの特徴を議論することは多くない。

平均次数と似たような指標で、ネットワーク内のリンクの多さを表す指標として、密度も存在する。密度は次式のように定義される。

$$\frac{2M}{N(N-1)} \quad (3.5)$$

$N$  ノードのネットワークの取り得るリンク数の最大値は  $N(N-1)/2$  である。 $N$  ノード全員が次数  $N-1$  を有する場合にリンク数が最大になる（無向ネットワークなので、リンク  $(u, v)$  と  $(v, u)$  は区別しないため 2 で割る）。つまり、密度は、ネットワークの取り得るリンクのうち、実際に存在するリンクの割合を表している。平均次数の式と比べると分かる通り、平均次数と密度の違いは、 $N$  で正規化するか、 $N(N-1)$  で正規化するかだけであり、どちらもネットワーク内のリンクの多さを表している。現実のネットワークでは、ノード数が大きくなっても、平均次数はそれほど大きくならないため、ノード数が大きいほど、密度は小さい値を取る傾向にある。例えば、日本のある市（例えば人口 10 万人くらい）の住民の知人関係のネットワークと、日本全体（人口 1 億人くらい）の知人関係のネットワークで、一人当たりの知り合いの数  $\langle k \rangle$  はそれほど大きく変わらないのに対して、ノード数  $N$  は大きく異なるため、密度は日本全体のネットワークの方が小さくなる。このような規模の大きく異なるネットワーク間の密度の比較には注意が必要である。

### 3.3 平均最短経路長

2 章では、ノード間の距離（最短経路長）を定義したが、これをネットワーク全体で平均したものが平均最短経路長（平均距離）である。ノード  $v_i$  と  $v_j$  の間の最短経路長を  $d(v_i, v_j)$  と定義すると、平均最短経路長  $L$  は次式で定義される。

$$L = \frac{2}{N(N-1)} \sum_{1 \leq i < j \leq N} d(v_i, v_j) \quad (3.6)$$

$N(N-1)/2$  通りのノードペア全てについて、最短経路長の平均を求めている。

平均距離は、ネットワーク内でノード間をどの程度効率的に移動できるかを表している。平均距離が短いということは、たとえば通信網であれば、短い時間でコンテンツを配信できる、交通網であれば、少ない乗り換えで 2 地点間を移動できる、ソーシャルネットワークであれば、知り合いの知り合いをたどって多くの人に到達できるといった解釈ができる。

現実のネットワークは、平均距離が（ノード数のわりに）短いことが知られている。「スマートワールド」（世界は狭い）という言葉や、ミルグラムの実験 [38] で知られ「6 次の隔たり」という言葉が有名である。6 次の隔たりとは、友達の友達をたどると世界中の人があ

6 ホップでつながる（リンクを 6 回辿れば到達できる）ことを示唆したミルグラムの実験結果に由来している。ミルグラムの実験の 6 という数字については、色々と議論があったが、現実のネットワークの平均距離が短いという観察自体は正しかったことが、後の色々な研究で明らかになっている [39, 40]。

次数と同様に考えると、最短経路長に関しても、分布や平均以外の要約統計量を考えることも可能であるが、最短経路長に関して平均以外に頻繁に用いられる統計量は「最大値」である。最短経路長の最大値のことを、ネットワークの直径と呼ぶ。ネットワーク内で一番遠いノード同士が何ホップで接続されるかを意味する。現実のネットワークは、平均距離だけでなく、直径も短いことが知られている。

### 3.4 クラスタリング係数

「自分の知り合いの知り合いが自分の知り合いである」あるいは「自分の友人 2 人も友人同士である」といったことが、人の社会ではよくある。このような事象をネットワークで考えると、ネットワーク内に三角形（図 3.3）が多く存在するということである。このような三角形を形成するノードの集合をクラスタと呼び、ネットワークのクラスタ度合い（三角形の多さ）を定量化する指標に、クラスタリング係数<sup>\*5</sup>（clustering coefficient）が存在する。なお、人のネットワークの例で説明したが、人以外のネットワークでも三角形の構造（クラスター構造）が多く現れることは知られている。

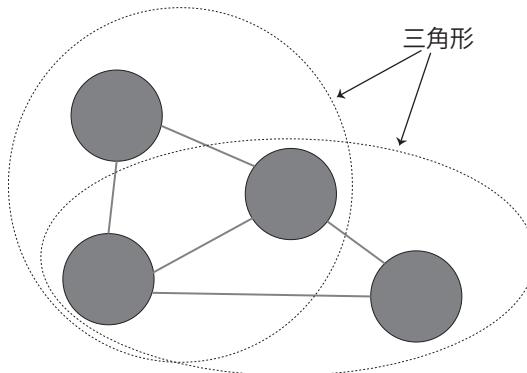


図 3.3: 三角形の例

クラスタリング係数、あるいはそれに類似した概念の定義には、いくつかのバリエーションが存在するが、ここでは最も有名と思われる Watts らのクラスタリング係数の定義 [2] を紹介する。ネットワークのクラスタリング係数  $C$  を定義する前に、ノード  $v_i$  のクラスタリング係数  $C_{v_i}$  を定義する。 $v_i$  の隣接ノードの集合を  $\Gamma(v_i)$  と表記すると、ク

<sup>\*5</sup> 日本語ではクラスター係数という表記も存在する

ラスタリング係数は次式で定義される。

$$C_{v_i} = \frac{\sum_{j,k \in \Gamma(v_i)} A_{j,k}/2}{k_i(k_i - 1)/2} = \frac{\sum_{j,k \in \Gamma(v_i)} A_{j,k}}{k_i(k_i - 1)} \quad (3.7)$$

ここで、 $A$  は隣接行列で、ノード  $v_i$  と  $v_j$  が隣接していれば、 $A_{i,j} = 1$  となる。また、 $k_i$  はノード  $i$  の次数である。分子はノード  $v_i$  の隣接ノード間のリンクの数を表す。つまり、ノード  $v_i$  の関与する三角形の数である。それに対して、分母はノード  $v_i$  とその隣接ノードの間で形成することのできる三角形の数の最大値である。つまりノードのクラスタリング係数は、そのノードと隣接ノードで形成する三角形のうち、実際にそのネットワークに存在している三角形の割合を表している。クラスタリング係数の計算例を図 3.4 に示す。

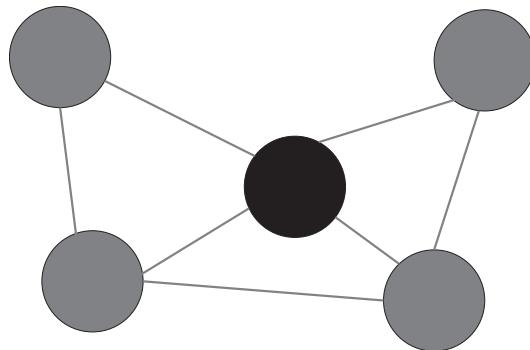


図 3.4: クラスタリング係数の計算例: 中心のノードとその隣接ノードの間で構成される三角形の数は 3 である。隣接ノードが 4 つあるので、構築し得る三角形の最大値は 6 である。したがって、中央ノードのクラスタリング係数は  $3/6$  となる。

ノードのクラスタリング係数を全ノードについて平均することで、ネットワークのクラスタリング係数  $C$  は次式のように定義される。

$$C = \frac{1}{N} \sum_{i=1}^N C_{v_i} \quad (3.8)$$

### 3.5 実ネットワークの特徴量の計算

これまでに紹介したネットワークの指標を Python と NetworkX を用いて計算する手順を紹介する。NetworkX を用いれば、次数分布、平均最短経路長、クラスタリング係数などの基本的なネットワークの特徴量を算出することができる。以降では、NetworkX の基本的な使い方を簡単に説明した後、特徴量の計算方法について解説する。

まず、NetworkX を利用するには、プログラムの冒頭で下記のように import する。

```
import networkx as nx
```

次に、分析対象のネットワークを読み込むところから分析はスタートする。エッジリスト形式でネットワークが準備してある場合は、以下のようにして読み込むことができる。

```
file=sys.argv[1]
G=nx.read_edgelist(file,nodetype=str,encoding='utf-8',
delimiter="\t")
```

コマンドライン引数で指定したファイル名のエッジリストを読み込んで G という変数に格納する。“nodetype” のオプションは、ノードを表す識別子が文字列なのか、整数値なのかといったことを指定する。ノードの識別子は 0 もしくは 1 から始まる番号であることが多いため、その場合は int でよい。“delimiter” はエッジリストの各行の区切り文字を指定するオプションである。ここではタブを指定しているが、何も指定しなければ、ホワイトスペースであることが仮定される。G がネットワークを表す NetworkX における基本的なオブジェクトであり、これに対して様々な関数を適用することで、ネットワークの指標が計算できる。ネットワークの読み込みができれば、代表的なネットワークの指標は、1 行書くだけでほとんど計算できてしまう。が、指標の計算の前に、もう少し基本的な操作の説明を続ける。ネットワークに対する典型的な操作として、全ノードに対して順になにかしらの操作をするということがある。それは、以下のように書ける。

```
for v in G.nodes():
    print(v)
```

次数の大きい順にノードを調べたければ以下のように書ける

```
deg=dict(G.degree())
for v, k in sorted(deg.items(), key=lambda x: -x[1]):
    G.remove_node(v)
```

上の例では、次数が高い順にネットワークからノードを削除している。ノードやリンクの追加は以下のように書ける

```
G.add_node(v)
G.add_node(u)
G.add_edge(u,v)
```

ノードを追加せずに、いきなりリンクだけ追加してもよい。このあたりの基本的な操作については、NetworkX のドキュメント<sup>\*6</sup>の中にサンプルが色々と掲載されているので、一度そちらを確認するとよいであろう。

<sup>\*6</sup> <https://networkx.org/>

さて、次数分布、クラスタリング係数、平均経路長などを計算する方法を紹介しよう。まず次数分布から説明する。これは1行というわけにはいかないが、以下のように書ける。

### 次数分布

```
import numpy as np
deg_seq = list(dict(G.degree()).values())
maxk = np.max(deg_seq)
ks= arange(0,maxk+1)
prob = np.zeros(maxk+1) # P(k)
for k in deg_seq:
    prob[k] = prob[k] + 1
prob = prob/sum(prob)
```

これで、prob に次数が  $k$  である確率が入る。最大を求めたりするのに numpy を使っている。これをグラフにプロットする場合は、例えば以下のようにすればよい。

### 次数分布のプロット

```
import matplotlib.pyplot as plt
plt.figure()
plt.scatter(ks,prob)
plt.xlabel("Degree $k$", fontsize=15)
plt.ylabel("Probability $P(k)$", fontsize=15)
outfile="degree_dist_"+str(file)+".eps"
plt.savefig(outfile)

plt.loglog(ks,prob,"o")
plt.xlabel("Degree $k$", fontsize=15)
plt.ylabel("Probability $P(k)$", fontsize=15)
outfile="degree_dist_log_"+str(file)+".eps"
plt.savefig(outfile)
```

プロットには、matplotlib を使っている。線形軸でプロットした場合、対数軸でプロットした場合、それぞれをファイルに保存している。

次数分布は少しややこしかったが、クラスタリング係数や平均経路長は本当に1行書くだけである。まとめて紹介すると以下のように書くことができる。

### ネットワークの特徴量の計算

```
#平均次数  
deg_seq = list(dict(G.degree()).values())  
avedeg=np.average(deg_seq)  
#ノード数  
len(G.nodes())  
#密度  
nx.density(G)  
#クラスタリング係数  
nx.average_clustering(G)  
#平均経路長  
nx.average_shortest_path_length(G)  
#直径  
nx.diameter(G)
```

さて、これらを用いて、いくつかの現実のネットワークの特徴量を計算してみよう。データは、バラバシ本 [16] の付録資料<sup>\*7</sup> から、インターネットのルータレベルトポロジ、WWW のハイパーリンク関係ネットワーク [41]、タンパク質の相互作用ネットワーク [36]、論文の共著関係ネットワーク [1] を用いる。また SNAP データセット [42]<sup>\*8</sup> から Facebook の友人関係のネットワーク [35] を用いる。図 3.5 にこれらのネットワークの次数分布を、表 3.1 にこれらのネットワークの特徴量を示す。これらの結果から、ドメインの異なるネットワークでべき分布に近い次数分布が観測されることが確認できる。また、平均距離はノード数に対してどのネットワークも短いこと、ソーシャルネットワークである共著関係や Facebook のネットワークは特にクラスタリング係数が高いことも確認できる。なお、クラスタリング係数はノード数が大きくなるほど低い値となりやすいため、WWW やルータのネットワークのクラスタリング係数もそこそこ高い値であると解釈できる。このような手法を用いることで、様々なネットワークの類似点や相違点を定量的に評価できる。

<sup>\*7</sup> <http://networksciencebook.com/translations/en/resources/data.html>

<sup>\*8</sup> <http://snap.stanford.edu/data/index.html>

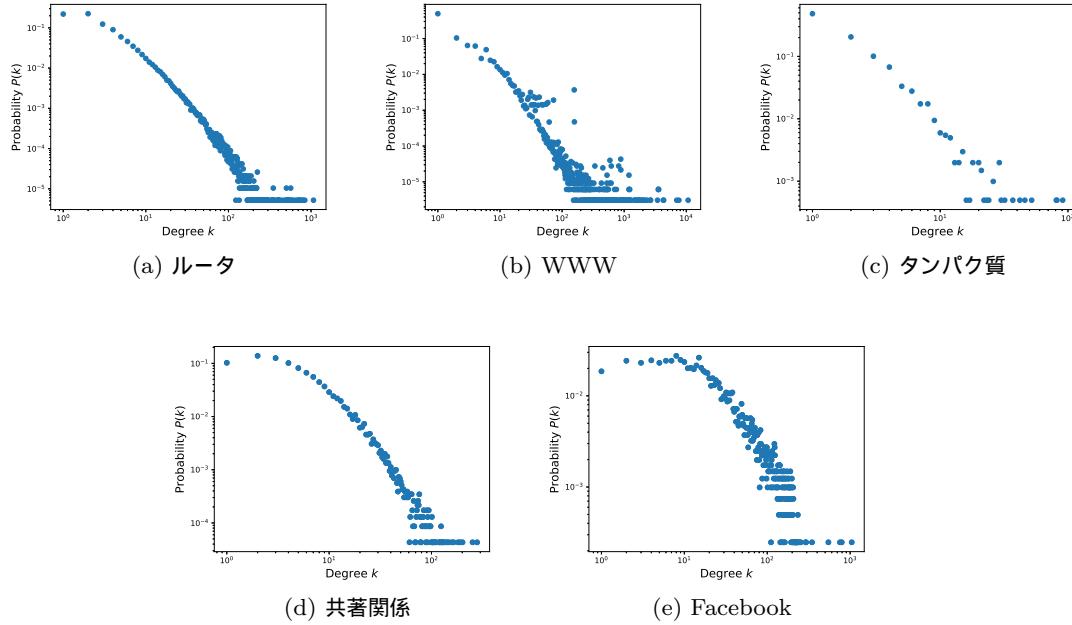


図 3.5: 様々な種類のネットワークの次数分布

表 3.1: 様々な種類のネットワークの特徴量

	ルータ	WWW	タンパク質	共著関係	Facebook
ノード数	192,244	325,728	2,018	23,133	4,039
平均次数	6.33	6.86	2.90	8.07	43.7
クラスタリング係数	0.16	0.23	0.05	0.63	0.61
連結成分の数	308	1	185	567	1
最大連結成分の割合	0.99	1.0	0.81	0.92	1.0
平均最短経路長	6.98	7.17	5.61	5.35	3.69

### コラム：昔は良かった？

ネットワーク科学の盛り上がりのきっかけとなる Watts や Barabási の論文が出たのが 1998 年～2000 年あたりである。その後、ネットワークを対象とした研究が盛り上がるわけであるが、初期の研究というのは、非常に雑に説明すると、ネットワークのデータを集めてきて、本稿で紹介するような基本的な特徴を調べ、「次数分布がべき則でした、スモールワールドでした、クラスタ性が強いです」といった感じのもののが多かった。そういった研究が一流の雑誌に掲載されていた。筆者が学生で研究を始めたのは、2007 年のことであるが、その当時、それらの論文を見て「昔はこんな簡単な分析で論文が書けていい時代だったんだな」という感想を持ったことを記憶している。なんとも、分野を切り開いた先人達に失礼な感想である。確かに、技術的に簡単な手法で論文が書けたのはその通りであろうが、当時には当時の困難さがあったはずで、「昔は良かった」という簡単な話でないことは、今では理解できる。ネットワークのデータを集めることも、そう簡単なことではないし、今となってはあたりまえで、python で一行書くだけの解析手法も、当時の人達が試行錯誤しながら洗練されていき、今日に至っているのである。これはどんな分野でもそうだと思うが、やはり研究においてどれだけ手法が簡単であろうが、「先にやった」というのはそれだけでとても重要なことであり、敬意を払うべきものであろう。

# 4

## 中心性解析

ネットワーク内での重要なノードを特定することは、ネットワーク分析における最も基本的なタスクの1つである。ネットワークにおけるノードの重要度を定量化する指標は「中心性」と呼ばれ、いくつかの定義が存在する。中心性指標は、ソーシャルネットワークから多くのノードに情報を広げることのできるインフルエンサーを特定したり、Webページのネットワークから重要なページを発見したり、通信ネットワークからそのノードの故障がネットワーク全体の接続性に大きな影響を与えるノードを特定したりするのに有用である。本章では、代表的な中心性の定義を紹介する。

### 4.1 導入

中心性という概念は、ネットワーク科学の研究が盛り上った時期よりもさらに前の1970年代には既に登場している[43]。社会科学の分野で、社会ネットワーク上の重要な人物を特定するのに、いくつかの中心性指標が提案された。その中でも特に、Freeman[43]の次数中心性、近接中心性、媒介中心性は、2021年の現在においても広く用いられている。その後、Googleの初期の検索エンジンのコア技術であるPageRank[8]が提案されたのが、1990年代後半であり、ちょうどネットワーク科学の研究が盛り上った時期と一致している。このあたりから、情報検索など社会ネットワーク以外の分野でも中心性の研究や利用が活発となった。

どのようなノードが「中心」であるか、どのようなノードが「重要」であるかは、その文脈に大きく依存し、様々な考え方がありえるため、Freemanらの中心性以降、歴史的にも、また近年においても、様々な中心性指標が提案してきた。当然ながら新しい中心性指標の中には、応用的なタスクにおいて、Freemanの中心性やPageRankより

も高い有効性を発揮するものが多く存在する。一方で、古典的な<sup>\*1</sup> Freeman の中心性や PageRank は現在も根強くネットワーク分析に利用されている。例えば、2020 年の筆者の論文でもこれらの中心性を用いている [44]<sup>\*2</sup>。本稿では、これら基本的かつ現在も利用されているいくつかの中心性指標について解説する。

## 4.2 次数中心性

次数中心性 (degree centrality) は、既に紹介したノードの次数で定義される。多くのリンクを持つノードは重要である (そのようなノードが中心である) という考えに基づく、非常に単純かつ自明な定義である。ノード  $v_i$  の次数中心性の定義は次式の通りである。

$$C_{\text{deg}}(i) = k_i \quad (4.1)$$

また、ノード数で正規化する (0~1 の値を取るようにする) と

$$\tilde{C}_{\text{deg}}(i) = \frac{k_i}{N - 1} \quad (4.2)$$

となる。

次数中心性は単純である一方で、実際にはかなり強力なことも多い。ソーシャルネットワークにおいて多くのノードに情報を広げられるようなインフルエンサーを特定するアルゴリズム [45] や、誤情報に対する訂正情報を発信するノードを決定するアルゴリズム [46] を開発する研究をする時には、次数中心性の高いノードを選ぶというのが自明なベースラインとなる。新しいアルゴリズムを提案する研究であれば、この自明なベースラインに勝たなければいけないわけであるが、なかなか勝てずに苦労することも多い<sup>\*3</sup>。

## 4.3 近接中心性

近接中心性 (closeness centrality) は、ノードからネットワーク中の他のノードへの距離 (最短経路長) の短かさに基づき定義される。他のノードへ短い距離で到達できるノードは重要であるという考えに基いている。ノード  $v_i$  の近接中心性は次式で定義される。

$$C_{\text{clo}}(i) = \frac{1}{\sum_j d_{i,j}} \quad (4.3)$$

ここで、 $d_{i,j}$  はノード  $v_i$  から  $v_j$  への最短経路長を表す。また、ノード数で正規化する (0~1 の値を取るようにする) と

$$\tilde{C}_{\text{clo}}(i) = \frac{C_{\text{clo}}(i)}{N - 1} \quad (4.4)$$

<sup>\*1</sup> という表現も少し失礼な感じもあるが

<sup>\*2</sup> 研究の目的にもよるが、「こんな古い手法ではなく最新のこれこれを使いなさい」と査読者に怒られるという雰囲気ではない

<sup>\*3</sup> あくまで筆者の個人的な経験であるが、ベースラインが強すぎて大変というのは共感の得られる経験であると信じている

となる。

次数中心性が各ノードの次数という局所的な情報のみに基づく指標であるのに対して、近接中心性は、各ノードへの距離というネットワークの大域的な情報に基づく指標である。あるノードの近接中心性を求めるには、そのノードから全ノードへの最短経路の計算が必要となり、次数中心性など局所的な情報のみに基づく指標と比較すると計算量が大きい。また、大域的な情報に基づく指標としては、近接中心性よりも、次に紹介する媒介中心性の方が広く用いられている。筆者の研究 [45] でも、近接中心性は他の中心性よりも、ソーシャルネットワークのインフルエンサー推定の問題における有効性が低いことを確認しており、個人的には、何か1つ中心性を計算しようと考えた時に、近接中心性を選ぶということはほとんどない。ただし、例えば、「ネットワークの各ノードから等確率でコンテンツへの要求がある時に、ネットワーク全体での平均コンテンツ配送遅延を最小化するコンテンツの配置場所を決めたい」のような問題を考えた時には、近接中心性が最大のノードにコンテンツを配置するというアプローチは良いであろう。やはり、あくまでどの中心性を使うべきかは問題依存である。

## 4.4 媒介中心性

媒介中心性 (betweenness centrality) は、ノードを通過する最短経路の本数に基づき定義される。直感的な説明としては、「ネットワークにおいて他のノードと他のノードの橋渡しをする役割のノード」や「ネットワークにおいてそのノードが欠落すると、ネットワークが分断されてしまうノード」が重要であるという考えに基づいている。ノード  $v_i$  の媒介中心性は次式で定義される。

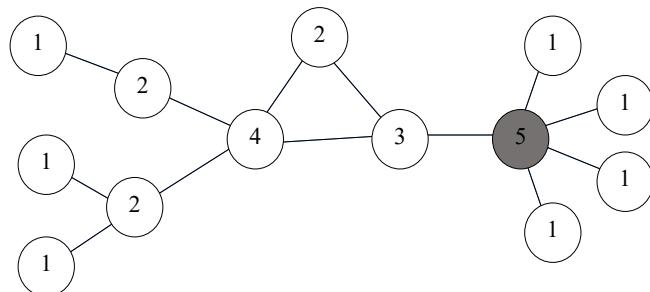
$$C_{\text{bet}}(i) = \frac{\sum_{s,t \in V, s \neq t \neq i} g_{st}(i)}{\sum_{s,t \in V, s \neq t} g_{st}} \quad (4.5)$$

ここで、 $g_{st}$  はノード  $v_s$ 、 $v_t$  間の最短経路の数、 $g_{st}(i)$  はノード  $v_s$ 、 $v_t$  間の最短経路のうちノード  $v_i$  を経由する最短経路の数である。全ノード間の最短経路を考え、その中でノード  $v_i$  を経由するものを数える。多くの最短経路が通過するノードは、そのネットワークの要であると考え、媒介中心性が高くなる。また、ノード数で正規化すると次式の通りである。

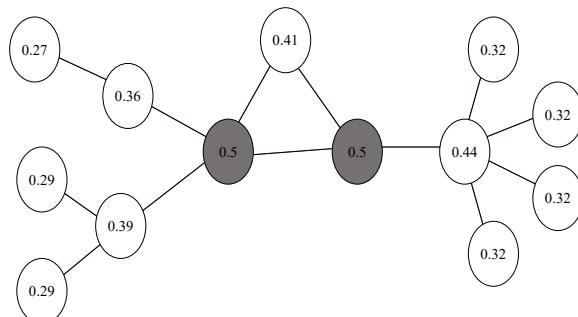
$$\tilde{C}_{\text{bet}}(i) = \frac{2}{(N-1)(N-2)} \frac{\sum_{s,t \in V, s \neq t \neq i} g_{st}(i)}{\sum_{s,t \in V, s \neq t} g_{st}} \quad (4.6)$$

ネットワーク内で全ノードペア間を均一な大きさのフローが流れていると考えた時の、通過するフローの数で媒介中心性が定義されていると解釈すると理解しやすいかもしれない。この時、フローは最短経路を通ると仮定するのがオリジナルの媒介中心性の定義である。一方、他の経路を通ると考える場合の拡張も多く存在する。詳細は文献 [47] にまとめられたものが参考になるが、メジャーなものとしては、random walk betweenness [48]、flow betweenness [49]、routing betweenness [50] などが存在する。

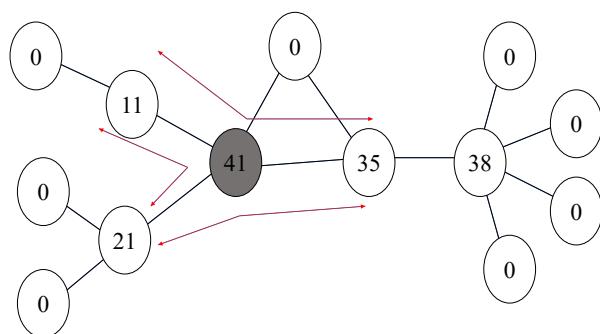
ここで、図 4.1 に次数中心性、近接中心性、媒介中心性を計算した例を示す。次数中心性は、次数さえ高ければ、ネットワークの端の方のノードでも高い値を取り得る。近接中心性や媒介中心性はネットワークの「中心」という言葉のイメージと直感的にも一致している。ただし、現実のネットワークでは中心性間の相関は強く [51]、すごく次数は低いが媒介中心性が高いといったことはあまり生じない。



(a) 次数中心性



(b) 近接中心性



(c) 媒介中心性

図 4.1: 中心性の計算例

## 4.5 固有ベクトル中心性

固有ベクトル中心性は、中心性の高いノードに隣接しているノードは中心性が高いという再帰的な関係に基づく指標である。ノード  $i$  の固有ベクトル中心性は次式で定義される。

$$C_{\text{eig}}(i) = \frac{1}{\lambda} \sum_j^N A_{ij} C_{\text{eig}}(j) \quad (4.7)$$

$\lambda$  は規格化のための定数である。ここで全ノードの固有ベクトル中心性を縦方向に並べた  $N$  次元のベクトル  $C$  を考えると、固有ベクトル中心性は以下のようにも書ける

$$C = \frac{1}{\lambda} AC \quad (4.8)$$

$$\lambda C = AC \quad (4.9)$$

この式を見ると、 $C$  は隣接行列  $A$  の固有ベクトル、 $\lambda$  が固有値の関係になっていることが確認できる。つまり、隣接行列  $A$  の固有値、固有ベクトルを計算することで、 $C_{\text{eig}}(i)$  が求まる。特に最大固有値に対応する固有ベクトルを固有ベクトル中心性と呼ぶ。

ここで固有ベクトル中心性は、非連結なネットワークや非対称な隣接行列を持つネットワーク（有向ネットワーク）に対してはうまく定義されない。固有ベクトルと同様に隣接ノードの中心性を自身の中心性に反映させるという考えに基づき、非連結なネットワークや有向ネットワークにも適用可能なように拡張したのが次に紹介する PageRank である。

## 4.6 PageRank

PageRank [8] は Google の検索システムにおける Web ページのランキングに用いられたのがその起源であるが、その後様々なネットワークのノードの重要度を測る指標（中心性指標）としても広く用いられてきた。固有ベクトル中心性と同様、PageRank の高いノードにリンクされているノードはページランクが高いという考えに基づいて、ノード  $v$  の PageRank 値は次式で定義される。

$$PR(v) = \frac{1 - \alpha}{N} + \alpha \sum_{u \in \Gamma(v)} \frac{PR(u)}{k_u^{\text{out}}} \quad (4.10)$$

ここで、 $\Gamma(v)$  はノード  $v$  に対するリンクを有するノードの集合、 $k_u^{\text{out}}$  はノード  $u$  の出次数である（有向ネットワークを前提とする定義）。 $\alpha$  はパラメータであり、慣例的には 0.85 とすることが多い。

PageRank は「ランダムサーファーモデル」という Web ページ閲覧（Web サーフィン）を行うユーザの行動パターンで解釈できる。Web ページを閲覧するユーザはページ  $u$  を訪れた後、確率  $\alpha$  でページ  $u$  のリンク先のページ  $k_u^{\text{out}}$  個の中からランダムに選んだ一つのページを訪問する。また、確率  $1 - \alpha$  で訪問中のページ  $u$  とは無関係に、Web

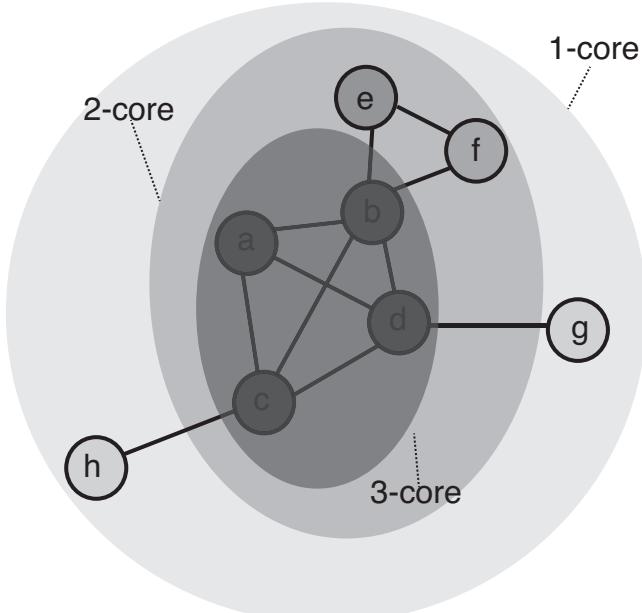


図 4.2:  $k$ -コアの例：ノード  $a$ 、 $b$ 、 $c$ 、 $d$  は、1-コア、2-コア、3-コア全てに属している。そのため、これらのノードのコアネスは 3 である。ノード  $e$ 、 $f$  は 1-コアと 2-コアに属しているので、コアネスは 2、ノード  $g$ 、 $h$  は 1-コアのみに所属しているため、コアネスは 1 である。

上のページ ( $N$  個ある) からランダムに一つ選びそのページを訪問する。このような行動を繰り返すユーザが各ページ  $v$  に滞在する確率の定常分布を表したもののが式 (4.10) の PageRank の式である。

## 4.7 コアネス ( $k$ -コア指標)

コアネス (coreness、 $k$ -core あるいは  $k$ -shell、などとも呼ばれる) [52, 53, 54, 55] は、ノードがどの程度大きなコア (密なサブネットワーク) に属しているかに基づきノードの重要度を推定する指標である。ノード  $v$  の  $k$ -core に基づくコアネスの定義にあたって、まず、ネットワーク  $G$  におけるノード  $v$  の属する  $k$ -コアを定義しよう。 $H$  を  $G$  のサブネットワークとし、 $\delta(H)$  を  $H$  に属するノードの、サブネットワーク  $H$  における次数が最小のノードの次数と定義する。つまり、サブネットワーク  $H$  に属する全てのノードは  $\delta(H)$  以上のサブネットワーク  $H$  に属するノードと隣接しているということである。 $\delta(H) \geq k$  であるとき、サブネットワーク  $H$  は、ネットワーク  $G$  の  $k$ -コアであると呼ばれる。この定義を用いて、ノード  $v$  のコアネスは、ノード  $v$  が属する  $k$ -コアのうち、最大の  $k$  の値であると定義される。図 4.2 に  $k$ -コアとコアネスの例を示す。

## 4.8 実ネットワークの中心性指標の計算

それでは、これまで学んだ中心性指標をいくつかのネットワークで計算してみよう。例によって NetworkX を使えば、中心性の計算自体は以下のように 1 行でできてしまう。

### 中心性指標の計算

```
#入次数  
indeg=nx.in_degree_centrality(g)  
  
#出次数  
outdeg=nx.out_degree_centrality(g)  
  
#媒介中心性  
bet=nx.betweenness_centrality(g)  
  
#近接中心性  
clo=nx.closeness_centrality(g)  
  
#PageRank  
pr = nx.pagerank(g)  
  
#コアネス  
kcore=nx.core_number(g)
```

まずは、ハイテク企業の管理者のネットワーク [56]<sup>\*4</sup>を用いる。このネットワークはある企業の 21 人の管理職それぞれに、「仕事上のアドバイスを求める」、「友人である」、「報告をする」という関係にある人の名前を挙げてもらい、名前を挙げた人から挙げられた人への有向のリンクを生成することで構築されている。3 種類の関係それぞれについて 1 つのネットワークが存在する（3 層の multiplex network と考えてもよい）。このネットワークを可視化した結果を図 4.3 に示す。「報告する」の関係は、部署などの構造で決まっており、階層化されていることが分かる。友人関係やアドバイスの関係はそれほどきれいな構造はない<sup>\*5</sup>。

これらの 3 種類のネットワークで、それぞれのネットワークの中心人物がどの程度一致しているかを調査してみよう。ここでは解釈が容易な入次数中心性のランキング上位 5 ノードをリストアップした。表 4.1 に入次数に基づく中心ノードを示す。ノード 2 は友達が多く、アドバイスもたくさん受けるが、報告を受ける数では 4 位である。組織構造上の部下は多くないが、色々な人に頼りにされている人物ではないかと推測される。一方、もっとも報告を受ける 14 は、アドバイスと友人のネットワークでは上位に入ってこない。3 つのネットワークで中心ノードにある程度オーバーラップがあるものの、それに特有の中心ノードの存在も確認できた。このように、中心性指標は、組織のコミュニケーション構造を分析したり、さらにはそこから課題を抽出したりするのに用いることがで

<sup>\*4</sup> <http://vlado.fmf.uni-lj.si/pub/networks/data/WaFa/default.htm>

<sup>\*5</sup> 友人関係が非対称なのがなんとも悲しい

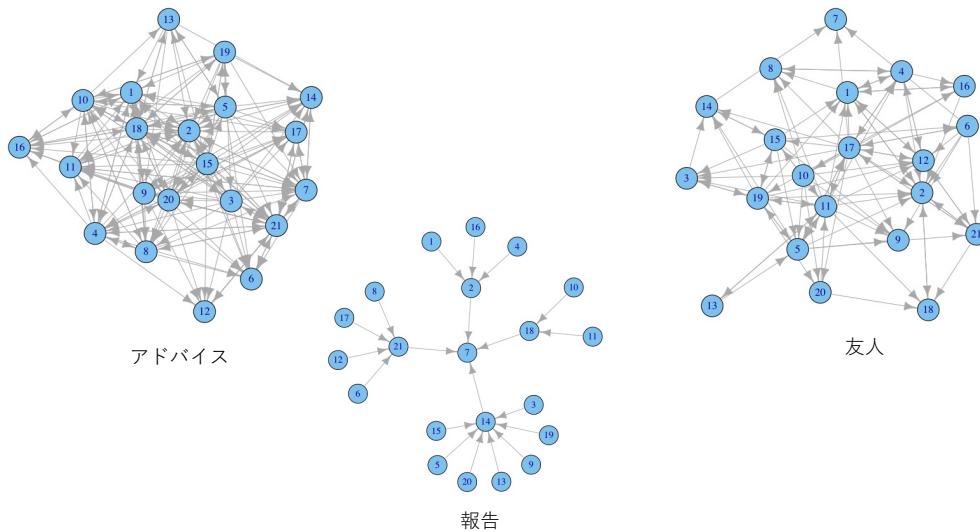


図 4.3: ハイテク企業の管理職のネットワーク

表 4.1: ハイテク企業の管理職ネットワークにおける入次数中心性の高いノード

順位	アドバイス	友人	報告
1	2	2	14
2	18	1	7
3	21	12	21
4	1	5	2
5	7	9	18

ある。

次に、もう少し大きなネットワークで中心性を計算してみよう。ここでは、物理分野の論文の共著ネットワーク<sup>\*6</sup>から、中心性指標を用いて、この分野の権威を発見する問題を検討する。データを読み込み、中心性を計算して、上位のノードを抽出すればよさそうである。ただ、物理分野の背景知識が筆者にはないため、中心性の計算結果が、どの程度物理分野の権威度合いを反映しているかが判断できない。そこで、何らかの指標で、中心性指標のランキング結果がどのくらい実際の権威度と相関があるかを測ることを考える。研究者の評価をどのようにすべきかは、それ自体が研究となり得るわけであるが、ここでは簡単な指標として、研究者の書いた論文の被引用数を研究者の権威度と考えよう。同一分野の中で比較するのであれば、たくさん引用されている人がすごいというのは、それほど違和感なく受け入れられるのではないかと思う。ということで、ここでは、共著関係ネット

<sup>\*6</sup> <http://www-levich.enr.ccny.cuny.edu/~hmakse/SOCIAL/APSdata.rar>

トワークから計算した中心性と、論文の被引用数の相関を調査してみる。共著関係ネットワークと被引用数のデータは整形済みのものを GitHub 上に置いてあるので、それを利用する。

図 4.4 に次数中心性、PageRank、コアネス ( $k$ -コア)、被引用数の 4 つの指標の間の散布図行列を示す。この結果から、次数中心性、PageRank、コアネスとともに被引用数とある程度の相関が存在することが確認できる。相関係数は次数中心性、PageRank、コアネスの順に、0.45、0.32、0.69 であった。このことから、研究者ネットワークにおける中心性指標は、その分野の権威となる研究者を特定するのに有用であることが期待される。

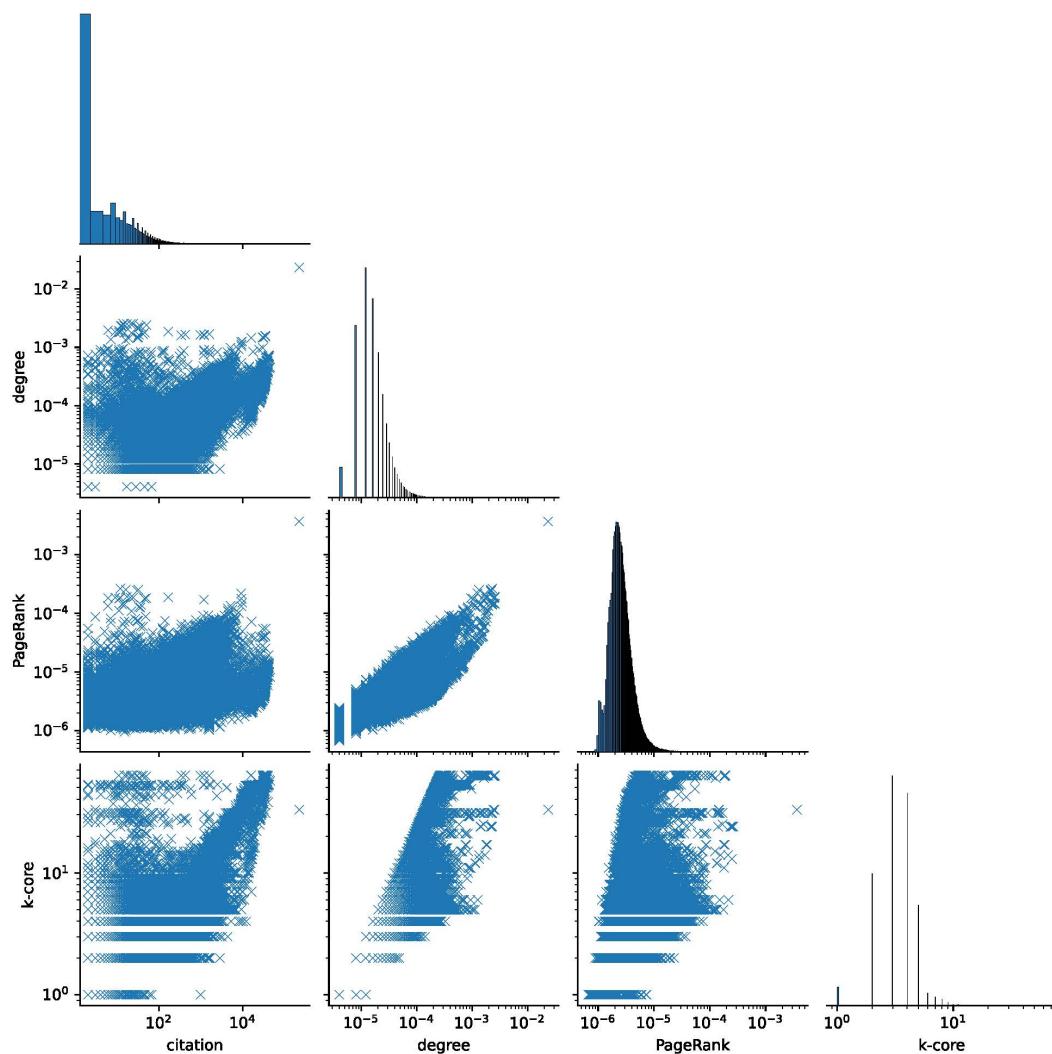


図 4.4: 物理分野の共著ネットワークにおける中心性指標と被引用数の散布図行列

## 4.9 さらなる学習のために

本稿では基本的な中心性指標をいくつか紹介したが、ネットワークからなんらかの意味での重要なノードを特定するというタスクは、今でも活発に研究されている。本稿では、ノードの中心性 (centrality) という言葉で説明したが、ノードの影響力 (influence) を推定する、influential なノードを特定するという言い方もする。他にも、spreader や key node のような似たようなものが違う言葉で呼ばれていたりするので、論文をキーワード検索する時には少し注意が必要である。さて、重要なノードの特定のためのサーベイ論文としては、文献 [57] が、網羅性が高く参考になる。また、各ノードの重要度を推定するのと似たタスクとして、ネットワークから重要なノード（影響力の強いノード）の「集合」を特定するという研究も活発である。重要なノードの集合を特定する研究にもいくつかバリエーションがあるが、「影響最大化」 [14] というのがキーワードである。影響最大化に特化したサーベイ論文としては、文献 [58] が詳しい。

### コラム：50年後、100年後も引用される論文

筆者がこれまで執筆した論文の中で最も引用している回数が多いのは、おそらく Freeman の中心性の論文 [43] である。この論文が発表されたのは、1979 年であり、今から 40 年以上も前である。どんどん新しい技術が登場しては消えていくが、やはり基本的な概念を提案した論文というのは、時間が経っても引用される。今後、ネットワーク科学分野はどうなっていくか分からないが、筆者が引退する 30 年くらい先にもまだ Freeman 先生の論文を引用しているような気がする。このように何十年か後にも引用されるような論文を書いてみたいものである。

# 5

## コミュニティ抽出

現実のネットワークの多くは、ノード同士が互いに密に接続されたコミュニティと呼ばれるノードの集合と、コミュニティ間を結ぶ疎なリンクで構成されていることが多い。このような構造をコミュニティ構造と呼び、ネットワークの中からコミュニティを発見するタスクはコミュニティ抽出 (community detection) と呼ばれる。本章では、「コミュニティとは何か？」というところから始まり、ネットワークからコミュニティを抽出するためのいくつかの代表的な手法を紹介する。

### 5.1 コミュニティとは？

コミュニティ構造のイメージを図 5.1 に示す。このようなネットワークを人が見ると、破線で囲ったいくつのグループにノードを分けられるように見える。それぞれのノードのグループをコミュニティと呼び、このようにコミュニティにうまく分けられるようなネットワークをコミュニティ構造を有するネットワークと呼ぶ。図 5.2 にもう少し大規模なコミュニティ構造を有するネットワークの例 (Facebook ネットワークのデータ [35] にコミュニティを表す色を付けて可視化) を示す。現実のネットワークは、このようなコミュニティ構造を有していることが多い。一般名詞としての「コミュニティ」は人の属する集団のような意味合いがあるが、ネットワーク分析における「コミュニティ」というテクニカルタームは、単にネットワーク構造上の密なノードの集合のことである。ノードが人ではなくとも「コミュニティ」と呼ぶ。

ネットワークにおけるコミュニティは、現実の何らかの意味のある集団と対応することが経験的に知られている [59]。そのため、コミュニティの抽出は、ソーシャルネットワークでの情報推薦 [60]、情報拡散の予測 [61]、脳機能の推定 [62] など色々な応用が考えられ

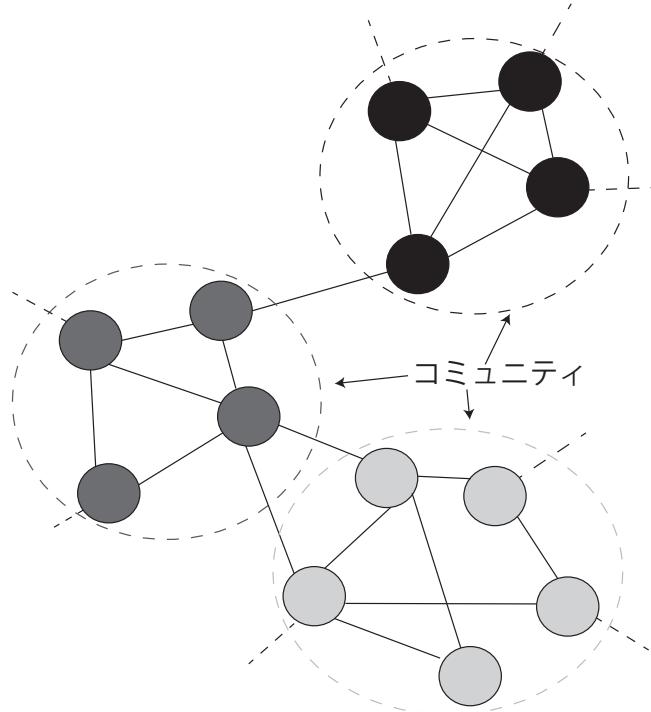


図 5.1: コミュニティ構造のイメージ

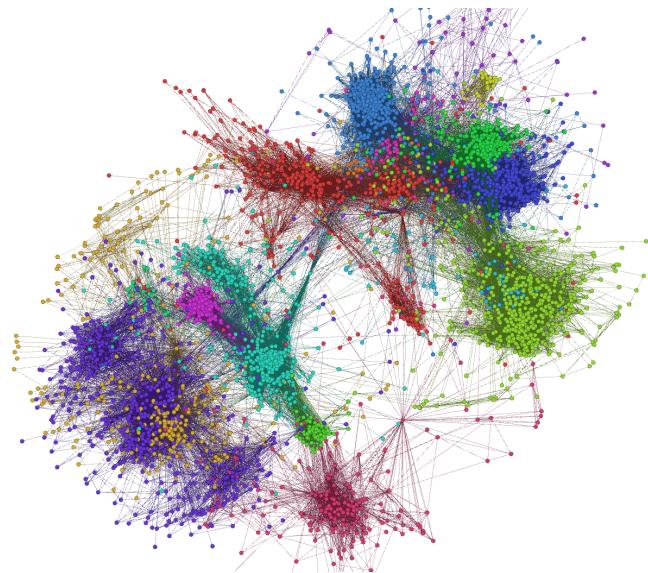


図 5.2: コミュニティ構造を有する大規模なネットワークの例 (Facebook)

ている。

さて、コミュニティについて図を用いて「このような感じ」とあいまいに説明してきたのを厳密に定義したいところであるが、実はこれ以上の厳密な合意の得られているコミュニティの定義というのは存在しない。コミュニティというのは、非常にあいまいな概念であり、様々な考え方がある [63, 64]。広く合意の得られていることとしては、コミュニ

ティの中のリンクが相対的に密であり、コミュニティの間を結ぶリンクは相対的に疎であるということである [63]。ネットワークの構造からこのような特徴を持つノードの集合を抽出することがコミュニティ抽出の目的である。

しかしこのようなあいまいな定義のままでは、問題の解きようがないため、コミュニティ抽出アルゴリズムを考える際には、それぞれのアルゴリズムで「このようなコミュニティを抽出する」というコミュニティ抽出の基準が定められている。多くのコミュニティ抽出アルゴリズムは、「コミュニティらしさ」を定量化する指標を定義し、その指標を最大化するようなコミュニティを見つけるという形でコミュニティ抽出問題を定式化し、解いている。コミュニティらしさを定量化する指標の中で現在広く用いられているものが、「モジュラリティ (modularity)」と呼ばれる指標である [65]。本稿では特に、モジュラリティに基づくアプローチを紹介した後、他のアプローチについても若干 5.7 節で議論する。

具体的な手法の話の前に、コミュニティ抽出アルゴリズムの分類を説明する。コミュニティ抽出アルゴリズムには、全てのノードを 1 つのコミュニティに対応させる disjoint community detection と、1 つのノードが複数のコミュニティに所属させる overlapping community detection が存在する [63, 66] \*<sup>1</sup>。人のネットワークを考えると、人は複数の集団に属していると考えるのが自然であり、そのような構造を抽出するのが overlapping community detection であり、ネットワークを複数のコミュニティに分割してしまうのが disjoint community detection である。以降で紹介するのは、disjoint community detection である。disjoint community detection のことをコミュニティ分割と呼ぶこともあり、本稿の中でもそのように呼ぶ場合もある。また、あるコミュニティが複数のコミュニティで構成されるような階層構造を持っている場合に、その階層構造を陽に抽出する階層的 (hierarchical) コミュニティ抽出 [63, 67] も存在する。例えば「筑波大学」のコミュニティはいくつかの「学群」のコミュニティで構成され、各学群はいくつかの「学類」のコミュニティで構成される\*<sup>2</sup>と考えられる。このような階層構造を抽出したい場合には階層的なコミュニティ抽出が有用である。本稿で紹介するのは、非階層的なコミュニティ抽出であるが、次節で紹介する Girvan-Newman 法は結果的に、階層的なコミュニティが得られる\*<sup>3</sup>。また、ネットワーク中の全ノードのコミュニティを見つけるのではなく、ある指定したノードのコミュニティを発見する community search [68] と呼ばれるタイプの手法も存在する。また、ネットワーク全体をコミュニティに分割してしまうのではなく、コミュニティらしい部分だけを抽出する community extraction [69] というタイプの手法も存在する。本稿では、community search や extraction については扱わない。

\*<sup>1</sup> クラスタリングにおけるハードクラスタリングとソフトクラスタリングの違いである

\*<sup>2</sup> 学群は学部、学類は学科に対応する組織である。

\*<sup>3</sup> 階層構造を得ることを目的に設計されたわけではない

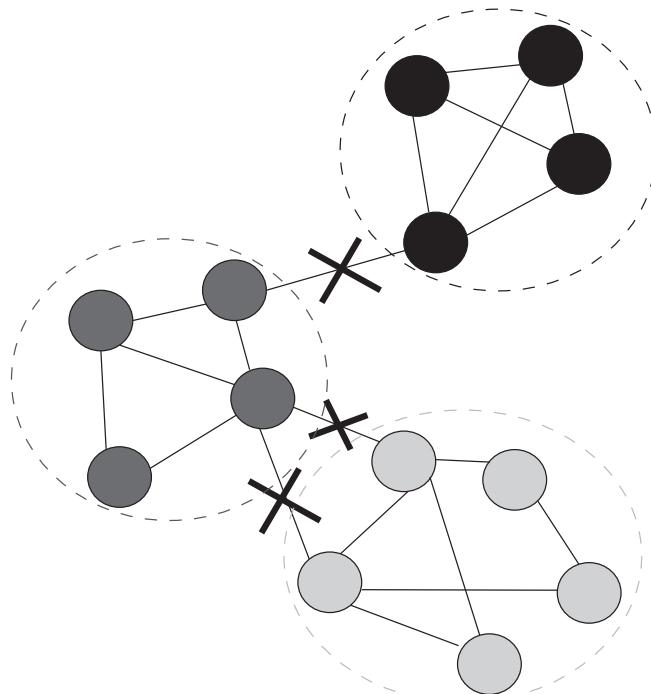


図 5.3: Girvan-Newman 法によるコミュニティ抽出のイメージ

## 5.2 Girvan-Newman 法

ネットワーク科学分野でコミュニティ抽出の研究が盛り上がるきっかけとなったのが、Girvan と Newman の手法 [65] である。ちなみに、Girvan 先生も Newman 先生も著名なネットワーク科学者である。

Girvan-Newman 法の基本的な考え方は、ネットワークからコミュニティをまたがるリンクを削除していく、ネットワークを複数の連結成分に分解した時の、それぞれの連結成分をコミュニティと考えるということである。図 5.3 は Girvan-Newman 法のイメージを表す図である。このように「コミュニティの境界らしい」リンクを順に切断していくことで、ネットワークが複数の連結成分に分かれていく。連結成分の数（コミュニティの数）が所望の値になった時点で分割をストップすることで、コミュニティが得られる。分割を早期に止めることで粗いコミュニティが、分割を繰替えすることで細かいコミュニティが得られるため、階層的なコミュニティを抽出していると見ることもできる。

さて、削除するリンクの順番を決めるに当たっての問題が、リンクの「コミュニティの境界らしさ」をどのように定量化するかである。Girvan-Newman 法では、リンクのコミュニティの境界らしさを、リンクの媒介中心性 (link betweenness) [65] で定量化する。ノードの媒介中心性を 4 章で紹介したが、それをリンクに拡張したのがリンクの媒介中心性である。リンク  $e$  の媒介中心性は次式で定義される。

$$C_{\text{bet}}(e) = \frac{\sum_{s,t \in V, s \neq t \neq e} g_{st}(e)}{\sum_{s,t \in V, s \neq t} g_{st}} \quad (5.1)$$

ここで  $g_{st}(e)$  はノード  $s, t$  間の最短経路のうち、リンク  $e$  を通過するものの数である。Girvan-Newman 法では、媒介中心性が最大のリンクを削除する。削除したネットワークで媒介中心性を再計算し、媒介中心性最大のリンクを削除するという手順を繰り返す。整理すると、Girvan Newman 法の手順は以下の通りである。

#### Girvan Newman 法

1. 各リンクの媒介中心性を計算する。
2. ネットワークから媒介中心性が最大のリンクを削除する。
3. リンクを削除したネットワークで媒介中心性を再計算し、媒介中心性が最大のリンクを削除する。
4. ステップ 3 を所望のコミュニティ数（連結成分の数）に分かれるまで繰り替えす。

Girvan Newman 法の弱点の 1 つとして、計算量が挙げられる。Girvan Newman 法の計算量は疎なネットワークで、 $\mathcal{O}(N^3)$  である。リンクの媒介中心性の計算が  $\mathcal{O}(MN)$ 、最悪は全てのリンクを削除するまで再計算を繰り返すので、全体で  $\mathcal{O}(M^2N)$ 、スパースなネットワークでは、リンク数はノード数の定数倍のオーダーとすると、 $\mathcal{O}(N^3)$  ということである。Girvan Newman 法は直感的にもコミュニティらしい構造を抽出するのに有用ではあるが、計算量の問題から、現在においてこの手法を使うことは稀である。

もう一つ、この方式を使う際には、いつコミュニティ分割を止めればいいかを決める必要があるという問題もある。これについては、実は、Girvan と Newman らが止めるための基準も提案している [65]。それがモジュラリティと呼ばれる、後にコミュニティ抽出のために広く用いられる指標である。次節ではそのモジュラリティの定義を紹介する。

## 5.3 モジュラリティ

モジュラリティはコミュニティ抽出結果の良さを測る代表的な指標である [65, 59]。モジュラリティの問題点を指摘する研究 [70] も存在するが、実用的にはモジュラリティは有用であり、現在も広く用いられている。ネットワーク  $G$  からコミュニティ  $C = \{c_1, c_2, \dots, c_k\}$  が抽出されたとする。ここで、 $c_i \in C$  は空でないノード集合であり ( $\forall i, V \supseteq c_i \neq \emptyset$ )、 $\bigcup_{i=1}^k c_i = V$  を満たすとする。この時、ネットワーク  $G$  における抽出されたコミュニティ  $C$  に対するモジュラリティ  $Q(C)$  は次式で定義される。

$$Q(C) = \sum_{c_i \in C} \left( \frac{e_{c_i}}{M} - \left( \frac{a_{c_i}}{2M} \right)^2 \right) \quad (5.2)$$

ここで  $e_{c_i}$  はコミュニティ  $c_i$  に属するノード同士を接続するリンクの数、 $a_{c_i}$  は、コミュニティ  $c_i$  に属するノードの次数の和である。

このモジュラリティの意味するところを説明する。良いコミュニティ分割とは、コミュニティ内のリンクが多く、コミュニティをまたがるリンクが少ないような分割である。そのような場合にモジュラリティ  $Q(C)$  は大きな値を取るように設計されている。総和の

中の前の部分は、全リンクに対するコミュニティ内のリンクの割合を表している。つまり、モジュラリティはコミュニティ内のリンクが多いほど高い値を取る。では、コミュニティの中のリンクを多くしたければ、どのように分割すればよいだろうか？自明な方法は、ネットワーク全体を一つのコミュニティとする方法である。こうすれば、全てのリンクはコミュニティ内に存在することになる。しかし、それは良いコミュニティの直感と反する。このような、「コミュニティの中のリンクは多いが、コミュニティらしくない」ような分割にはペナルティを与えることを考える。そのペナルティの項が総和の後の部分である。ノードの次数分布はそのままに、各リンクの接続先をランダムに貼り替えるという操作を考える。その時、コミュニティ  $c_i$  の中のノード同士を接続するリンクの割合の期待値が  $(\frac{a_{c_i}}{2M})^2$  である。各ノードのリンクを全て切断し、各ノードが接続先の決まっていない「手」を持っている状況を考え、それぞれの手の接続先をランダムに決めるとしよう。コミュニティ  $c_i$  に属するノードの有する手の総数は  $a_{c_i}$  本である。それらの接続先の手の候補は  $2M$  本ある。それぞれの手の接続先を  $2M$  本の接続先の候補の中からランダムに選ぶわけであるが、 $c_i$  に属するノードから出ていった 1 本の手が、 $c_i$  に属するノードの有する手と接続される確率は  $a_{c_i}/2M$  である（自己ループも許容する）。よって、次数分布そのままにランダムにリンクを貼り替えるという操作をした時の、コミュニティ  $c_i$  同士で結ばれる手の本数の期待値は  $(a_{c_i})^2/2M$  である。全部の手のうちコミュニティ  $c_i$  同士で結ばれる手の割合が  $(a_{c_i}/2M)^2$  となる。つまり、モジュラリティ  $Q(C)$  は実際に観測されたコミュニティ内リンクの割合から、ランダムにリンクをつなぎかえるという操作をした時のコミュニティ内リンクの割合を引いたものとして定義されている。

前節で紹介した Girvan と Newman の方法において、コミュニティ分割を止めるための基準として、このモジュラリティを用いることができる。リンクを切断し、コミュニティ分割が更新されるたびにモジュラリティを計算する。リンクを全て切断した時点で、モジュラリティが最大であったコミュニティ分割を求めるコミュニティとして出力する。これによって、コミュニティ数を予め決めずに、ネットワーク構造のみからコミュニティを求めることができる。

さて、Girvan-Newman 法の計算量が大きいことは既に述べた通りであるが、モジュラリティが最大となるようなコミュニティ分割を求めたいのであれば、リンクの媒介中心性にこだわる必要はない。そこで、モジュラリティが提案されて以降は、モジュラリティを直接的に最大化するというアプローチの手法が多く提案されている。以降、モジュラリティ最大化の初期の手法として Newman 法 [71] を、また大規模ネットワークに対しても適用可能な手法として Louvain 法 [72] をそれぞれ紹介する。

## 5.4 Newman 法

厳密な意味でのモジュラリティ最大化は NP-hard である [73] ため、Newman<sup>\*4</sup> は貪欲的にモジュラリティを高めるアルゴリズムを提案した [71]。Girvan-Newman 法はネットワーク全体を 1 つのコミュニティと考え、小さなコミュニティに分割していくが、Newman 法は、各ノードが別々のコミュニティであるという状態から、コミュニティの併合を繰り返すことで、コミュニティを求める。2 つのコミュニティを併合すると、モジュラリティが変化するが、コミュニティの併合によってモジュラリティの増加が最大となる 2 つのコミュニティを順に貪欲的に併合していくのが Newman 法である。Newman 法でコミュニティを求める手順は以下の通りである。

### Newman 法

1. 各ノードそれぞれを一つのコミュニティとする（ノード  $V = \{v_1, v_2, \dots, v_N\}$  をそれぞれ  $C_1 = \{c_1, c_2, \dots, c_N\}$  に割り当てる）。
2. 現在のコミュニティ分割  $C_t$  に属する全コミュニティの組合せについて、コミュニティを併合した時のモジュラリティの増加分  $\Delta Q(i, j)$  を求める。
3.  $\Delta Q(i, j)$  が最大となるコミュニティ  $c_i$  と  $c_j$  を併合することでコミュニティ分割  $C_{t+1}$  を得る。
4. ステップ 2、3 を全コミュニティが併合され 1 つのコミュニティとなるまで繰り返す。
5. モジュラリティ  $Q(C_t)$  が最大となる  $C_t$  を出力する。

ここで、 $\Delta Q(i, j)$  の計算自体はそれほど重たくはない。式 (5.2) を見るとわかる通り、コミュニティ  $c_i$  と  $c_j$  の統合で変化するのは、 $c_i$  と  $c_j$  の和の部分だけである。実際に計算すると、

$$\Delta Q(i, j) = 2(2M e_{i,j} - a_i a_j) \quad (5.3)$$

となる。ここで  $e_{i,j}$  はコミュニティ  $i$  と  $j$  をつなぐリンクの割合である。さて、Newman 法の計算量を考える。1 回の併合に関して、コミュニティの組合せだけ  $\Delta Q(i, j)$  の計算が必要であるが、リンクのないコミュニティペアの併合はモジュラリティの増加をもたらさないため、考えなくてよい。ということで、1 回の併合あたりリンク数  $M$  のオーダーのコミュニティペアについて比較が必要となる。併合するコミュニティが決まった後、モジュラリティ計算のための更新処理のオーダがノード数  $N$  である。 $N - 1$  回併合を繰り替えすとアルゴリズムが停止するため、全体では計算量は  $\mathcal{O}((M + N)N)$  である。疎なグラフであれば、 $\mathcal{O}(N^2)$  で Girvan-Newman 法よりもだいぶ高速である。なお、Newman 法の後、CNM 法 [74]<sup>\*5</sup> という拡張も提案されて、その手法は  $\mathcal{O}(M \log(N))$  である。

<sup>\*4</sup> Girvan-Newman 法の Newman 先生と同一人物である

<sup>\*5</sup> Clauset, Newman, Moore のイニシャル。彼らもやはりこの筋の有名人である

## 5.5 Louvain 法

Newman 法や CNM 法よりもさらに高速なアルゴリズムとして登場したのが Louvain 法 [72]<sup>\*6</sup>である。計算量としては  $\mathcal{O}(M)$  を達成している [75]。Louvain 法のさらなる改良も色々と提案されているが、Louvain 法でも数百万ノード程度のネットワークを扱うことができるため、現在も Louvain 法はよく使われる手法の 1 つである。

Louvain 法は (1) 局所的なモジュラリティの最適化、(2) コミュニティを構成するノードとリンクの集約の 2 つのステップを繰り返すことで、コミュニティを求める。1 つのステップでは、各ノード  $i$  を隣接するノードと同じコミュニティに併合した際のモジュラリティの増分  $\Delta Q$  を求め、 $\Delta Q > 0$  が最大となるコミュニティに、ノード  $i$  を併合する。2 つのステップでは、1 つのコミュニティを 1 つのノードに集約した新たなネットワークを構築する。1 つのコミュニティを構成する全てのノードが 1 つのノードに集約され、コミュニティ内のリンクは集約後のネットワークでは、自己ループとして表現される。ただし、自己ループの重みは元のコミュニティ内リンク数の 2 倍である。また、集約後のノード  $i, j$  間のリンクの重みは、集約前のコミュニティ  $i, j$  間のリンク数に等しい値とする。この 2 つのステップをモジュラリティが上昇しなくなるまで繰り替えす。図 5.4 に Louvain 法の各ステップの概要を示す。

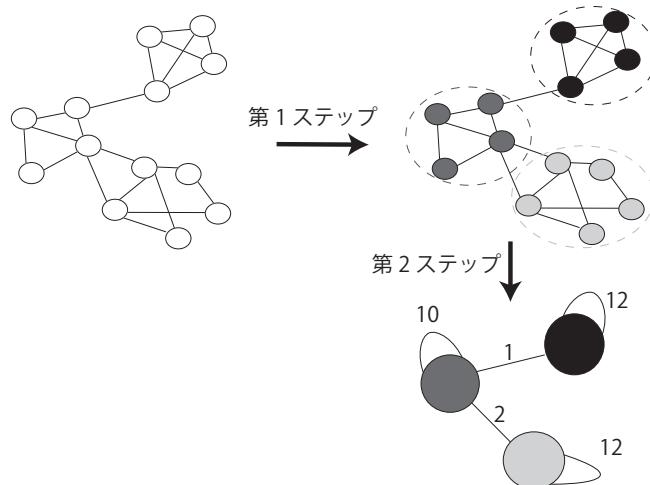


図 5.4: Louvain 法の概要

Louvain 法の手順をまとめると以下の通りである。

<sup>\*6</sup> これは人の名前ではない。University of Louvain で研究されたことにちなんんでいる

### Louvain 法

全ノードがそれぞれ異なるコミュニティに所属する状態からスタートし、モジュラリティが上昇しなくなるまで以下を繰り替えす。

#### 1. (局所的なモジュラリティの最適化)

- 各ノード  $i$  を隣接するノードと同じコミュニティに併合した際のモジュラリティの増分  $\Delta Q$  を求め、 $\Delta Q > 0$  が最大となるコミュニティに、ノード  $i$  を併合する。
- 上記の手順を全ノードについて、コミュニティの変化が発生しなくなるまで繰り返す

#### 2. (コミュニティの集約) 1つのコミュニティを構成する全てのノードを1つのノードに集約したネットワークを構築する。集約前のコミュニティ内リンク数の2倍の重みを持つ自己ループを集約後のネットワークの各ノードに生成する。また、集約後のノード $i, j$ 間のリンクの重みは、集約前のコミュニティ $i, j$ 間のリンク数に等しい値とする。集約したネットワークに対して、ステップ1(局所的なモジュラリティの最適化)を実施する。

Louvain 法では、コミュニティを1つのノードに集約することで、ノードおよびリンクの参照回数を削減し、高速化を実現している。なぜ集約してしまっても、 $\Delta Q$  が計算できるのかについて説明していなかったが、直感的な説明としては、結局コミュニティの中のリンク数とコミュニティ間のリンク数さえ分かれれば、モジュラリティは計算できるためである。集約後のネットワークではリンク数の情報を重みとして持っているため、モジュラリティを計算することができる。また、Louvain 法のステップ1では、各ノードの隣接ノードとコミュニティ併合の可能性を検討する。この結果として、コミュニティサイズの大きな偏りが緩和される（非常に巨大なコミュニティができてしまうことを防ぐ）という利点も確認されている [72]。

## 5.6 実ネットワークのコミュニティ抽出

これまでに学んだコミュニティ抽出の手法を実ネットワークに適用してみよう。ここでは、コミュニティの話をする時には欠かせない、Zachary's Karate Club [76] のデータを用いる。図 5.5 に Zachary's Karate Club のネットワークを示す。これは、ある空手クラブの交友関係を表したネットワークである。このクラブは、Mr. Hi (instructor) と John A. (Officer) の2人の対立により分断されてしまう。空手クラブの交友関係のネットワークが、2つの対立する派閥を反映しており、その派閥を正しく検出できるかどうかが、コミュニティ抽出のベンチマークに広く用いられている<sup>\*7</sup>。

\*7 ネットワーク科学のコミュニティには、Zachary Karate Club Club というものがあり、国際会議での Karate Club のデータを使って発表をした人に（何らかの基準で選ばれて？）賞が贈られる。

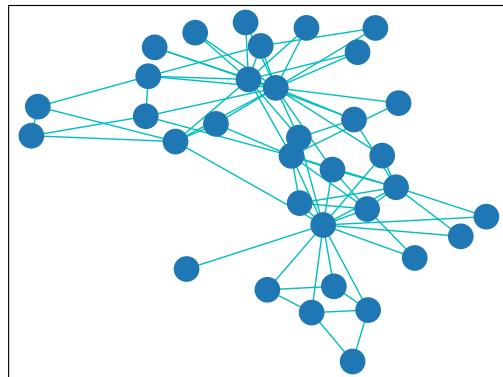


図 5.5: Zachary's Karate Club のネットワークの可視化

Karate Club のネットワークから Girvan-Newman 法でコミュニティ抽出するプログラムは以下の通りである。

#### Karate Club のコミュニティ抽出

```
1 import networkx as nx
2 import matplotlib.pyplot as plt
3
4 g=nx.karate_club_graph()
5 labels=nx.get_node_attributes(g,'club')
6
7 #コミュニティ抽出
8 from networkx.algorithms import community
9 com=community.girvan_newman(g)
10 list=tuple(sorted(c) for c in next(com))
11
12 # 可視化
13 pos=nx.spring_layout(g)
14 nx.draw_networkx_nodes(g, pos, nodelist=list[0],
node_shape='^',node_color='b', alpha=0.5)
15 nx.draw_networkx_nodes(g, pos, nodelist=list[1],
node_color='g', alpha=0.5)
16 nx.draw_networkx_labels(g, pos, labels=labels)
17 nx.draw_networkx_edges(g, pos, alpha=0.4, edge_color='C')
18 plt.savefig("karate_com.eps")
```

有名なデータなので NetworkX にデータが付属している。後半は可視化のためのコードであり、コミュニティ分割自体は、8~10行目だけである。

Karate Club のコミュニティ抽出結果と、実際の派閥の対応を図 5.6 に示す。ノードの形が抽出した 2 つのコミュニティを表している。ラベルは派閥を示している。これを見ると、中央付近の 2 人を除いて正しく派閥通りにコミュニティを分けることができている<sup>\*8</sup>。交友関係のみから対立派閥を特定できるというのはなかなか興味深い話である。

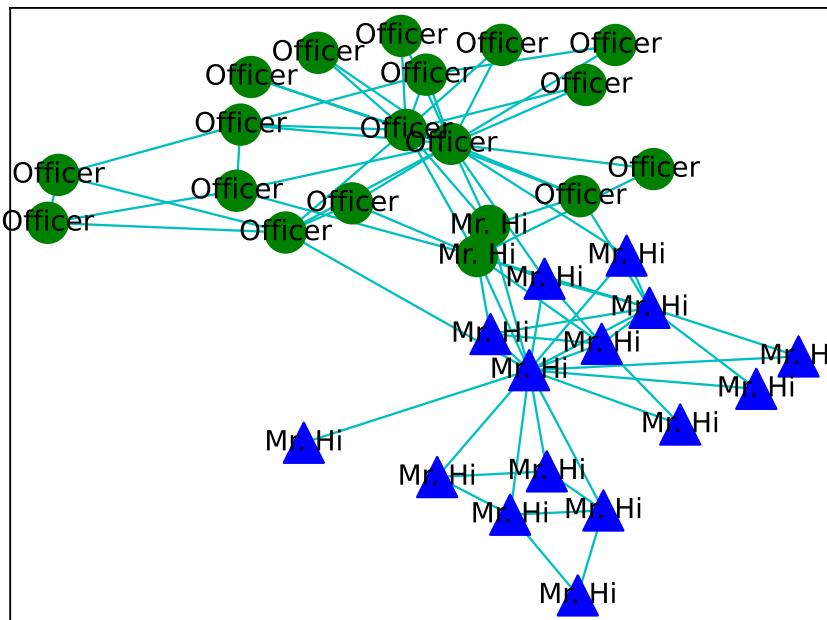


図 5.6: Zachary's Karate Club に対するコミュニティ抽出結果（ノードの形がコミュニティの違いを表す）

次にもう少し大きなネットワークのコミュニティを抽出してみよう。SNAP[42] で公開されているヨーロッパの研究組織のメールのネットワーク<sup>\*9</sup>[77] を用いる。図 5.7 にネットワークを可視化した図を示す。カラー版ではノードの色はコミュニティを表している。このデータには、各ノードの所属組織のラベルも付与されている。そこで、この所属組織のラベルと抽出したコミュニティがどの程度一致しているかを調べてみる。

<sup>\*8</sup> 元々の Girvan と Newman [65] の論文では、1 ノードだけ間違っているという結果が報告されている。Karate Club のデータは 2 種類存在するので、その違いに起因するものと思われるが、詳しく調べていない。

<sup>\*9</sup> <http://snap.stanford.edu/data/email-Eu-core.html>

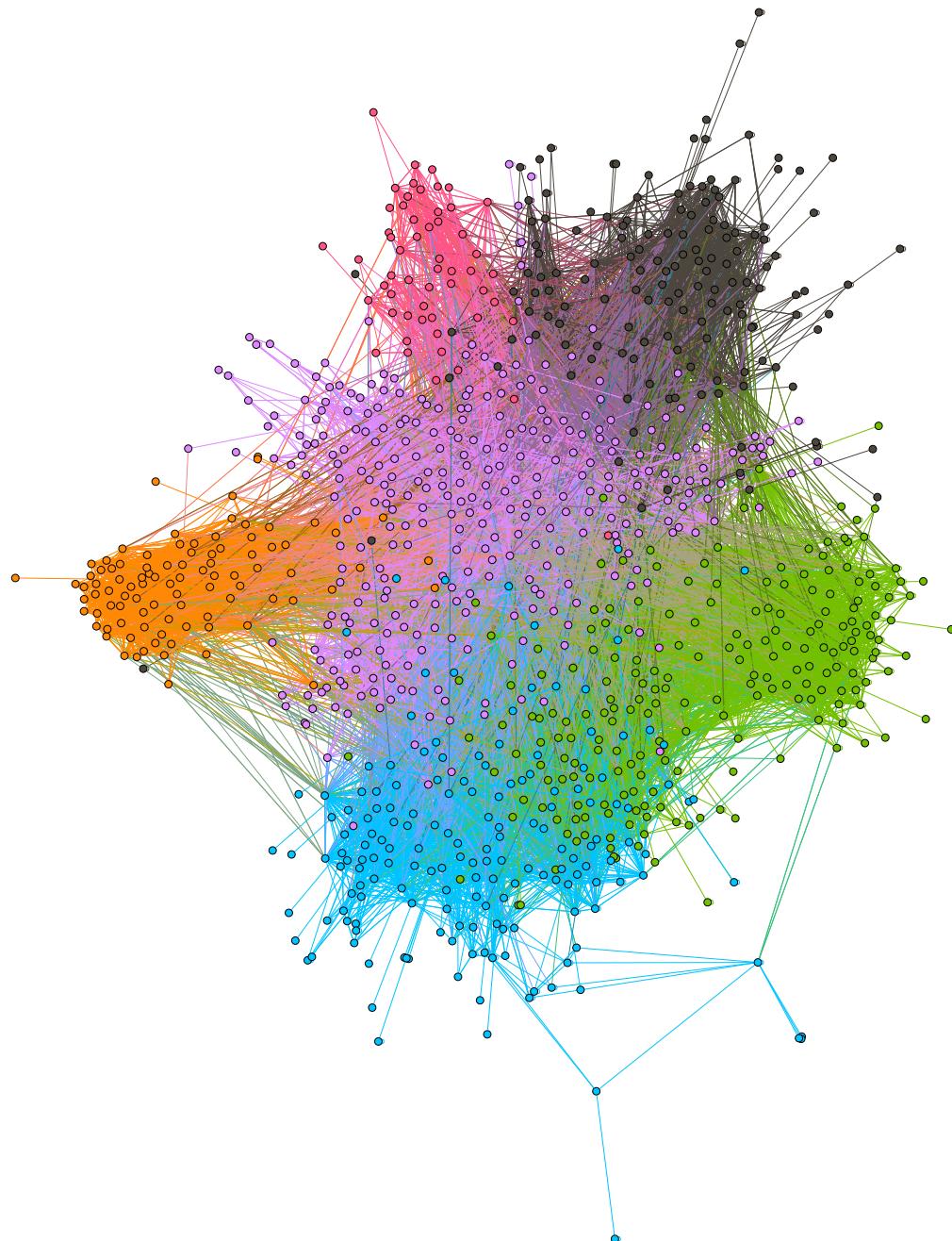


図 5.7: EU email ネットワークの可視化 (カラー版でノードの色はコミュニティを表す)

以下に抽出したコミュニティと実際のラベルの対応を出力するプログラムを示す。ここでは Newman 法を高速化した CNM 法 [74] でコミュニティを求めている。

## EU email ネットワークのコミュニティ抽出

```
1 import sys
2 import networkx as nx
3
4 def each_line(filename):
5     with open(filename, encoding='utf-8') as f:
6         for line in f:
7             yield line.rstrip('\r\n')
8
9 file=sys.argv[1]
10 label_f=sys.argv[2]
11 g=nx.read_edgelist(file)
12
13 #ノードラベルの読み込み
14 label_id={}
15 for l in each_line(label_f):
16     [v,label]=l.split(" ")
17     label_id[v]=int(label)
18
19 #CNM法でコミュニティを抽出
20 from networkx.algorithms.community import
greedy_modularity_communities
21 c=list(greedy_modularity_communities(g))
22
23 #ノードにコミュニティの番号を割り当てる
24 id=0
25 com_id={}
26 for com in c:
27     id+=1
28     for v in sorted(com):
29         com_id[v]=id
30 #ノードのコミュニティとラベルの対応を出力
31 for v in g.nodes():
32     print(v,com_id[v],label_id[v])
```

さて、この出力の最初の 10 ノード分は以下の通りである。

```

0 2 1
1 2 1
2 1 21
3 1 21
4 1 21
5 3 25
6 2 25
7 4 14
8 4 14
9 4 14

```

ノード 0 と 1 は同じコミュニティ 2 に所属しており、部署も同じ 1 番である。2~4 は同じコミュニティ 1 で、部署も 21 番、7~9 もコミュニティ 4 番で部署は 14 番、とまずまずコミュニティと部署が対応しているように見える。

このネットワークのノード数は 1005 であり、全てを目で見たり、Karate Club のようにネットワーク中に可視化しても、全体としてコミュニティ分けがうまくいったかはよく分からなさそうである。得られたコミュニティと、ノードのラベルが与えられた時に、コミュニティとラベルがどのくらい一致しているかを評価できるとうれしい。

ノードのラベルが与えられている時のコミュニティ抽出の評価指標は色々とあるが、その中でも直感的な解釈が容易と思われる、Rand Index という指標を紹介する。Rand Index を求めるには、全てのノードペア  $u$  と  $v$  について、 $u$ 、 $v$  が同一のコミュニティに属するか否か、同一のラベルを有するか否かを調べる。そして、同一コミュニティでかつ同一ラベルのペアの数 (TP)、同一コミュニティだが異なるラベルのペアの数 (FP)、異なるコミュニティで異なるラベルのペアの数 (TN)、異なるコミュニティだが同じラベルのペアの数 (FN) を求める (表 5.1 に TP、FP、TN、FN の対応を示す)。同一コミュニティでかつ同一ラベルのペアの数 (TP) と異なるコミュニティで異なるラベルのペアの数 (TN) が多い方が、コミュニティとラベルが一致していると考えられる。逆に同一コミュニティだが異なるラベルのペアの数 (FP) と、異なるコミュニティだが同じラベルのペアの数 (FN) は少ない方が、コミュニティとラベルが一致していると考えられる。そこで、Rand Index は次式で定義される。

$$RI = \frac{TP + TN}{TP + FP + TN + FN} \quad (5.4)$$

表 5.1: TP、FP、FN、TN の説明

	同じコミュニティ	異なるコミュニティ
同じラベル	TP	FP
異なるラベル	FN	TN

EU email network で Rand Index を計算してみると、0.77 であった。Rand Index は 0~1 の値を取る指標であり、0.77 という値は、そこそこ良いコミュニティが得られていると考えても良さそうである。ただし、Rand Index は対象によって、適当なコミュニティ分割をした時の値が異なる。ランダムにコミュニティを分けてもそこそこ高い値になってしまうこともある。そこで、チャンスレベルの Rand Index の値で補正をかけた Adjusted Rand Index [78] が実際にはよく用いられる。

## 5.7 コミュニティ抽出に関する議論

本稿では、モジュラリティに基づくコミュニティ抽出手法を紹介したが、モジュラリティに基づくコミュニティ抽出の問題点について、いくつかの議論が存在する [70, 63, 79]。コミュニティ構造のはっきりしたネットワークで、良いコミュニティ分割ができればモジュラリティは高くなるが、コミュニティ構造がそもそも存在しないようなネットワークであっても、モジュラリティを高めることが可能である [80]。resolution limit [81] というテクニカルタームで知られているが、モジュラリティ最大化のアプローチでは、小さなコミュニティを抽出できないといったことが指摘されている。また、モジュラリティの最大値に近い値で、大きく異なる複数のコミュニティ分割が存在するということも指摘されている [82]。

では、このような問題点が存在するので、モジュラリティに基づくコミュニティ抽出は役に立たないかというと、実用上は、Louvain 法などのモジュラリティに基づく手法でもそこそこ良い（直感的に妥当そうな）コミュニティが得られると考えられている（少なくとも筆者はそう考えている）。対象のネットワークがはっきりとしたコミュニティ構造を有していれば、上記の問題点もそれほど深刻にならない。ただし、コミュニティ抽出手法は今でもアクティブな研究トピックであるので、今後トレンドは大きく変わる可能性がある。

モジュラリティ以外のアプローチについても、ここで簡単にピントを提示しておく。モジュラリティ以外にもコミュニティの良さを定量化する指標はいくつか存在し、それらを最適化することでコミュニティを得る手法が提案されている。代表的なものとしては、normalized cut [83] や conductance [84] などが挙げられる。また行列分解 [85] に基づく手法やラプラス行列のスペクトル指標に基づく手法 [86] も存在する。ネットワーク科学のコミュニティでホットなアプローチは統計的推論に基づく方法である [87, 88, 79, 89, 90]。ネットワークがある確率モデルに従って生成されると仮定し、観

測されたネットワークがそのモデルに適合するようなパラメータを求めてることでノードの所属するコミュニティを求めるというのが概要である。詳細はサーベイ論文 [90] などを参照してほしい。

## 5.8 さらなる学習のために

コミュニティ抽出は現在も活発に研究が続いている、コミュニティの定義や抽出方法も様々である。本稿では、初学者がとりあえず最初におさえておくべきコンセプトや基本的な方法を解説したが、コミュニティ抽出の研究の全体像を把握するには本稿の内容だけでは不十分と考えられる。本章の最後に、今後コミュニティ抽出についてさらに学習する際に有用と思われる資料をいくつか提示する。コミュニティ抽出手法の比較や全般的なサーベイについては文献 [66, 91, 64, 92, 93, 94] などが参考になる。より最近のトピックとして、多層ネットワークのコミュニティ抽出についても既にサーベイ論文 [95] が出版されている。

# 6

## ネットワーク分析の実践 1：ネットワークのロバスト性

通信網や交通網を表現したネットワークでは、ネットワークの連結性が重要である。ネットワークからノードやリンクが欠落した時に、ネットワークの連結性がどの程度維持されるかをネットワークのロバスト性や頑健性と呼ぶ。本章では、ネットワークのロバスト性を評価する方法について解説する。

### 6.1 導入

ネットワーク全体としての「良さ」はどのように評価すればよいであろうか？当然、ネットワークが何を表現しているかや良さを評価する文脈に依存するので、この問いは答えようがない。インターネットであれば、平均遅延や平均スループットなどが例えば良さの評価尺度になるであろう。ソーシャルメディアユーザのネットワークであれば、ネットワークを通じてユーザが得られる情報の量や質が評価尺度になるであろうか。

ノードとリンクで抽象化されたネットワークの世界でネットワークの良さを評価する観点の一つとして、ネットワークのロバスト性（頑健性）という考え方がある [9, 96]。ネットワークからノードやリンクが欠落した時にネットワークがバラバラになりにくいほど、ネットワークは（ノードやリンクの欠落に対して）ロバストであると言われる。これは、通信網や交通網を考えた時に、ノード同士がつながっている（ノード間に経路が存在する）ことが、ネットワーク全体の機能を維持する上で重要であるという考え方から来たネットワークの評価の観点である。多少ノードやリンクが欠落してもネットワークのつながりが全体として維持されている方がよいというのは、納得できる考え方ではないだろうか。

ネットワークのロバスト性を評価するための尺度は、いくつか存在するが、最もよく用いられるのが、ノードやリンクを削除した時のネットワークの最大連結成分の大きさである [9]。ノードやリンクを削除していくと、連結なネットワークもいずれは非連結になる。おさらいになるが、非連結なネットワークのそれぞれの島を連結成分と呼ぶ。連結成分の中で一番大きいもの（属するノード数が最大のもの）が最大連結成分である。ノードやリンクを削除していくと最大連結成分のサイズがどんどん小さくなっていくが、たくさん削除しても最大連結成分のサイズが大きいネットワークはロバストであると評価される。連結しているかどうかだけでなく、ノード間の距離も考慮するような他のロバスト性の指標 [97] もあるが、ここでは最大連結成分の大きさだけにフォーカスして話を進める。

ネットワークのロバスト性に関する初期の重要な発見は、バラバシらの「インターネットのアキレス腱」のキャッチフレーズでも有名な、次数分布がべき分布に従うネットワークのロバスト性に関する性質である。バラバシら [9] は、インターネットのような次数分布がべき分布に従うネットワークは、ノードをランダムに削除した場合には高い連結性を維持できるが、次数が高いハブノードを優先的に削除した場合には、少数のノードを削除しただけで連結性が著しく低下することを示している。このことから、べき分布に従うネットワークはランダムなノードの故障 (random failure) に対して非常にロバストである一方、ハブノードを狙った攻撃 (targeted attack) に対して非常に脆弱であることを主張している。targeted attack に非常に弱いので、ここが「アキレス腱」であるということである。実際にインターネットが脆弱であるかということに対しては、異論もあるかと思われるが、この発見から、どのようにすれば、ネットワークのロバスト性を高めることができるか [98]、逆にどのようにノードを削除すればネットワークをバラバラにできるか [99] といった研究が活発化していった。現在でも、ネットワークのロバスト性というのはホットトピックの一つである。

以降では、このバラバシらの発見をなぞる形で、ロバスト性の評価方法を学ぶ。

## 6.2 ネットワークのロバスト性評価のためのプログラム

ロバスト性の評価の手順は、ネットワークからノードをなんらかの基準で削除し、削除した時の最大連結成分の大きさを計算する、ということである。最大連結成分の大きさは NetworkX では簡単に求まる。そのため、ノード削除の基準さえ決めてしまえば、ロバスト性を評価するプログラムはすっきりと書くことができる。ランダムにノードを削除する場合のサンプルプログラムは以下の通りである。

### ランダム削除

```
1 import sys
2 from networkx import *
3 import numpy as np
```

```
4
5 #データ読み込み
6 input=sys.argv[1]
7 g=read_edgelist(input)
8
9 n=len(g.nodes())
10
11 #最大連結成分の取得
12 giant = max(nx.connected_components(g), key=len)
13 removed=0
14 while(len(giant)>1):
15     #ランダムに1つノードを選択
16     v = np.random.choice(list(g.nodes()))
17     g.remove_node(v)
18     removed+=1
19     giant = max(nx.connected_components(g), key=len)
20 f_gc=len(giant)/n
21 f_removed=removed/n
22 print(f_removed,f_gc)
```

また、次数順に削除したければ以下のように書ける。

#### 次数順削除

```
1 import sys
2 from networkx import *
3 import numpy as np
4
5 #データ読み込み
6 input=sys.argv[1]
7 g=read_edgelist(input)
8
9 n=len(g.nodes())
10 #最大連結成分の取得
11 giant = max(nx.connected_components(g), key=len)
12 removed=0
13 deg=dict(g.degree())
14 #次数の高い順にループ
```

```
15 for v, k in sorted(deg.items(), key=lambda x: -x[1]):  
16     g.remove_node(v)  
17     removed+=1  
18     giant = max(nx.connected_components(g), key=len)  
19     f_gc=len(giant)/n  
20     f_removed=removed/n  
21     print(f_removed,f_gc)  
22     if(len(giant)<=1):  
23         break
```

上のプログラムでは、最初に次数を計算し、その次数順にノードを削除していたが、1つノードを削除すると、その影響で他のノードの次数は変化する。そこで、ノード削除のたびに次数を計算し直して削除対象（攻撃対象）を選ぶという考え方もある。このような攻撃手法（削除対象の決定手法）は high degree adaptive (HDA) と呼ばれ、次のようなプログラムで表現できる。

#### HDA

```
1 import sys  
2 from networkx import *  
3 import numpy as np  
4  
5 input=sys.argv[1]  
6 g=read_edgelist(input)  
7  
8 n=len(g.nodes())  
9  
10 giant = max(nx.connected_components(g), key=len)  
11 removed=0  
12 while(len(giant)>1):  
13     deg=dict(g.degree())  
14     max_v = max(deg, key=deg.get)  
15     g.remove_node(max_v)  
16     removed+=1  
17     giant = max(nx.connected_components(g), key=len)  
18     f_gc=len(giant)/n  
19     f_removed=removed/n  
20     print(f_removed,f_gc)
```

### 6.3 インターネットのアキレス腱

さて、前節のプログラムを用いて、バラバシらの発見をなぞってみよう。まず、次数分布とネットワークのロバスト性の関係を理解するため、2つのモデルから生成したネットワークを対象に実験を行う。1つは Erdős–Rényi (ER) モデルで生成した ER グラフ [100]、もう1つは Barabási–Albert (BA) model [7] で生成した BA グラフである。ここでは各モデルの説明は省略するが、NetworkX を用いれば、確率的なモデルに従ってネットワークを簡単に生成することができる<sup>\*1</sup>。ここでは、GitHub に生成済みのネットワークのエッジリストを置いてあるので、それを使うという前提で話を進める。ER グラフ、BA グラフの次数分布を図 6.1 に示す。ER グラフは次数分布が二項分布に従い、BA グラフは次数分布がべき分布に従う。また、可視化した結果を図 6.2 に示す。どちらもノード数 1,000、平均次数 8 である。

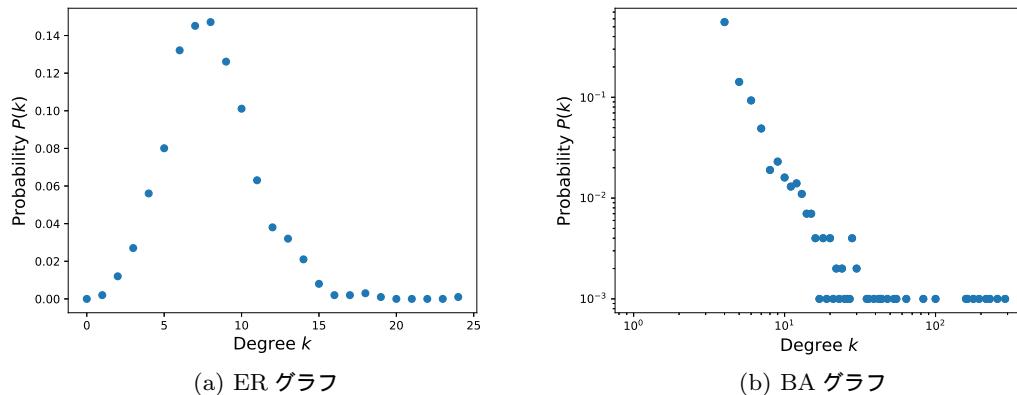


図 6.1: ER グラフと BA グラフの次数分布

<sup>\*1</sup> <https://networkx.org/documentation/stable/reference/generators.html>

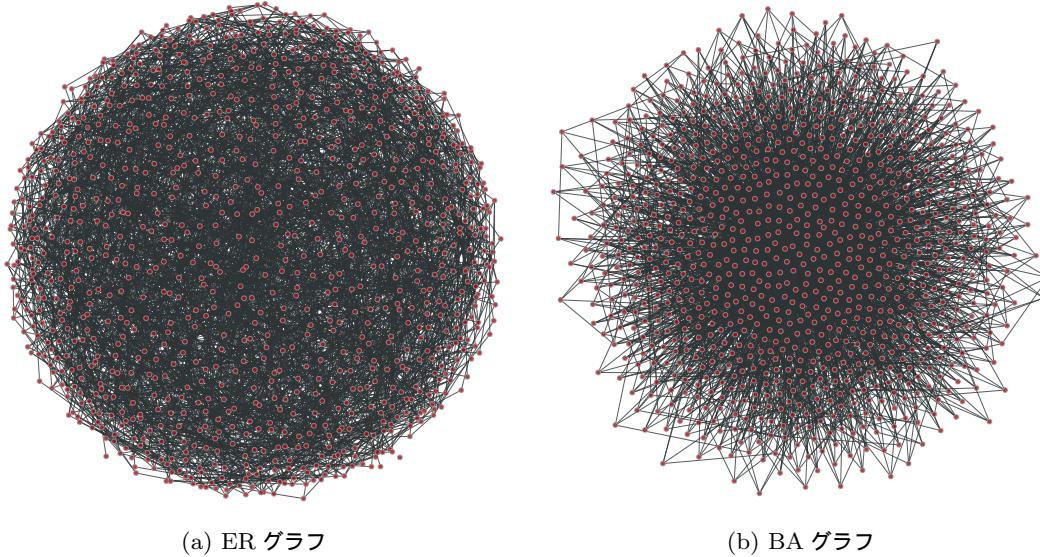


図 6.2: ER グラフと BA グラフの可視化

さてこれらのネットワークでノードを削除した時の削除するノードの割合と最大連結成分の割合の関係を図 6.3 に示す。この結果より、確かに次数分布がべき分布に従う BA グラフでは、次数順にノードを削除した場合に、少ないノードの削除で最大連結成分の大きさが著しく小さくなっていることが確認できる。これは本章の導入で説明したように、BA グラフでは、次数の非常に大きなハブノードがネットワーク全体の連結性を保つ上で重要な役割を果たしているためである。

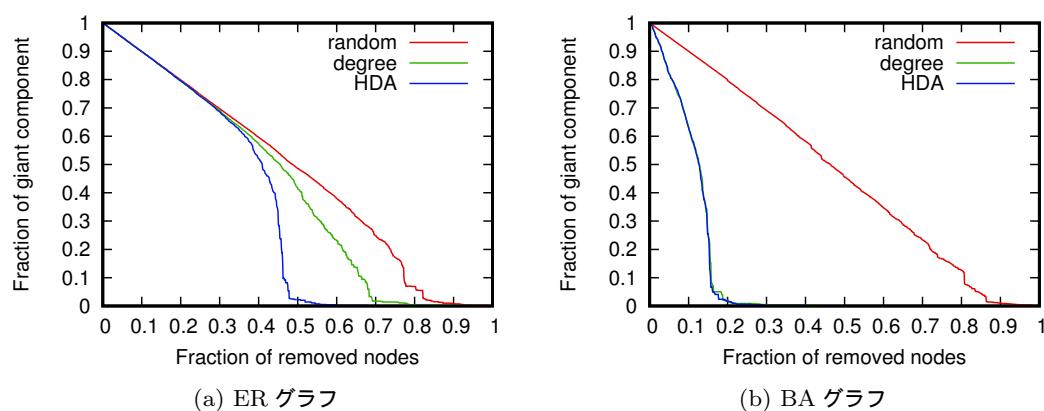


図 6.3: ER グラフと BA グラフにおける削除するノードの割合と最大連結成分の割合の関係

さらに、インターネットのトポロジとして Center for Applied Internet Data Analysis

(CAIDA) の収集したルータレベルトポロジ<sup>\*2</sup> を用いて同様の実験を行う。図 6.4 に次数分布と、ノードを削除した際の最大連結成分の割合を示す。ノード数が 192,244 と非常に大規模であるため、可視化した結果は掲載していない。この結果から、ルータレベルトポロジの次数分布もべき分布に従うことが確認できる。また、BA グラフの結果と同様に、次数順削除によって、ネットワークが急速に分断されていくことも分かる。このことから、ルータレベルトポロジのランダム故障に対する耐性と、攻撃に対する脆弱性が確認できる。

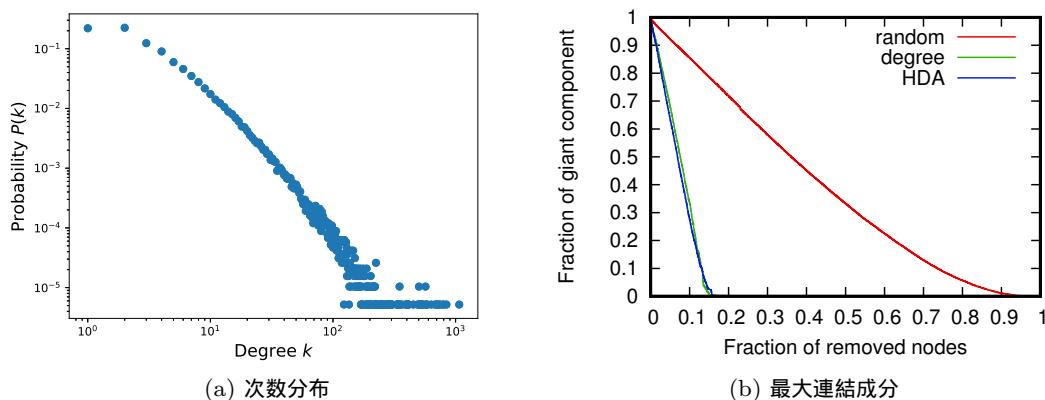


図 6.4: ルータレベルトポロジの次数分布とロバスト性評価の実験結果

本章では、既存研究の発見をなぞる形でネットワークのロバスト性の評価方法を解説した。頑健なネットワークの構成法 [101] や、ネットワークへの攻撃手法 [99] を提案するような研究をする際には、このような手順で提案の良さを評価することができる。

<sup>\*2</sup> <http://networksciencebook.com/translations/en/resources/data.html>

# 7

## ネットワーク分析の実践 2：ネットワーク 上のウィルス拡散

COVID-19 の流行によって、社会におけるウィルス拡散現象やその対策に関心が高まっている。社会におけるウィルス拡散現象を考える上で、ネットワークは重要な役割を果たす。本章では、「人と人の接触頻度を減らすことでの感染がどの程度抑えられるか？」、「コミュニティをまたがる移動を抑制することが、ウィルス拡散を抑制するのにどの程度有効であるか？」などの疑問に答えるためのネットワーク上のウィルス拡散シミュレーションの方法について解説する。

### 7.1 導入

ネットワーク上のウィルスや情報の拡散現象は重要な研究トピックである。特に最近では、COVID-19 のパンデミックの話題の中で、Susceptible–Infected–Recovered (SIR) モデル [102] など感染症の拡散モデルについて目にする機会も多いのではないかと思われる。ウィルスは人ととの接触によって拡散する。古くは、人と人の接触頻度は全ての人の間で均一であり、「世の中の人と人が単位時間あたり平均的に何人と出会う」のような仮定でウィルス拡散のダイナミクスが研究されてきた。これは、ソーシャルネットワークのノード全員が（同じ重みで）接続されているような状況である。一方、本稿でも何度か触れているように、現実のソーシャルネットワークの次数分布はべき則に従い、また、コミュニティ構造という特徴的な構造も有している。このようなソーシャルネットワークの構造がウィルス拡散のダイナミクスに大きく影響することが分かっている。

本章では、ネットワーク上のウィルス拡散のシミュレーションによって、ソーシャル

ネットワークの構造がウィルス拡散にどのような影響を与えるかを調査する。ウィルス拡散のモデルとして代表的なSIRモデルを用いる。ネットワークの構造を変化させた時に、SIRモデルによるウィルス拡散のダイナミクスがどのように変化するかを分析する。

## 7.2 SIR モデル

SIRモデルでは、ウィルス拡散は、ノードの状態の確率的な遷移で表現される。ネットワークを構成するノードは状態S(Susceptible)、I(Infected)、R(Recovered)のうちのいずれかの状態にある。状態Sは健康でウィルスにまだ感染していない状態を表す。状態Iはウィルスに感染し、隣接ノードをウィルスに感染させる状態を表す。状態Rはウィルスから回復し、免疫を獲得した状態を表す。状態Rのノードは二度とウィルスに感染することはなく、また他のノードを感染させることもない。なお、状態RをRemoved(つまり死亡した状態)と解釈する場合もある。

SIRモデルの状態遷移の例を図7.1に示す。離散時間での状態遷移を考える。時刻tにおいて状態Iのノードは確率 $\gamma$ で状態Rに遷移する。状態Iのノードvは、そのノードのそれぞれの状態Sの隣接ノードuについて、確率 $\beta$ で状態Iに遷移させる。 $\beta$ はウィルスの感染力を表すパラメータで感染率と呼ばれる。 $\gamma$ は回復の早さを表すパラメータで回復率と呼ばれる。状態Iのノードがネットワークから存在しなくなった時点で状態遷移が発生しなくなり、収束状態となる。

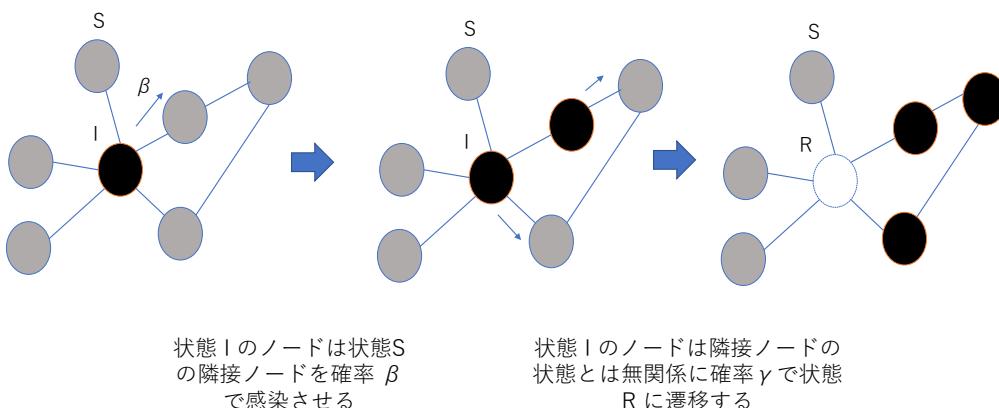


図7.1: SIRモデルの状態遷移

なお、状態S、I、Rを用いて、他のモデルを考えることもできる。治ることのないような病気であれば、SIモデルという状態Rのないものを考える場合もある。ある程度立つと免疫が失なわれるようなタイプであれば、SIRSモデルとして表現できる(状態Rから

S に確率的に遷移する)。免疫を獲得しないようなものであれば、SIS モデル（状態 I から S に遷移する）を用いる場合もある。他にも潜伏期間を表す状態として状態 E (Exposed) を追加した SEIR モデルなども用いられる。(COVID-19 は潜伏状態があるので、SEIR モデルが使われることもある)

### 7.3 シミュレーションプログラム

本節では、ネットワーク上の SIR モデルのシミュレーション実験によって、ネットワーク構造によってウィルスの感染規模がどのように異なるかを分析するためのプログラムについて解説する。確率的な事象を扱うので、乱数を生成し、その乱数の値によってノードの状態を遷移させていく。ある条件（ネットワークの構造や感染率など）で、乱数のシードを変えながら何度もシミュレーションを繰り返すことで、感染規模の平均を求める。これまでの「分析」とは少し毛色が異なるが、SIR モデルのシミュレーションプログラムを実装するのは、そう難しくない。

以下では、SIR モデルのシミュレーションを実装する方法をソースコードを用いて説明する。まず、状態遷移の前の設定部分を以下に示す。

#### SIR モデルの準備

```
1 from numpy.random import *
2 import sys, os, os.path
3 from networkx import *
4
5 # ネットワークの読み込み
6 g=read_edgelist(sys.argv[1])
7
8 # model parameters
9 beta=0.04 # infection rate
10 r=0.2 #recovery rate
11
12 inf={} #感染ノードを管理する辞書
13 # ランダムに選んだノードを1つ状態 I にする
14 i = choice(list(g.nodes()))
15 inf[i]=1
```

ここでは、引数で指定したファイルからネットワークを読み込み、感染率、回復率などのパラメータを設定している。SIR モデルで状態遷移をスタートさせるためには、最初にいずれかのノード（複数でもいい）を状態 I に遷移させる必要がある。ここでは、ランダムに選択したノードを1つ状態 I に遷移させている。次に、肝心の状態遷移の部分を以下

に示す。

### SIR モデルの状態遷移

```
18 max_count = 100
19 num_r=0
20 for time in range(1,max_count):
21     inflist=list(inf.keys())
22     shuffle(inflist)
23     for v in inflist:
24         for nbr in g[v]:
25             if nbr not in inf:
26                 if (random()<beta):
27                     inf[nbr]=1
28             if(random()<r):
29                 del inf[v]
30                 g.remove_node(v)
31                 num_r+=1
32             num_inf=num_r+len(inf)
33             # 時刻、状態 S の数、I の数、R の数、R+I の数
34             print(time,len(g.nodes()),len(inf),num_r,num_inf)
35             if len(inf)==0:
36                 break
```

時刻 1 から max\_count までシミュレーションするという設定である。状態 I のノードがいなくなるのが収束状態なので、収束状態までシミュレーションするのもよい。なお、35~36 行目に、状態 I のノードが存在しなくなった時点で打ち切る処理を入れてあるので、max\_count を十分大きくしておけば、収束状態まで状態遷移を繰り返す。各シミュレーション時刻ごとに、各ノードの確率的な状態遷移を行う。状態が遷移するのは、状態 I のノードと、I のノードに隣接しているノードだけなので、状態 I のノードを inflist という変数に入れて(21行目)、ループを回している(23行目以降)。24~27行目は、ノード v がその隣接ノードを感染させるかどうかチェックする処理である。numpy の random() メソッドによって 0~1 の一様乱数を生成し、それが beta (感染率) より小さければ、v の隣接ノード nbr を状態 I に遷移させる。28~31行目は、状態 I から R への遷移のチェックである。状態 R のノードは今後の状態遷移に影響しないため、ネットワーク g からノードを取り除いてしまっている(30行目)。最後に 34 行目で、状態 S、I、R のノード数を出力している。

これが、ある設定に対する1回のシミュレーションの試行である。通常、ある設定での感染の規模を知りたければ、100回、1000回と何度も試行を繰り替えして平均を求める

いう操作をする。100回なり1000回のforループをpythonのプログラムの中に書いてもよいし、以下のようなシェルスクリプトを用いててもよい。

```
run-sir.sh

net=$1
for i in `seq 1 100`
do
    python3 sir.py $net >>$net.txt
done
```

変数netには、シミュレーション対象のネットワークのファイル名を入れる。for文はsir.pyを100回繰り替えすという意味である。100回分のシミュレーション結果は、そのままテキストファイルに保存しておく。後は

```
$ sh run-sir.sh
```

を実行すると、ネットワークのファイル名.txtにシミュレーション結果が出力される。あとはこれを時刻ごとに平均するなどすれば、求めたい結果が得られる。

## 7.4 ネットワーク構造がウィルス拡散に与える影響

さて、プログラムが準備できたら、さっそくネットワーク構造によってウィルス拡散の規模がどのように異なるかを確認してみよう。まず設定としては、1000人の社会で、1人あたり平均8人の知人がいるとしよう。つまりノード数 $N = 1000$ で、平均次数 $\langle k \rangle = 8$ である。感染率および回復率はそれぞれ仮に $\beta = 0.04$ 、 $\gamma = 0.2$ としよう。時刻0の時点でランダムに選択したノードを1つだけ状態Iに遷移させる。他のノードは全て状態Sである。

まずは、2つの異なる次数分布を持つネットワークで感染の規模を比較する。前章でも登場したERグラフ[100]、およびBAグラフ[7]を用いることとする。ERグラフとBAグラフの次数分布は前節の図6.1の通りである。ERグラフは次数分布が二項分布に従い、BAグラフは次数分布がべき分布に従う。人のネットワークの次数分布はべき分布に従うため、次数分布の観点では、BAグラフの方が人のネットワークらしいということになる。

ERグラフとBAグラフそれぞれで、SIRモデルのシミュレーションを行い、時刻ごとに感染した経験のあるノード数をプロットした結果を図7.2に示す。縦軸は感染した経験のあるノード数なので、状態Iのノード数ではなく、状態I+Rのノード数ということに注意する。

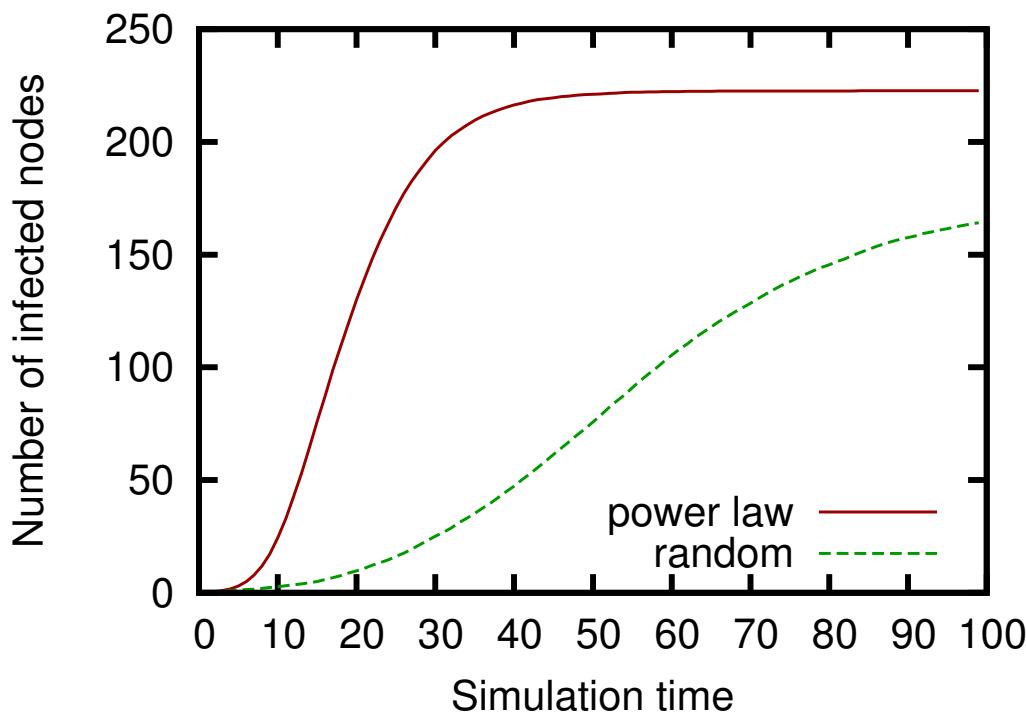


図 7.2: 次数分布の違いによる感染規模の比較

この結果より、ER グラフと比べて、BA グラフの方が感染の規模が大きいことが分かる。同じ感染率で、平均次数が同じであっても、次数の分布によって感染規模は大きく異なるのである。このように、人のネットワークのような次数分布がべき分布に従うようなネットワークでは、ウィルスの拡散速度が非常に速いことが分かっている。この原因の直感的な説明としては、ハブノード（感染症の文脈ではスーパースプレッダーと呼ぶこともある）が存在するということである。ウィルスがハブノードに到達してしまうと、そこから全体に一気に拡散してしまう。また、ハブノードは多くのノードとの距離が近いため、ハブノード自体が感染する確率は他の次数の小さなノードと比べて高い。それに対して、ER グラフのような、8人としかつながっていない人が大多数のネットワークでは、それほど一気にウィルスが拡散することはない。拡散の速度が遅ければ、徐々に回復するノードも増えていき、拡散が収束しやすい。ここから得られる教訓としては、人の社会というのは、そもそもウィルスを大規模に広げやすい構造をしているので、他の国で発生した新種のウィルスでまだそれほど広がっていないというような状況であっても、急速に広まってしまう恐れがあるため、慎重に対策する必要があるということであろう。

次に、平均次数を下げるによって感染規模がどの程度抑えられるかを調査する。「接触を8割減らしましょう」というメッセージが話題となつたが、その接触を減らすことの効果を確認してみよう。ここでは、8割ではなく、半分に減らしてみる。つまり、ネットワークの平均次数を半分にして同様のシミュレーションを実施する。図 7.3 に平均次数が8の BA グラフと平均次数が4の BA グラフにおける感染経験のあるノード数の推移

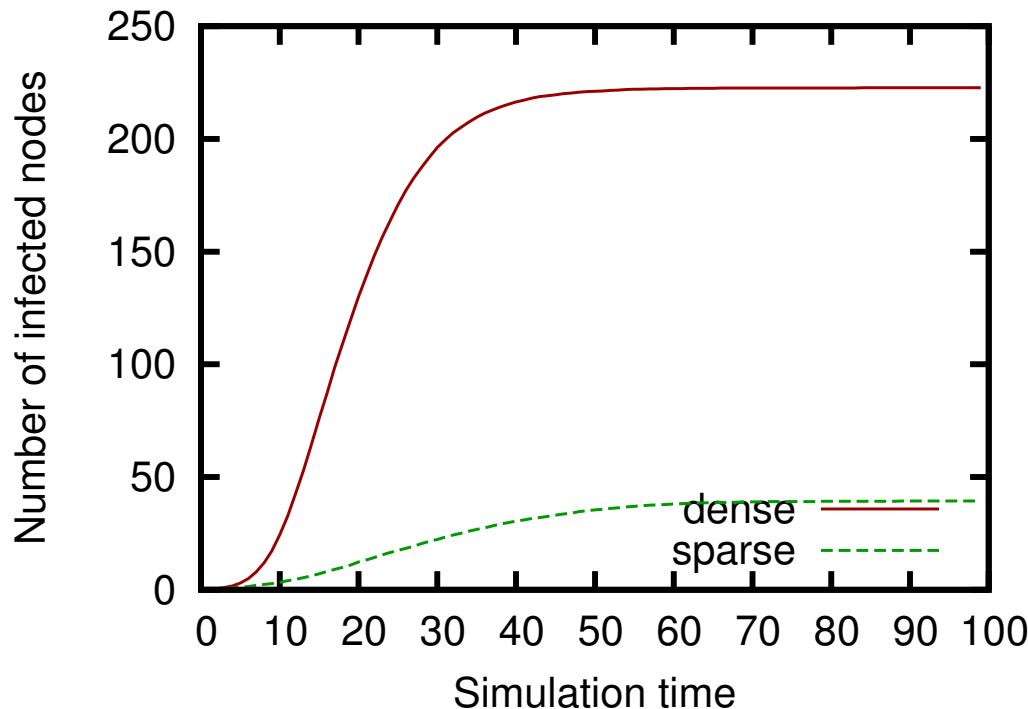


図 7.3: 平均次数の違いによる感染規模の比較

を示す。感染率や回復率は同様で、平均次数のみが異なる。

この結果より、平均次数を半分にすることで、次数分布がべき分布に従うネットワークでも、感染規模を大幅に抑制できていることがわかる。この結果は、ウィルス拡散を抑制するためには、やはり人ととの接触頻度を下げることが重要であることを示している。

最後に、コミュニティ構造の影響を調査する。人のネットワークはコミュニティ構造を有している。実社会では、例えば、沖縄に住んでいる人同士は頻繁に接触するが、沖縄に住んでいる人と東京に住んでいる人の間の接触はそれほど頻繁には発生しない。このような居住地による接触頻度の違いは、ソーシャルネットワーク上では居住地をコミュニティとするコミュニティ構造として表現される。「県をまたぐ移動は控えましょう」という感染症対策のメッセージが出されたこともあるが、これはソーシャルネットワーク上での「コミュニティ間のリンクを減らしましょう」ということを意味する。では、コミュニティ間のリンクを減らすことにどの程度、感染規模を減らす効果があるのか調べてみよう。

図 7.4 にコミュニティ構造を有する 2 つのネットワークを示す。これは、Stochastic Block Model (SBM) [103] と呼ばれるモデルで生成したネットワークである。SBM ではコミュニティ内のリンクの生成確率とコミュニティ間のリンクの生成確率を指定して、ネットワークを生成する。ここでは SBM の詳細な説明は割愛し、生成済みのネットワーク (GitHub に置いている) を使って話を進める。図 7.4 の左側のネットワークと右側のネットワークで、コミュニティ内のリンクの割合は同じであるが、コミュニティ間のリンクの割合だけが異なっている。左側に比べて、右側のネットワークはコミュニティ間リンク

クの割合が半分になっている。全体としては、コミュニティ内リンクの割合が非常に大きいため、平均次数は2つのネットワークでそれほど違わない。密な方で  $\langle k \rangle = 20.28$ 、疎な方で  $\langle k \rangle = 19.46$  である。

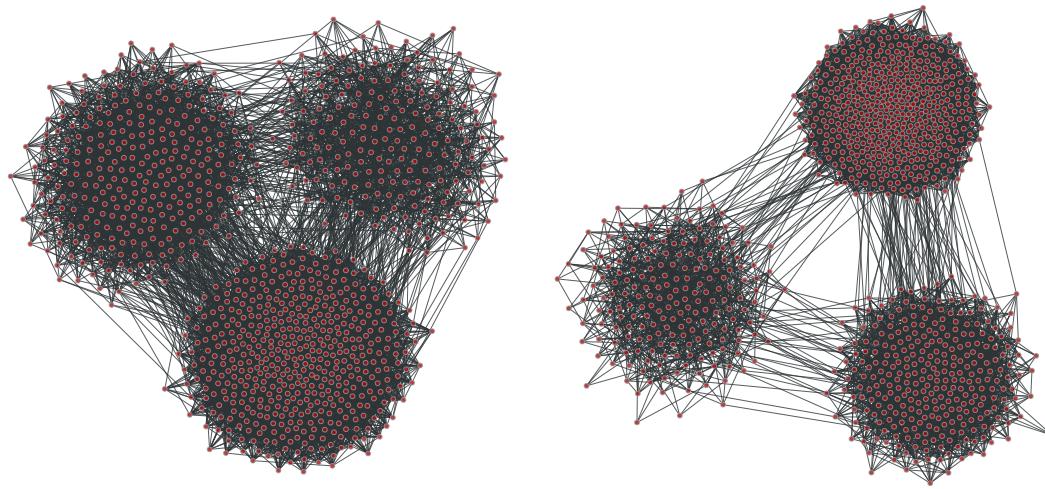
(a) コミュニティ間リンク：密 ( $\langle k \rangle = 20.28$ )(b) コミュニティ間リンク：疎 ( $\langle k \rangle = 19.46$ )

図 7.4: コミュニティ構造を有するネットワークの可視化 (SBM で生成)

図7.5に、コミュニティ間リンクの割合が異なる2つのコミュニティ構造を有するネットワークの感染規模の変化を示す。BA グラフ、ER グラフと比べて平均次数が高いため、感染率  $\beta = 0.02$  としている。回復率は  $\gamma = 0.2$  のままである。感染率が高すぎると、すぐにネットワーク全体にウィルスが拡散してしまい、構造の影響が分かりにくくなるため、このようにしている。

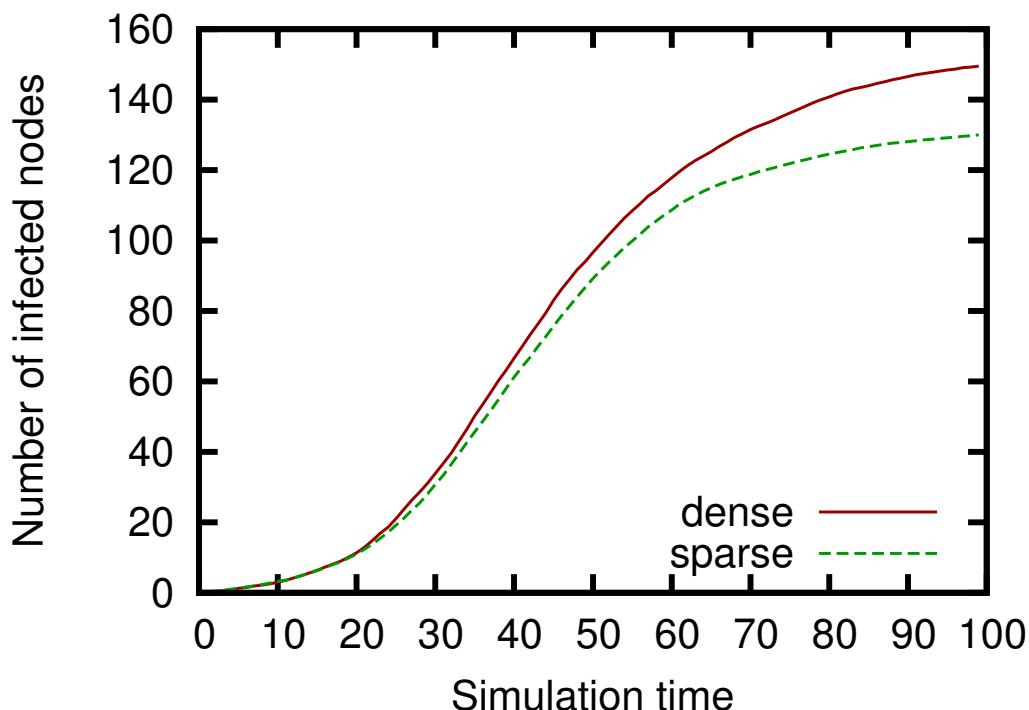


図 7.5: コミュニティ構造間リンクの割合の異なるネットワークでの感染規模の比較

図 7.5 から、平均次数がほとんど同じであっても、コミュニティ間リンクの割合によって感染規模が大きく異なることが分かる。コミュニティ構造の強いネットワークで、感染率がそれほど高くない場合には、コミュニティの中だけでウィルス感染が収束する確率が高い。ところが、もしコミュニティをまたがってウィルスが拡散すると、拡散先のコミュニティの中でも急速にウィルスが広がってしまう。このようなコミュニティ構造の強いネットワーク（人の社会はそれに該当する）では、コミュニティ間のリンクを減らす（接触を抑える）ことも有効な感染対策となり得る。

本章では、ウィルス拡散にネットワーク構造が大きく影響することを簡単なシミュレーションによって確認した。COVID-19 の流行を抑えるために、様々な対策が取られてきたが、その効果を確認するためにも、このような手法は有効である。もちろん、本稿の例はかなり簡略化したものであり、東京の COVID-19 の感染規模をシミュレーションしたいといった具体的な場合には、より精緻なモデル化が必要であろう。ただ、基本となる考え方としては、ここで紹介したものが役立つはずである。本稿で取り扱わなかった話題としては、限られた数の免疫を付与できる時に、どのように免疫を付与すればよいかという研究も存在する [104]。ネットワーク分析はこのような感染症対策の道具としても用いることができる。

# 8

## ネットワーク分析の実践 3：Twitter のエゴネットワーク解析

本章では、代表的なソーシャルメディアである Twitter 上のデータ収集から解析までの一連の流れを解説する。特に、注目するアカウントのフォロワーで構成されるエゴネットワーク (ego network) の構造的な特徴を解析する方法を紹介する。

### 8.1 導入

ネットワーク分析の研究の発展と、ネットワークで表現されるデータが入手可能となつたことは、切っても切り離せない。特にここ 10 年くらい、研究者らはソーシャルメディア、特に Twitter のデータを用いて様々な研究を行なってきた。ここ最近の COVID-19 に関するソーシャルメディア分析の研究だけでも、すごい勢いで発表されている（例えば [105]）<sup>\*1</sup>。ソーシャルメディアのデータを解析する動機は、社会の現在の状態を推定する、さらには将来の動向を予測する、社会現象に関するなんらかの仮説を大規模データで検証するなど様々である [106, 107]。いずれにしても、人と人の関係や行動を大規模かつそこそこ細粒度に、また準リアルタイムに観測可能なプラットフォームとして、ソーシャルメディアは他に類を見ないデータ源であることが、研究者を惹きつけている。ちなみに、研究者がソーシャルメディアの中でも特に Twitter のデータを使う理由のほとんどは、次節で説明する API によってデータの収集が比較的容易なことである。特に Twitter が重要であると考えているわけではない（と思われる）。

\*1 筆者も少しデータを手元に持ってくるくらいはやってみたが、スピード感に着いていけずにやめた

ソーシャルメディア解析には、ツイート自体のテキストの解析やリツイートやハッシュタグ、URLなどの拡散のダイナミクスの解析など、いくつかの観点があるが、本稿では、これまで学んだネットワーク分析に絞って話をする。ソーシャルメディア上のネットワークに限定しても、リツイートのネットワークやメンションのネットワークなどいくつかの分析対象がありえるが<sup>\*2</sup>、本稿では、一番オーソドックスと思われる、フォロー関係のネットワークの解析を扱う。

Twitter のデータの収集は他のソーシャルメディアと比較して容易であるとはいえ、フォロー関係のネットワークのデータを大規模に収集するのは、次節でも説明するが、実はとてもたいへんである。というのも、Twitter API には単位時間あたりのアクセス制限があり、フォロー関係の取得は特に厳しい。15 分間に 15 回までしかアクセスできない。正確ではないが、1 時間で 60 アカウント、1 日で 1440 アカウントのフォロワーを収集するのが精一杯である<sup>\*3</sup>。ということで、大規模な Twitter のフォロー関係のネットワークを得るにはそれなりに時間をかける必要がある。色々とノウハウもあるので、本格的に取り組む際にはその筋の専門家<sup>\*4</sup> に相談するのがよい。

そこで、そこまで大規模にデータを集めなくても可能な分析として、ここでは Twitter アカウントのエゴネットワーク (ego network) の分析というアプローチを紹介する。あるユーザ  $u$  のエゴネットワークとはユーザ  $u$  の周辺のノードで構成されるネットワークのことである（図 8.1）。注目するノードの 1 ホップだけ見るのか 2 ホップまで見るのかなど、どこまでを「周辺」とするかは定義によるが、ここでは注目するノードの 1 ホップの範囲（隣接ノード）を周辺と考える。おそらくこれが、もっともよくある考え方である。つまりユーザ  $u$  のエゴネットワークは、ユーザ  $u$  の隣接ノードと、隣接ノード間のリンクで構成されるネットワークということになる。Twitter の場合はリンクに向きがあるので、「隣接」の方向を明確にしておく必要がある。以降では、ユーザ  $u$  のエゴネットワークを、ユーザ  $u$  をフォローしているユーザ ( $u$  のフォロワー) をノード集合、フォロワー間のリンクをリンク集合とするネットワークとする。ここで、ユーザ  $u$  自身はエゴネットワークには含まれないという扱いにする。一般には、ユーザ  $u$  自身を含むと定義することの方が多いが、後の解析の都合でこのように定義しておく。

<sup>\*2</sup> 他にもハッシュタグの共参照関係、リストに登録した/された、など色々考えられる

<sup>\*3</sup> 1 回のアクセスで取得できるフォロワーの ID の数は 5,000 までなので、5,000 以上フォロワーがいる場合、1 アカウントのフォロワーのリストを得るだけでも何度も API にアクセスする必要がある

<sup>\*4</sup> はたして、日本に何人いるのか。ちなみに筆者自身はデータ収集の専門家ではないという自己認識である。

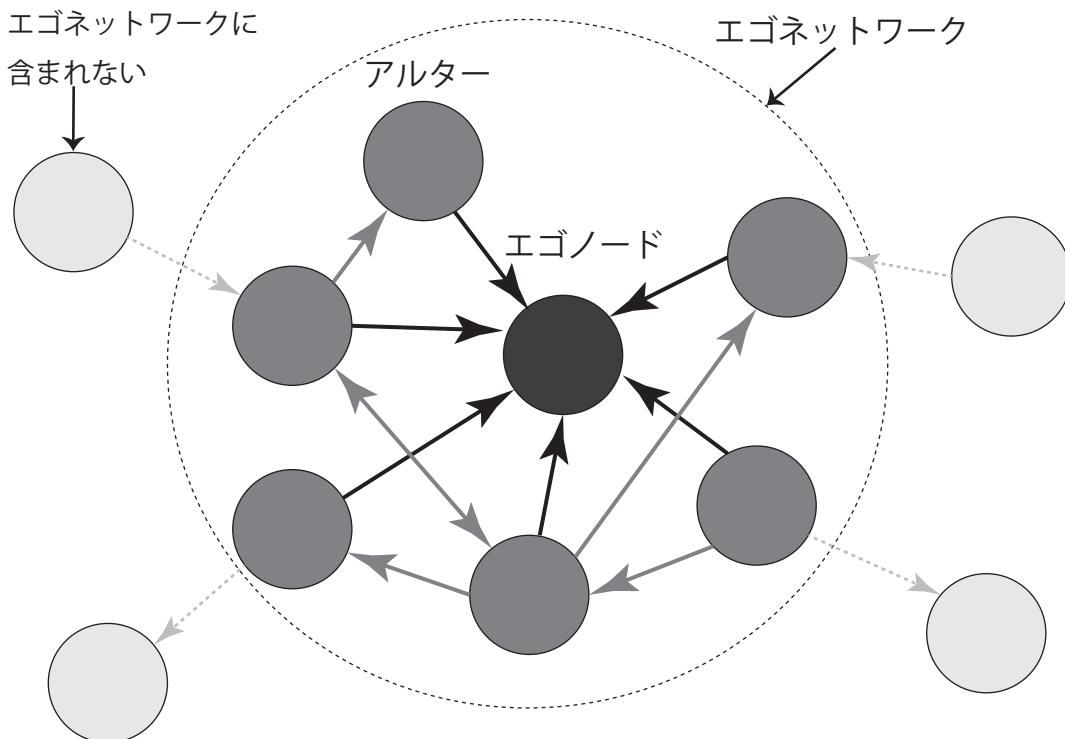


図 8.1: エゴネットワークの例

## 8.2 Twitter API を用いたソーシャルネットワークの収集

API を用いてデータを収集するというのは、慣れていないとなかなかハードルが高いことのように感じられるかもしれないが<sup>\*5</sup>、Twitter API をラッピングした便利なモジュールが色々と存在するため、やってみるとそれほど難しくもない。

さて、まずは準備として、Twitter API にアクセスするためのキーとアクセストークンを取得する。前提として、Twitter アカウントが必要となるので、Twitter のアカウントを持っていない場合は、Twitter への登録がまずは必要である。これは、メールアドレスさえあればすぐに完了する。Twitter アカウントが用意できたら、ログインした状態で Developer ページ (<https://developer.twitter.com/en/portal/dashboard>) にアクセスし、登録する。この後の手順はしばしば変更になり、この資料の中で解説するのはやや煩雑であるが、Web 上に情報がたくさんあるので、そちらを参照していただきたい。おおまかな流れとしては、利用目的を入力したり、メールで認証したりの手順を経て、キーとアクセストークンが発行される。

キーとアクセストークンが入手できたという前提で、Twitter のアカウント名を指定して、そのアカウントのフォロワー（あるいはフォローしているユーザ）を取得する方法を紹介する。Python から Twitter API を利用するための色々なモジュールが存在するが、

<sup>\*5</sup> 実際、大規模にやろうとすると難しいのだが

ここでは tweepy<sup>\*6</sup> を用いた方法を紹介する。pip install (pip3 install) でインストールできる。指定したユーザのフォロワーのリストを出力するサンプルプログラムを以下に示す。

### フォロワーの収集

```
1 import tweepy
2 import sys
3
4 # ここに自分のキーとトークンを入れる
5 consumer_key, consumer_secret = "", ""
6 access_token_key, access_token_secret = "", ""
7
8 name=str(sys.argv[1]) # Twitter アカウント名
9 path_follower="followers/"+name+".txt"#出力先
10
11 cursor = -1
12 with open(path_follower, mode='w') as f:
13     while cursor != 0:
14         auth = tweepy.OAuthHandler(consumer_key,
15                                     consumer_secret)
16         auth.set_access_token(access_token_key,
17                               access_token_secret)
18         api = tweepy.API(auth, wait_on_rate_limit=True)
19         itr = tweepy.Cursor(api.followers_ids, id=name,
20                             cursor=cursor).pages()
21         try:
22             for follower_id in itr.next():
23                 try:
24                     print(follower_id)
25                     f.write(str(follower_id)+'\n')
26                 except tweepy.error.TweepError as e:
27                     print(e.reason)
28             except ConnectionError as e:
29                 print(e.reason)
30             cursor = itr.next_cursor
```

\*6 <https://www.tweepy.org/>

プログラム中の、5、6行目で、発行したキーとトークンを入力する必要がある。出力結果はカレントディレクトリの `followers` というディレクトリに保存される。事前に `followers` という名前のディレクトリを作つておく。13行目以降が、APIにアクセスしてフォロワーの情報を取得する部分である。一回のアクセスでは、取得できるフォロワーのIDの数が決まっているため、フォロワーの多いアカウントの場合、何度もアクセスする必要がある。そのために、`while`文でループを回している。

また、エゴネットワークを構築するためには、注目するユーザのフォロワー間の関係を取得する必要がある。ユーザ  $u$  のフォロワーのリストが得られたとして、さらにそのフォロワーそれぞれのフォローしているユーザ（フォロイー）あるいは、フォロワーのリストが必要となる。エゴネットワークを構築する目的であれば、フォロワーを取得しても、フォロイーを取得してもどちらでもよいが、ここでは、フォロイーを取得する方法を採用する。フォロワー数の非常に多いインフルエンサーのアカウントが収集対象に含まれていた時に、フォロワーのリストの取得に非常に時間がかかることが懸念されるためである。フォロワー数は100万人ということもありえるが、100万人をフォローするということは考えにくいため、フォロイーのリストを取得する。フォロイーを取得するプログラムは、フォロワーを取得する場合とほぼ同じであり、上記プログラムの17行目 `api.followers_ids` の部分を `api.friends_ids` とするだけである。また、結果の出力先は、必要に応じて変更する。

注目するアカウントが決まつたら、フォロワーを取得し、さらにそのフォロワーのフォロイーを取得することを繰り替えせばよい。たとえばこんな感じのシェルスクリプトを書けばよいであろう。

```
id.txt に書かれたユーザ ID を順にクローリングするスクリプト

while read line
do
    python3 get_friends.py $line
done<id.txt
```

### 8.3 Twitter アカウントのエゴネットワークの特徴分析

ここではいくつかのTwitterアカウントのエゴネットワークの構造的な特徴を調査する。研究でTwitterデータの解析をする際には、まずどのような問い合わせるかが重要であるが、ここでは練習問題と思って、いくつかのアカウントの構造的な特徴を探索的に眺めてみることにする。

対象としては、電子情報通信学会のアカウント（ieice\_EIC）と、MIKAに協賛している学会のアカウントとして、電気学会（IEEJ\_denk）、機械学会（JSME\_Mech）、および映像情報メディア学会（ITE\_or\_jp）を用いた。これらの間で何か特徴的な違いや類似点は見

られるかを探索的に調査する。まずは、各アカウントのフォロワーを取得した。その後、そのフォロワーのリストを入力として、それぞれのフォロワーのフォロイーを取得した。アカウントが非公開であったり、一時的なエラーで全てのフォロワーのフォロイーを取得できたわけではないが、ここでは原因の特定や再取得を試みるということまではしていない。得られたフォロイーのリストからエゴネットワークを構築した。収集したデータから、エゴネットワークのエッジリストを出力するプログラムを以下に示す。

### エゴネットワークの構築

```
1 import sys,os
2 import glob
3
4
5 def each_line(filename):
6     with open(filename, encoding='utf-8') as f:
7         for line in f:
8             yield line.rstrip('\r\n')
9
10 ego=sys.argv[1]
11 target={}
12
13 for line in each_line(ego):
14     target[line]=1
15
16 files = glob.glob("following/*")
17 for file in files:
18     user =os.path.basename(file)
19     user=os.path.splitext(user)[0]
20     if user in target:
21         for line in each_line(file):
22             if line in target:
23                 print (user,line)
```

入力として、注目するアカウントのフォロワーのIDが1行ずつ書かれたファイルを与える。注目するアカウントの各フォロワー $u$ のフォロイーのリストは、`following/u.txt`というパスに存在することを仮定している。

このプログラムで出力したエゴネットワークに対して、これまでに学んだネットワーク分析の手法を適用する。まずは、可視化した結果を図8.2に示す。なお、孤立ノード（リンクを1本も持たないノード）は含まれていない。この結果より、どのアカウントに

おいても、エゴネットワークは基本的に1つの密集したコアと周辺ノードで構成されるcore-periphery構造を有しているように見える。特に電気学会は全体的にリンク密度が高い傾向にあり、機械学会はコアと周辺のメリハリが強い（周辺ノードが多い）ように見える。

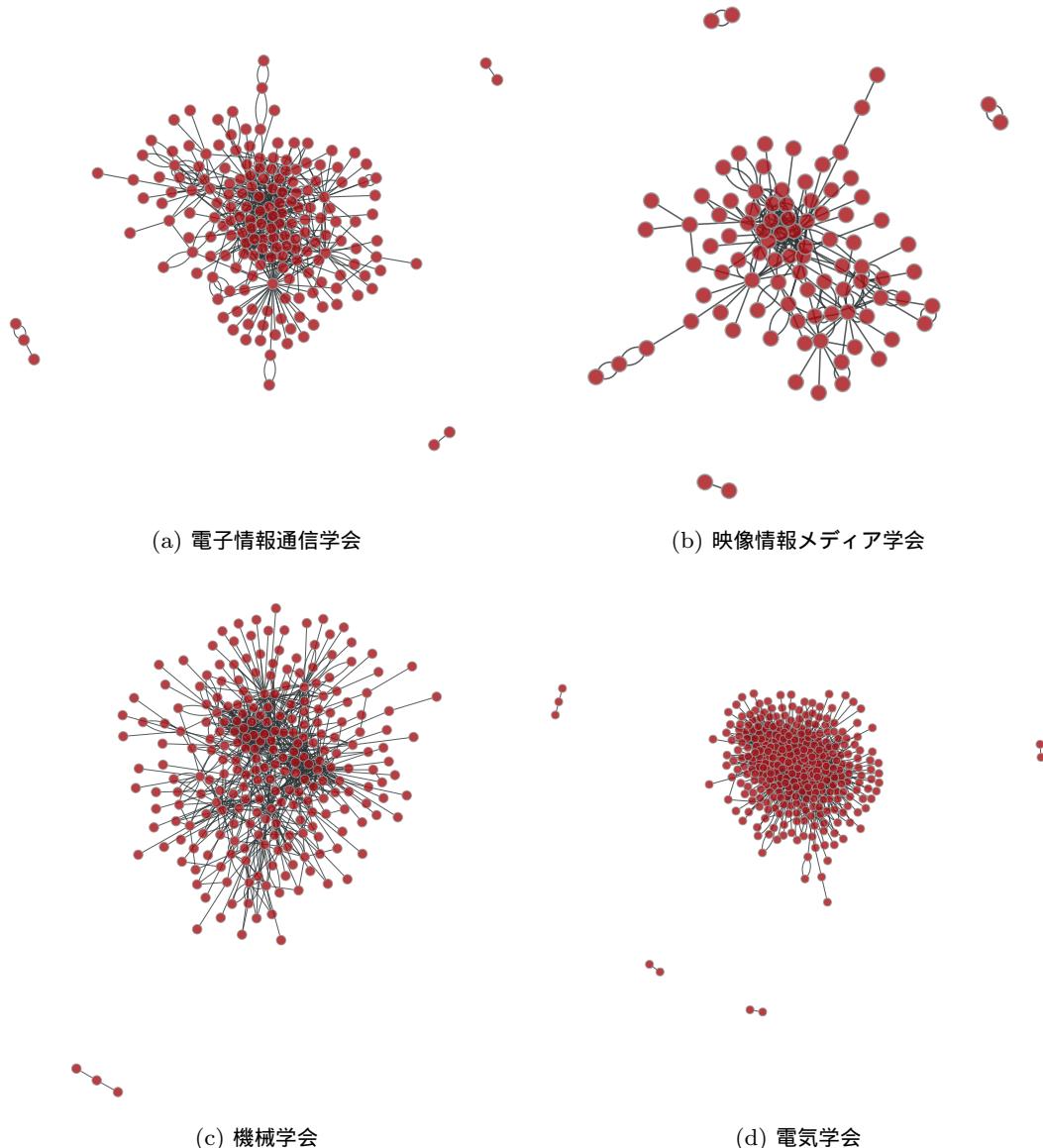


図8.2: MIKA主催・共催学会のTwitterアカウントのエゴネットワーク構造の可視化

次に、次数分布を図8.3に、エゴネットワークの特徴量を表8.1に示す。最短経路長および直径は最大連結成分に対して計算した。コミュニティ数およびモジュラリティは、最大連結成分に対して、CNM法でコミュニティを抽出した時のコミュニティ数とモジュラリティを表している。次数分布の形状としては、おおむねどれも似たような形状である。べき分布に近いと考えてよさそうである。特徴量のデータからも、4つのネットワークが

概ね近い特徴を有していることが確認できる。特徴的なのは電気学会のネットワークで、可視化した結果からも確認できたように密度(平均次数)が高く、クラスタ性も強い傾向が確認できる。平均経路長はどのネットワークも短かく、似たような値であるが、電気学会は他と比べてもやや短めである。機械学会は、他と比べてややクラスタリング係数が高い。可視化した結果でも、周辺ノードが多く見られたので、その影響であると考えられる。

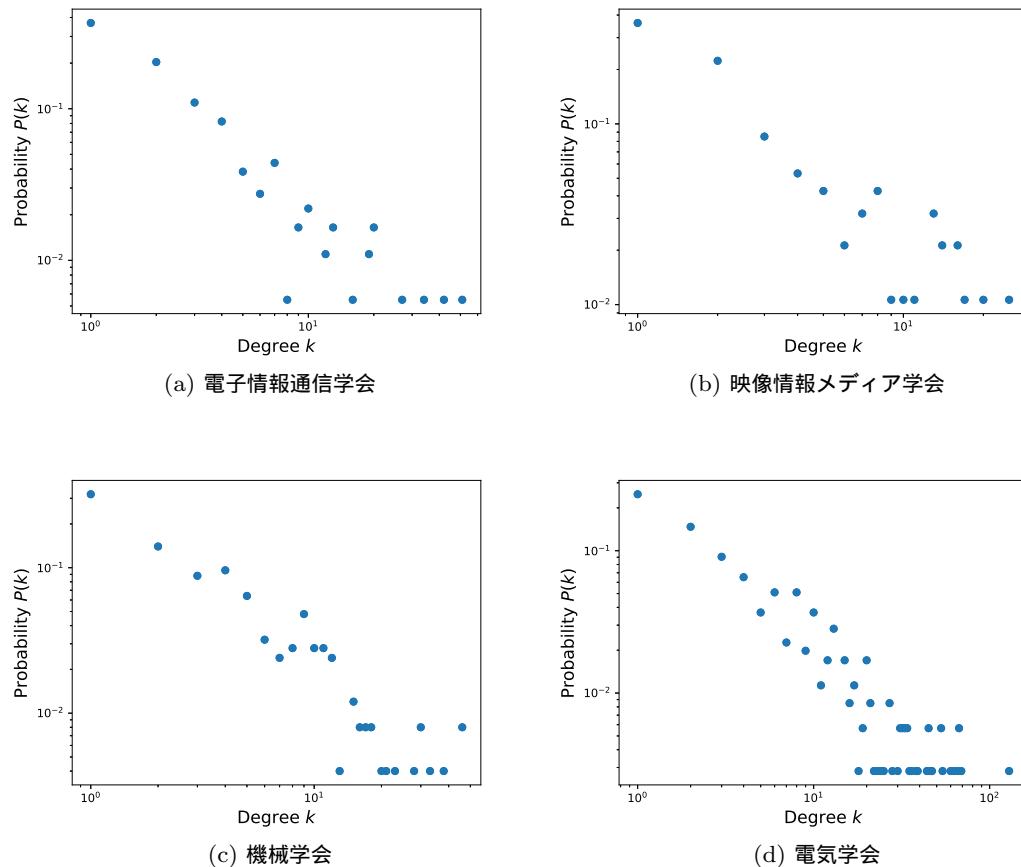


図 8.3: MIKA 主催・共催学会の Twitter アカウントのエゴネットワークの次数分布

表8.1: MIKA関連アカウントのネットワークの特徴量

	信学会	映メ	機械学会	電気学会	4学会マージ
ノード数	182	94	250	353	1069
リンク数	537	258	812	2168	5746
平均次数	4.3	4.2	5.36	9.02	8.45
クラスタリング係数	0.26	0.27	0.20	0.40	0.344
連結成分の数	4	4	2	5	4
最大連結成分の割合	0.96	0.94	0.99	0.97	0.99
最短経路長	3.12	3.26	3.32	2.8	2.8
直径	7	9	8	7	7
コミュニティ数	8	5	7	10	12
モジュラリティ	0.48	0.45	0.51	0.43	0.46

以上より、今回の調査対象の4学会の構造はそこそこ近いことが分かった。相対的には電気学会が比較的密につながっており結束性が強そうである。一方機械学会は相対的にクラスタ性が低く、わりと広い分野の人が学会のアカウントをフォローしているのかもしれない。余談であるが、情報分野の関連学会である、情報処理学会のフォロワー数は4,500程度、人工知能学会で1万程度である。学会の扱う分野の影響もあると考えられるが、今回の調査対象の4学会は、関連学会の中ではTwitter上の情報発信の影響力は相対的に小さそうである。

さて、最後に、4学会のエゴネットワークを全てまとめて可視化したものを図8.4に示す。特徴量は表8.1に含めている。4学会のノード数を足した値よりもマージした場合の方がノード数が多くなっているのは、孤立ノードを除いていることに起因している。学会ごとにばらばらになるかと思っていたが、以外とつながっていることが分かった。やはり研究界隈もスモールワールドで、他学会であっても、知人を辿っていくと、意外とつながるものである。



図 8.4: 信学会、映メ、機械学会、電気学会のいずれかをフォローしているアカウントのネットワーク（カラー版ではノードの色はコミュニティを表す）

# 9

## ネットワーク分析の実践 4：研究者ネットワークの解析

論文の共著関係や引用関係のネットワークの解析はソーシャルネットワーク分析のよくある題材である。本章では、MIKA に協賛している電子情報通信学会の各研専で発表された技術研究報告の著者情報から、MIKA 関連研専の研究者ネットワーク（共著関係ネットワーク）を構築する。ネットワークを可視化するとともに、コミュニティ構造や中心人物の特定を通じて、信学会における分野横断の状況について議論する。

### 9.1 信学会 MIKA 関係研専研究者ネットワークの構築

論文と論文の著者の関係は図 9.1 に示すような論文と論文の著者をノードとし、論文とそれを執筆した著者の間をリンクで結んだ二部ネットワークとして表現できる。これを論文の著者ノードに関して 1-mode network projection してやると、論文の著者間の関係を表す共著関係ネットワークが得られる。共著関係ネットワークでは同じ論文を執筆した著者同士にリンクが生成されている。共著で何回論文を書いたか、また 1 本の論文に何人著者がいるか、などによって重みを付けるやり方も考えられる [22] が、ここでは簡単のために、一度でも共著で論文を書いていれば、リンクで結ぶということにしよう。同じラボのメンバ同士でしか共著論文を書かない場合は、そのラボのメンバが孤立した島（連結成分）として共著ネットワークに現われる。他のラボのメンバとも共著で論文を書くと、他の島ともつながることとなり、大きな連結成分が構成される。このような研究者ネットワークを解析することで、どのくらいラボや組織をまたがった共同研究が行われているか、研究分野間でどのくらい交流があるか、研究分野の橋渡しをしているようなキーパー

ソンとなる研究者は誰か、といったことが分かると期待される。

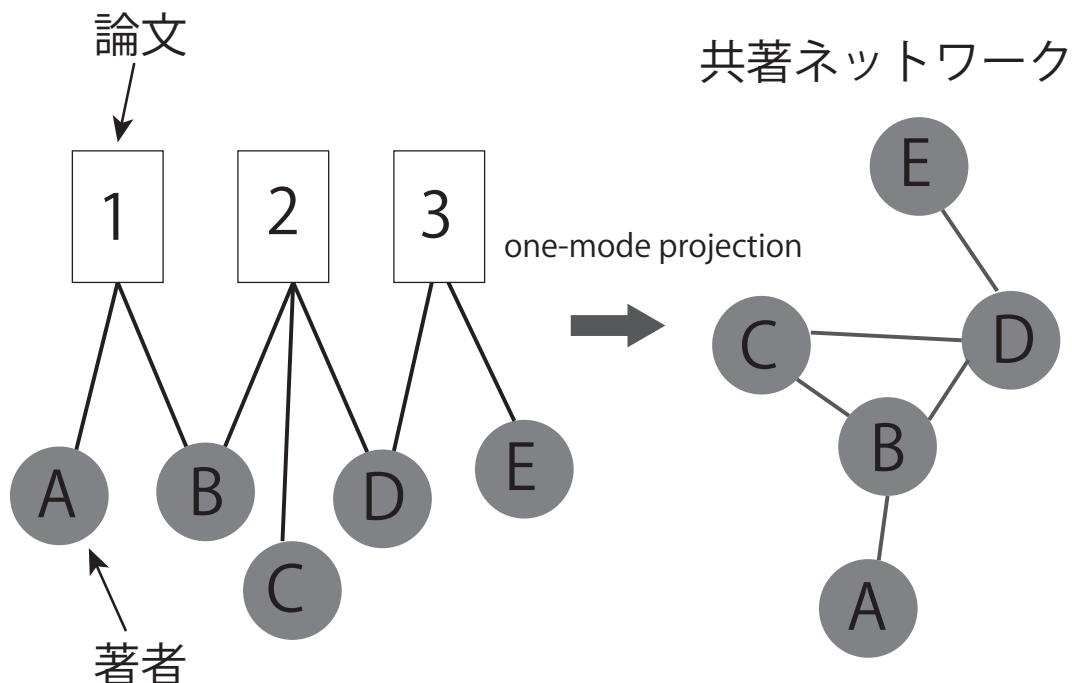


図 9.1: 論文の共著関係ネットワークのイメージ

さて、MIKA<sup>\*1</sup> に関する研究会の共著関係ネットワークを構築するために、各研究会で発表された技術研究報告の著者情報を収集する。Twitter とは異なり、API が公開されていたりはしないので、みなさんおなじみの研究会システム<sup>\*2</sup> のプログラムのページをスクレイピングすることで、各研究会の発表情報を収集する。公開情報であるので、システムにあまり負荷をかけないよう節度を守ってスクレイピングすればデータ収集自体は問題ないと考えているが、研究会プログラムの収集にあたって、もし何か規約上問題があれば、ご教示いただきたい。研究会活動の活性化に資するような分析が目的であるので、あまりうるさく言われることはないないと信じている。信学会の研究会システムに特化しているのであまり需要はないかもしれないが、研究会名(略称)と年度(西暦下2桁)を指定して、各発表の著者情報を出力するプログラムを以下に示す。

#### 研究会プログラムのスクレイピング

```

1 import requests
2 from bs4 import BeautifulSoup
3 import re

```

<sup>\*1</sup> MIKA とは電子情報通信学会の通信ソサイエティにおける分野横断を指向した新たなタイプの研究会である。

<sup>\*2</sup> <https://www.ieice.org/ken/program/index.php?layout=&lang=>

```
4 import sys
5 import time
6
7 year=int(sys.argv[1])
8 year=year-2
9 kensen=sys.argv[2]
10
11 #プログラム一覧の URL
12 url="https://www.ieice.org/ken/program/index.php?instsoc=
IEICE-B&tgid=IEICE-"+str(kensen)+"&year="+str(year)+"&region=0
&sch1=1&schkey=&pnum=0&psize=2&psort=0&layout=&lang=&term=&pskey
=&ps1=1&ps2=1&ps3=1&ps4=1&ps5=1&search_mode="
13
14
15 base=requests.get(url)
16
17
18 soup = BeautifulSoup(base.content, "html.parser")
19
20 elems=soup.find_all('a')
21 for i in elems:
22     if(i.text == "開催プログラム"):
23         url=i.get("href") #開催プログラムを取得
24
25         time.sleep(10) # 負荷をかけないよう 1 アクセスで 10 秒ス
リーブ
26         r = requests.get(url)
27
28         page = BeautifulSoup(r.content, "html.parser")
29
30         elems=page.find_all('td',{"style":"max-width:1000px;"})
31
32         for elem in elems:
33             list=[]
34             #著者名のリストを取得
```

```
35         for i in elem.find_all(href=re.compile("AUTHOR"
36             list.append(i.text)
37             print(", ".join(list))
```

以下のように実行すれば、2020年度のCQ研究会の著者リストが得られる。

```
$ python3 scraper.py 20 CQ
```

上記のプログラムを用いて、MIKAに協賛している研究会として、A・P、SANE、CQ、SR、SRW、CS、NS、MICT、RCS、WPT、RCC、WBS、CCS研究会の発表の著者リストを取得した。年度としては2019年度と2020年度の2年間のデータを収集した。最近の状態に興味があるが、2020年度はほとんどオンラインで、研究者間のコラボレーションも少なかったかもしれないと考え、2019年度のデータと合わせて用いることとした。経時的な変化を見るのもおもしろいが、とりあえず2019年度と2020年を合わせた1つのスナップショットを解析することにする。

収集した各発表の著者のリストから、ネットワークを構築する。同一の報告の著者にリンクを生成してやればいい。以下に発表リストから共著ネットワークのエッジリストを出力するプログラムを示す。

#### 共著ネットワークの構築

```
1 import sys,os
2 import networkx as nx
3
4 #1 行ずつ読んで改行を除去
5 def each_line(filename):
6     with open(filename, encoding='utf-8') as f:
7         for line in f:
8             yield line.rstrip('\r\n')
9
10 list=sys.argv[1] #入力ファイル
11
12
13 g=nx.Graph()
14 for line in each_line(list):
```

```
15     authors=line.split(',')
16     while len(authors)>0:
17         u=authors.pop(0)
18         for v in authors:
19             g.add_edge(u,v)
20
21     for e in g.edges():
22         print(str(e[0])+" "+str(e[1]))
```

収集した発表リストを全研専分まとめて list.txt に保存しておいたとすると、以下のようにすれば、共著ネットワークが出力できる。

```
$ python3 construct_net.py list.txt
```

## 9.2 信学会 MIKA 関係研専研究者ネットワークの分析

構築した 2019～2020 年度の MIKA 関連研専の共著ネットワークを分析する。ここで、一度しか発表していない学生などを含めるとネットワークを可視化した際の視認性が悪かったので、以降では発表回数が 1 回だけの著者は除いた場合の結果を報告する。表 9.1 にネットワークの基本的な統計量を、図 9.2 に次数分布を、図 9.3 に最大連結成分を可視化した結果を示す。紙面上では可視化した結果を細かく解釈することは難しいと思われるが、チュートリアル当日には、もう少し見やすいものをお見せできる予定である。最短経路長および直径は最大連結成分に対して計算した。コミュニティ数およびモジュラリティは、最大連結成分に対して、CNM 法でコミュニティを抽出した時のコミュニティ数とモジュラリティを表している。

表9.1: MIKA関連研専の共著ネットワークの特徴量

特徴量	値
ノード数	2446
リンク数	6426
平均次数	5.25
クラスタリング係数	0.73
連結成分の数	242
最大連結成分の割合	0.57
最短経路長	6.34
直径	17
コミュニティ数	31
モジュラリティ	0.82

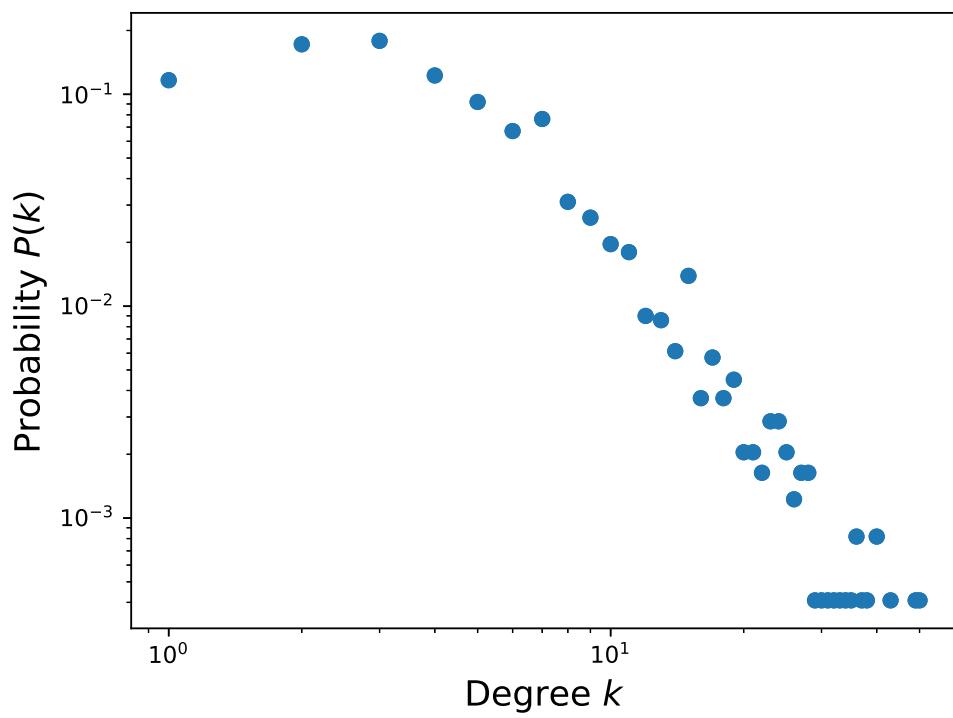


図9.2: MIKA関連研専の共著関係ネットワークの次数分布

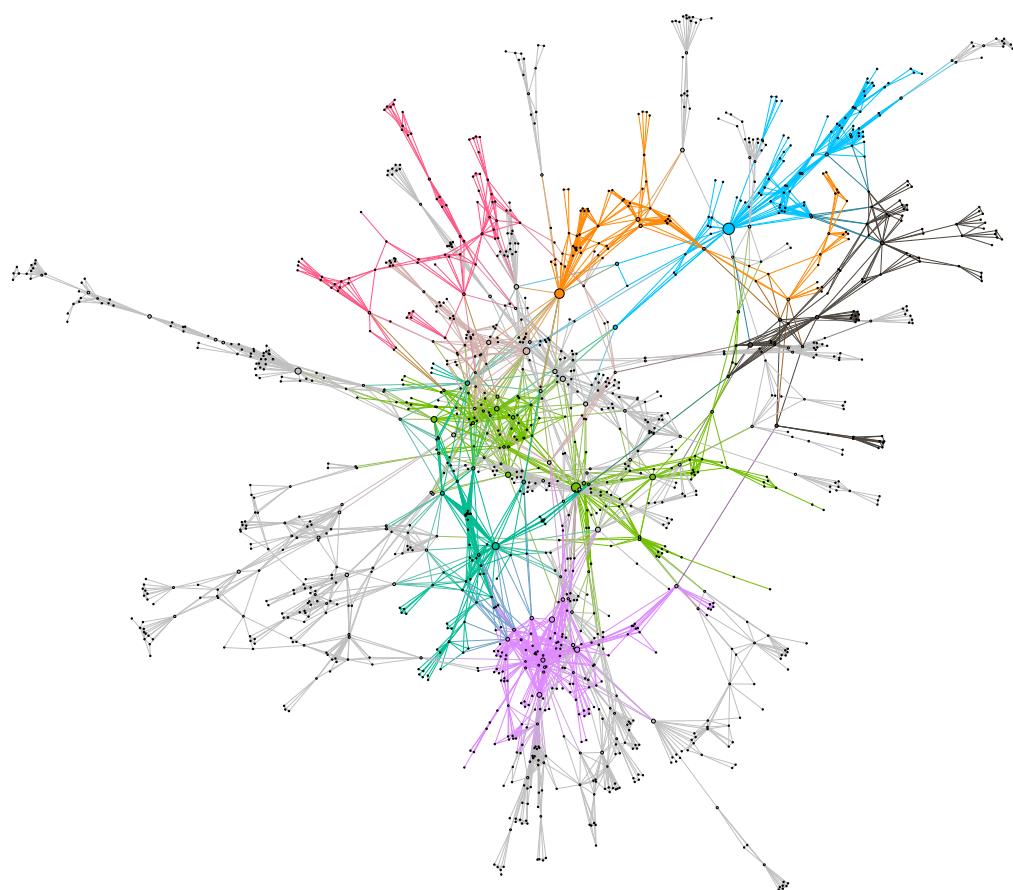


図 9.3: MIKA 関連研専の共著関係ネットワークの可視化（最大連結成分のみ、ノードの大きさは媒介中心性を、カラー版ではノードの色が抽出されたコミュニティを表す）

表 9.1 より、ネットワークは多くの連結成分に分断されているということが分かる。た

だし、これは必ずしも、研究分野によってネットワークが分断されていることを意味しない。図9.4に最大連結成分を除いた連結成分の大きさの分布を示す。この結果から分かるように、最大連結成分以外の連結成分はほとんどサイズが10以下の小さなものである。他の研究室や組織とあまりコラボレーションしない研究グループが一定数存在していて、これらの連結成分に対応すると考えられる。したがって、これらのグループを除けば、MIKA関連の研究者は最大連結成分の中でつながっていると言えそうである。

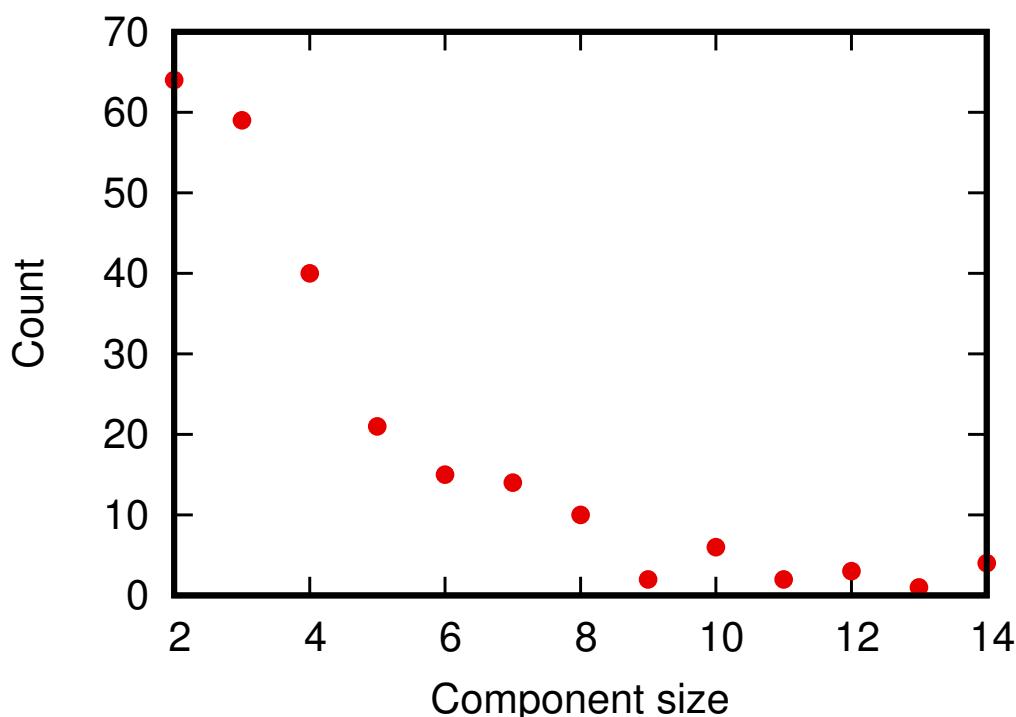


図9.4: MIKA関連研専の共著関係ネットワークの連結成分のサイズの分布

では、可視化した図(図9.3)から、最大連結成分においてどのくらい密なネットワークが構築されているかを確認すると、それほど密にはなっておらずどちらかと言えばスパースに見える。ただし、平均経路長は6.34でちょうど、「6次の隔たり」の6に近い値であり、そこそこ小さな値と言えそうである(直径はこのくらいの規模にしてはやや大きめではあるが)。この小さな平均経路長は、多くの研究者とコラボレーションしているハブとなるノードや、コミュニティ間の橋渡しをするブリッジとなるノードの存在によって達成されていると考えられる。また、可視化した結果からもある程度判断できるが、コミュニティ抽出した結果のモジュラリティが高く、かなり強いコミュニティ構造を持っていることが分かる。複数の研専の発表データを統合してネットワークを構築しているので、コミュニティ構造が強いのはある意味あたりまえではあるが、研専のカバーする研究分野に対応したコミュニティが形成されていることが、この結果からも分かる。整理すると、以下のことが観察される。

- 研専や研究分野を単位とすると思われる密なコミュニティに分かれている。
- 少数のハブやブリッジによって、ネットワーク全体としてはゆるくつながっている。
- ハブやブリッジの存在によって、直接関係ない分野の研究者同士も以外と小さなホップ数でつながっている。

ソーシャルネットワークの研究分野では、古くから「弱い紐帯の強さ」や「weak tie 仮説」という名前で、コミュニティ内の強いつながり（強い紐帯）だけでなく、コミュニティ間の弱いつながり（弱い紐帯）も重要であるということが指摘されている [108]。例えば、転職は弱い紐帯を通じた紹介の方がうまくいく、であるとか、革新的なアイディアは弱い紐帯を通じてもたらされる、といったことが指摘されている。MIKA に関連する分野でも、強い紐帯は各研専の研究会で強化していき、弱い紐帯は MIKA など分野横断型の研究会で維持していくと、分野全体として良いネットワークに発展していくのではないかと思う。

さて、最後にこの分野のキーパーソンが誰であるのか、中心性指標の観点から調査してみよう。例えば、少し分野の違う研究者に招待講演を依頼する場合や、共催のイベントを企画する場合など、分野の橋渡しのできるような研究者を特定できれば有用であろう。同じ分野であれば、誰がどんな研究をしていてということは把握できると思われるが、少し遠い分野のことが知りたい場合、ネットワーク分析で、中心的なノードを抽出してみるというのも有用な方法であると考える。構築したネットワークで媒介中心性をより次数中心性を計算し、その上位 10 ノードを求めた。表 9.2 に中心性上位 10 位以内の研究者の名前（敬称略）を示す。「あの先生の名前がない！」ということもあるかと思われるが、あくまでネットワーク構造上の中心を求めており、コミュニティや研究への貢献度とは必ずしも一致しないことに注意していただきたい。

表 9.2: MIKA 関連研専のネットワークにおける中心性の高い研究者

順位	次数中心性	媒介中心性
1	児島史秀 (NICT)	中尾彰宏 (東大)
2	藤井威生 (電通大)	岡本英二 (名工大)
3	田久 修 (信州大)	西森健太郎 (新潟大)
4	西森健太郎 (新潟大)	児島史秀 (NICT)
5	中尾彰宏 (東大)	岸山祥久 (NTT ドコモ)
6	須山 聰 (NTT ドコモ)	米田尚史 (三菱電機)
7	岡本英二 (名工大)	松村善洋 (JR 東海)
8	高田潤一 (東工大)	平栗健史 (日本工大)
9	岸 洋司 (KDDI 総合研究所)	今井哲朗 (長崎大)
10	米田尚史 (三菱電機)	藤井威生 (電通大)

表9.2には、各分野のアクティブな方の名前が挙がっていることが確認できる<sup>\*3</sup>。次数中心性が高いというのは、色々な人と共同研究をしているということ、媒介中心性が高いということは、コミュニティをまたがるような位置にいることを表している。次数が高ければ必然的に媒介中心性も高くなりやすいので、2つのランキングはある程度重なっている。他分野とのつながりを強化する際など、この名前のリストの方々が頼りになるかもしれない<sup>\*4</sup>。

<sup>\*3</sup> MIKA 関係者では、西森副委員長、平栗幹事がランクイン！

<sup>\*4</sup> ちなみに筆者は次数のランキングが1453位、媒介中心性で615位であり、全然頼りにならないと思われる

### コラム：学際研究の楽しさ難しさ

本文中でも何度か触れているように、ネットワーク科学は学際的で新しい研究分野である。Watts と Strogatz のスモールワードの論文 [2] が 1998 年、Barabási のスケールフリーの論文 [7] が 1999 年、スケールフリーネットワークのロバスト性の論文 [9] が 2000 年で、このあたりからネットワークを題材とした研究が色々な分野で盛り上がり始めた。筆者が修士課程で研究を始めた 2007 年には既に通信分野にもこの盛り上がりの波が来ており、Waxman のモデル [109] ではなく、Watts や Barabási のモデルを評価に使うのが珍しくなくなってきた頃かと思う。当時既にネットワーク科学の研究自体は活発であったが、まだ今よりも少し混沌とした状況だったように思う。

これは今でもそうかもしれないが、ネットワーク科学の研究者には元々の専門分野（物理、生物、社会科学、計算機科学など）がある。（たぶん私くらいの世代がいきなりネットワーク科学の研究をやり始めた最初の世代ではなかろうか？）なので、各研究者の元々の専門分野でバラバラと研究成果が発表されていた。研究にキャッチアップするためには、色々な分野の成果を見る必要があった（やっぱりこれは、今でも変わらないかもしれない）。色々な分野の人がネットワーク科学の研究をしているので、どこでも研究成果を発表できる一方で、なんとなくどこでもアウェイな感じがした。同じネットワーク科学分野の研究をしているはずなのに、元々の専門分野が違うために、言葉が違ってよく分からないといったこともあった。それが今では、ネットワーク科学に特化した国際会議もいくつかあって規模も大きくなり、質もそこそこ高い専門の雑誌が登場し、広い分野の国際会議（例えば、Web とかデータマイニングみたいなくくりの会議）でも、ネットワーク関連のセッションが独立で組まれるくらいになっている。筆者が修士で研究を始めて、博士、教員と進むなかで、ネットワーク科学分野の発展を実感してきた。難しさもあったが、やはり若い分野なので研究することはたくさんあって、分野全体として新しいことがどんどん出てきたし、個人としても、研究歴が浅くても色々できることがあって、楽しかった。

ということで、学際分野で 15 年くらいやってきた経験のまとめとしては、学際研究はやっぱりおもしろいということである。ただ、学際研究には難しさもあって、それを乗り越えるためには、コミュニティが重要であるように思う。通信分野は、わりと伝統的にそれぞれの学問分野がきっちり確立されているように私からすると見えるが、それをあえてごちゃまぜにして何かおもしろいことをやろうというのが、MIKA の試みであると理解している。学際分野のおもしろさを知っている経験から、MIKA をはじめとする分野横断型の研究会活動から何か新しくおもしろいものが出てくることを期待しつつ、運営にも携らせていただいている。

# 10 おわりに

## 10.1 最近の課題へのポインタ

最後に本稿でカバーできなかった最近の話題へのポインタを紹介しておきたい。本稿では紹介できなかったネットワーク分析における最近のトレンドとして、やはり機械学習の話題を避けることはできない。特にコンピュータサイエンスのドメインでは、ノードやサブネットワーク、あるいはネットワーク全体をベクトル空間上で表現するネットワーク埋め込み (network embedding, graph embedding, node embedding) [110, 111] や、ネットワークで表現されるデータを対象としたニューラルネットワークであるグラフニューラルネットワーク (GNN) [112, 113] が、色々なタスクで成果を挙げている。本稿でいくつか紹介した中心性指標というのは、影響力の強いノードの特徴を人間が考えて定量化したものであるが、大量のデータからそのような特徴を学習してしまおうというのが、埋め込みや GNN のアプローチである。このあたりは非常にスピードも早く、筆者自身も解説できるほどに理解できていないが<sup>\*1</sup>、参考となる資料へのポインタを示しておきたい。GNN やネットワーク埋め込みの教科書的な書籍としては、文献 [114] や文献 [115] が参考になる。前者は preprint が Web<sup>\*2</sup>でも入手可能である。また、Stanford の Leskovec の Machine Learning with Graphs<sup>\*3</sup> というコースの講義資料也非常に参考になる。YouTube で講義動画も見れるようである。

<sup>\*1</sup> GNN のチュートリアルとかあれば、自分が聞きたい

<sup>\*2</sup> [https://cse.msu.edu/~mayao4/dlg\\_book/](https://cse.msu.edu/~mayao4/dlg_book/)

<sup>\*3</sup> <http://web.stanford.edu/class/cs224w/>

## 10.2 まとめ

本稿では、ネットワーク分析に対する前提知識がない読者を対象として、ネットワーク分析の基本的な手法を解説した。具体的には、ネットワークの特徴量の計算方法、中心ノードの特定方法、コミュニティ抽出の方法を紹介した。ネットワーク分析を実施するための Python プログラムのソースコードも示し、学習しながら、すぐに実際のデータでネットワーク分析できることを目指した。プログラムは GitHub<sup>\*4</sup> 上にも公開しているため、ぜひ一度、ネットワーク分析を体験していただければと思う。

本稿の後半では、ネットワークの故障や攻撃に対するロバスト性の評価、ネットワーク上のウィルス拡散特性の分析、Twitter ユーザのエゴネットワークの構造解析、MIKA に関わる研究者ネットワークの分析、という研究に近い応用的かつ具体的トピックを通じて、ネットワーク分析がどのように生かされるかを紹介した。本稿で紹介したものは、かなりプリミティブな例であったが、良い問題や良い question を設定できれば、新たな研究にもつながるのではないかと考えている。もし読者らの分野の新たな問題を見つけるなんらかのきっかけになれば幸いである。

おそらく、MIKA 参加者の多数派の無線通信の研究者の方は、ネットワーク分析やネットワーク科学と聞いてあまりピンとこなかったであろうし、本稿を読んでチュートリアルを聞いた後にも、何か具体的に研究で使えそうだ、とは思わないであろう。ただ、横断型研究会の趣旨としては、それでいいと考えている。何の役に立つかは分からなければ、隣の隣くらいの分野の人々が、こういうのを面白いと思っているらしい、というのをインプットしておくことで、後々何かにつながるかもしれない。おそらく、すぐに「何かに使えそうだ!」と思えるようなことは、もう誰かがやっているので、結果的には画期的な研究にはつながらないであろう。「そういえば、こんなこともあったな」くらいで頭に入れておいて、役に立てばラッキーくらいの位置付けが、横断型研究会ではちょうどよいのかなと思っている<sup>\*5</sup>。

また、ネットワーク分析やネットワーク科学についての背景知識のある（少数派の）読者の方にとっては、本チュートリアルは簡単すぎて、退屈であったと思う。そこはご容赦いただきたいが、学生や後輩を指導する時などに何かの参考になれば幸いである。そこそこ研究的で具体的な分析例を入れたつもりであるので、研究室配属されてすぐくらいの学生の課題としては使えるのではないかと思っている。

本稿が、ネットワーク科学分野への研究者の参入のハードルを下げるのに少しでも貢献できれば幸いである。

<sup>\*4</sup> [https://github.com/s-tugawa/MIKA21\\_tutorial](https://github.com/s-tugawa/MIKA21_tutorial)

<sup>\*5</sup> とはいえ、少しでも役に立てばうれしい。

## 謝辞

今回のチュートリアル講演の機会を与えていただいた MIKA 2021 実行委員会の皆様に感謝申し上げます。幹事業務と並行してのチュートリアル講演の準備は大変でしたが、他の幹事や専門委員のみなさまの強力なサポートにより、なんとかチュートリアルの原稿を書き上げることができました。

本チュートリアル資料は、筆者がこれまでの研究の中で個別に勉強してきた内容をまとめ直したものです。これまでの共同研究者のみなさまに感謝申し上げます。特に、関西学院大学の大崎博之教授とは、学生時代から現在に至るまで 15 年ほどネットワーク科学に関する研究を実施してきました。本稿の内容の多くは、大崎先生との共同研究を通じて学んだものです。また、本チュートリアル資料は、研究室に配属される学生に最初に読んでもらうことを想定して執筆を始めました。実際には、これまで共同研究してきた学生と研究を進める中で個別に勉強したり、解説したりしてきた内容を改めてまとめ直す形となり、これまでの学生のみなさんとの共同研究が、本資料の執筆に大いに役立ちました。筑波大学・ネットワークマイニング研究室の学生ならびに卒業生の皆様にも改めて感謝申し上げます。

## 参考文献

- [1] J. Leskovec, J. Kleinberg, and C. Faloutsos, “Graphs over time: densification laws, shrinking diameters and possible explanations,” in *Proceedings of the eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining (KDD’05)*, 2005, pp. 177–187.
- [2] D. J. Watts and S. H. Strogatz, “Collective dynamics of ‘small-world’ networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [3] S. TSUGAWA and S. NIIDA, “Analyzing the effects of social network structure on the growth and survival of online communities in reddit,” *IEICE Transactions on Communications*, no. 7, pp. 760–769, 2021.
- [4] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts, “Everyone’s an influencer: Quantifying influence on Twitter,” in *Proc. 4th ACM International Conf. on Web Search and Data Mining (WSDM’11)*, 2011, pp. 65–74.
- [5] S. Tsugawa, “Empirical analysis of the relation between community structure and cascading retweet diffusion,” in *Proceedings of the International AAAI Conference on Web and Social Media (ICWSM’19)*, vol. 13, 2019, pp. 493–504.
- [6] A. D. Broido and A. Clauset, “Scale-free networks are rare,” *Nature Communications*, vol. 10, no. 1, pp. 1–10, 2019.
- [7] A.-L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [8] S. Brin and L. Page, “The anatomy of a large-scale hypertextual Web search engine,” *Computer Networks and ISDN Systems*, vol. 30, no. 1, pp. 107–117, 1998.
- [9] R. Albert, H. Jeong, and A.-L. Barabási, “Error and attack tolerance of complex networks,” *Nature*, vol. 406, no. 6794, pp. 378–382, 2000.
- [10] M. E. Newman, “Spread of epidemic disease on networks,” *Physical Review E*, vol. 66, no. 1, p. 016128, 2002.
- [11] S. Ressler, “Social network analysis as an approach to combat terrorism: Past, present, and future research,” *Homeland Security Affairs*, vol. 2, no. 2, 2006.
- [12] E. C. van Straaten and C. J. Stam, “Structure out of chaos: functional brain

- network analysis with EEG, MEG, and functional MRI,” *European Neuropsychopharmacology*, vol. 23, no. 1, pp. 7–18, 2013.
- [13] H. Yang, “Aligraph: A comprehensive graph neural network platform,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD’19)*, 2019, pp. 3165–3166.
- [14] D. Kempe, J. M. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in *Proc. 9th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining (KDD’03)*, 2003, pp. 137–146.
- [15] M. Richardson and P. Domingos, “Mining knowledge-sharing sites for viral marketing,” in *Proc. 8th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining (KDD’02)*, 2002, pp. 61–70.
- [16] A.-L. Barabási, *Network science*. Cambridge university press, 2016.
- [17] 池田裕一, 井上寛康, 谷沢俊弘, ネットワーク科学: ひと・もの・ことの関係性をデータから解き明かす新しいアプローチ. 共立出版, 2019.
- [18] 増田直紀, 今野紀雄, 複雑ネットワーク: 基礎から応用まで. 近代科学社, 2010.
- [19] T. P. Peixoto, “The graph-tool python library,” *figshare*, 2014. [Online]. Available: [http://figshare.com/articles/graph\\_tool/1164194](http://figshare.com/articles/graph_tool/1164194)
- [20] M. Bastian, S. Heymann, and M. Jacomy, “Gephi: An open source software for exploring and manipulating networks,” in *Proc. 3rd International AAAI Conf. on Weblogs and Social Media (ICWSM’09)*, 2009, pp. 361–362.
- [21] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, “Cytoscape: a software environment for integrated models of biomolecular interaction networks,” *Genome Research*, vol. 13, no. 11, pp. 2498–2504, 2003.
- [22] M. Coscia and L. Rossi, “The impact of projection and backboning on network topologies,” in *Proceedings of 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM’19)*, 2019, pp. 286–293.
- [23] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang, “Bipartite network projection and personal recommendation,” *Physical Review E*, vol. 76, no. 4, p. 046115, 2007.
- [24] M. Gao, L. Chen, X. He, and A. Zhou, “BiNE: Bipartite network embedding,” in *Proceedings of the 41st international ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR’18)*, 2018, pp. 715–724.
- [25] N. Masuda and P. Holme, “Predicting and controlling infectious disease epidemics using temporal networks,” *F1000prime Reports*, vol. 5, 2013.
- [26] T. Takaguchi, N. Masuda, and P. Holme, “Bursty communication patterns facilitate spreading in a threshold-based epidemic dynamics,” *PloS one*, vol. 8, no. 7, p. e68629, 2013.

- [27] P. Holme, “Modern temporal network theory: a colloquium,” *The European Physical Journal B*, vol. 88, no. 9, pp. 234:1–234:30, 2015.
- [28] P. Holme and J. Saramäki, “Temporal networks,” *Physics Reports*, vol. 519, no. 3, pp. 97–125, 2012.
- [29] N. Masuda and R. Lambiotte, *A guide to temporal networks*. World Scientific, 2016.
- [30] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, “Catastrophic cascade of failures in interdependent networks,” *Nature*, vol. 464, no. 7291, pp. 1025–1028, 2010.
- [31] X. Zou, “A survey on application of knowledge graph,” in *Journal of Physics: Conference Series*, vol. 1487, no. 1. IOP Publishing, 2020, p. 012016.
- [32] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, “Multilayer networks,” *Journal of Complex Networks*, vol. 2, no. 3, pp. 203–271, 2014.
- [33] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin, “The structure and dynamics of multilayer networks,” *Physics Reports*, vol. 544, no. 1, pp. 1–122, 2014.
- [34] Y. Wang, Y. Li, J. Fan, C. Ye, and M. Chai, “A survey of typical attributed graph queries,” *World Wide Web*, vol. 24, no. 1, pp. 297–346, 2021.
- [35] J. Leskovec and J. J. Mcauley, “Learning to discover social circles in ego networks,” in *Proceedings of the Neural Information Processing Systems (NIPS’12)*, Dec. 2012, pp. 539–547.
- [36] H. Yu, P. Braun, M. A. Yıldırım, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis *et al.*, “High-quality binary protein interaction map of the yeast interactome network,” *Science*, vol. 322, no. 5898, pp. 104–110, 2008.
- [37] W. Chen, Y. Wang, and S. Yang, “Efficient influence maximization in social networks,” in *Proc. 15th ACM SIGKDD International Conf. on Knowledge Discovery and Data Mining (KDD’09)*, 2009, pp. 199–208.
- [38] S. Milgram, “The small world problem,” *Psychology Today*, vol. 2, no. 1, pp. 60–67, 1967.
- [39] H. Kwak, C. Lee, H. Park, and S. Moon, “What is Twitter, a social network or a news media?” in *Proc. 19th International Conf. on World Wide Web (WWW’10)*, 2010, pp. 591–600.
- [40] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, “On the evolution of user interaction in Facebook,” in *Proc. 2nd ACM SIGCOMM Workshop on Social Networks (WOSN’09)*, 2009, pp. 37–42.

- [41] R. Albert, H. Jeong, and A.-L. Barabási, “Diameter of the world-wide web,” *Nature*, vol. 401, no. 6749, pp. 130–131, 1999.
- [42] J. Leskovec and R. Sosić, “SNAP: A general-purpose network analysis and graph-mining library,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 8, no. 1, p. 1, 2016.
- [43] L. C. Freeman, “Centrality in social networks conceptual clarification,” *Social Networks*, vol. 1, no. 3, pp. 215–239, 1979.
- [44] J. Haruta, S. Tsugawa, and K. Ogura, “Exploring the structure of social media application-based information-sharing clinical networks in a community in japan using a social network analysis approach,” *Family Medicine and Community Health*, vol. 8, no. 4, 2020.
- [45] S. Tsugawa and K. Kimura, “Identifying influencers from sampled social networks,” *Physica A: Statistical Mechanics and its Applications*, vol. 507, pp. 294–303, 2018.
- [46] C. Budak, D. Agrawal, and A. El Abbadi, “Limiting the spread of misinformation in social networks,” in *Proceedings of the 20th International Conference on World Wide Web (WWW'11)*, 2011, pp. 665–674.
- [47] U. Brandes, “On variants of shortest-path betweenness centrality and their generic computation,” *Social Networks*, vol. 30, no. 2, pp. 136–145, 2008.
- [48] M. E. J. Newman, “A measure of betweenness centrality based on random walks,” *Social Networks*, vol. 27, no. 1, pp. 39–54, 2005.
- [49] L. C. Freeman, S. P. Borgatti, and D. R. White, “Centrality in valued graphs: A measure of betweenness based on network flow,” *Social Networks*, vol. 13, no. 2, pp. 141–154, 1991.
- [50] S. Dolev, Y. Elovici, and R. Puzis, “Routing betweenness centrality,” *Journal of the ACM*, vol. 57, no. 4, pp. 25:1–25:27, 2010.
- [51] T. W. Valente, K. Coronges, C. Lakon, and E. Costenbader, “How correlated are network centrality measures?” *Connections (Toronto, Ont.)*, vol. 28, no. 1, p. 16, 2008.
- [52] S. B. Seidman, “Network structure and minimum degree,” *Social Networks*, vol. 5, no. 3, pp. 269–287, 1983.
- [53] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes, “K-core organization of complex networks,” *Physical Review Letters*, vol. 96, no. 4, p. 040601, 2006.
- [54] S. Pei, L. Muchnik, J. S. Andrade Jr, Z. Zheng, and H. A. Makse, “Searching for superspreaders of information in real-world social media,” *Scientific Reports*, vol. 4, p. 5547, 2014.
- [55] S. Carmi, S. Havlin, S. Kirkpatrick, Y. Shavitt, and E. Shir, “A model of Internet topology using k-shell decomposition,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 42, pp. 14114–14119, 2005.

- of Sciences*, vol. 104, no. 27, pp. 11 150–11 154, 2007.
- [56] S. Wasserman, K. Faust *et al.*, “Social network analysis: Methods and applications,” 1994.
- [57] L. Lü, D. Chen, X.-L. Ren, Q.-M. Zhang, Y.-C. Zhang, and T. Zhou, “Vital nodes identification in complex networks,” *Physics Reports*, vol. 650, pp. 1–63, 2016.
- [58] Y. Li, J. Fan, Y. Wang, and K.-L. Tan, “Influence maximization on social graphs: A survey,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1852–1872, 2018.
- [59] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.
- [60] S. Sahebi and W. W. Cohen, “Community-based recommendations: a solution to the cold start problem,” in *Proc. Workshop on Recommender Systems and the Social Web*, 2011, p. 60.
- [61] L. Weng, F. Menczer, and Y.-Y. Ahn, “Virality prediction and community structure in social networks,” *Scientific Reports*, vol. 3, p. 2522, 2013.
- [62] K. L. Arnemann, A. J.-W. Chen, T. Novakovic-Agopian, C. Gratton, E. M. Nomura, and M. D’Esposito, “Functional brain network modularity predicts response to cognitive training after brain injury,” *Neurology*, vol. 84, no. 15, pp. 1568–1574, 2015.
- [63] S. Fortunato, “Community detection in graphs,” *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.
- [64] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, “Metrics for community analysis: A survey,” *ACM Computing Surveys*, vol. 50, no. 4, pp. 54:1–54:32, 2017.
- [65] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
- [66] J. Xie, S. Kelley, and B. K. Szymanski, “Overlapping community detection in networks: The state-of-the-art and comparative study,” *ACM Computing Surveys*, vol. 45, no. 4, pp. 43:1–43:35, 2013.
- [67] A. Lancichinetti, S. Fortunato, and J. Kertész, “Detecting the overlapping and hierarchical community structure in complex networks,” *New Journal of Physics*, vol. 11, no. 3, p. 033015, 2009.
- [68] M. Sozio and A. Gionis, “The community-search problem and how to plan a successful cocktail party,” in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD’10)*, 2010, pp. 939–948.

- [69] A. Miyauchi and Y. Kawase, “What is a network community?: A novel quality function and detection algorithms,” in *Proc. 24th ACM International on Conf. on Information and Knowledge Management (CIKM’15)*, 2015, pp. 1471–1480.
- [70] A. Lancichinetti and S. Fortunato, “Limits of modularity maximization in community detection,” *Physical Review E*, vol. 84, no. 6, p. 066122, 2011.
- [71] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.
- [72] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.
- [73] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, “On modularity clustering,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 2, pp. 172–188, 2008.
- [74] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, p. 066111, 2004.
- [75] V. A. Traag, “Faster unfolding of communities: Speeding up the Louvain algorithm,” *Phys. Rev. E*, vol. 92, p. 032801, Sep 2015.
- [76] W. W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.
- [77] J. Leskovec, J. Kleinberg, and C. Faloutsos, “Graph evolution: Densification and shrinking diameters,” *ACM transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 2–41, 2007.
- [78] L. Hubert and P. Arabie, “Comparing partitions,” *Journal of Classification*, vol. 2, no. 1, pp. 193–218, 1985.
- [79] S. Fortunato and D. Hric, “Community detection in networks: A user guide,” *Physics Reports*, vol. 659, no. 11, pp. 1–44, 2010.
- [80] R. Guimera, M. Sales-Pardo, and L. A. N. Amaral, “Modularity from fluctuations in random graphs and complex networks,” *Physical Review E*, vol. 70, no. 2, p. 025101, 2004.
- [81] S. Fortunato and M. Barthélémy, “Resolution limit in community detection,” *Proceedings of the National Academy of Sciences*, vol. 104, no. 1, pp. 36–41, 2007.
- [82] B. H. Good, Y.-A. de Montjoye, and A. Clauset, “Performance of modularity maximization in practical contexts,” *Physical Review E*, vol. 81, no. 4, p. 046106, 2010.
- [83] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

- [84] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, “Statistical properties of community structure in large social and information networks,” in *Proc. 17th International Conf. on World Wide Web (WWW’08)*, 2008, pp. 695–704.
- [85] J. Yang and J. Leskovec, “Overlapping community detection at scale: a non-negative matrix factorization approach,” in *Proc. 6th ACM International Conf. on Web Search and Data Mining (WSDM’13)*, 2013, pp. 587–596.
- [86] M. E. Newman, “Finding community structure in networks using the eigenvectors of matrices,” *Physical review E*, vol. 74, no. 3, p. 036104, 2006.
- [87] M. E. J. Newman, “Communities, modules and large-scale structure in networks,” *Nature Physics*, vol. 8, no. 1, pp. 25–31, 2012.
- [88] B. Karrer and M. E. J. Newman, “Stochastic blockmodels and community structure in networks,” *Physical Review E*, vol. 83, no. 1, p. 016107, 2011.
- [89] P. Zhang, C. Moore, and M. E. J. Newman, “Community detection in networks with unequal groups,” *Physical Review E*, vol. 93, no. 1, p. 012303, 2016.
- [90] E. Abbe, “Community detection and stochastic block models: recent developments,” *arXiv preprint arXiv:1703.10146*, 2017.
- [91] F. D. Malliaros and M. Vazirgiannis, “Clustering and community detection in directed networks: A survey,” *Physics Reports*, vol. 533, no. 4, pp. 95–142, 2013.
- [92] A. Lancichinetti and S. Fortunato, “Community detection algorithms: a comparative analysis,” *Physical Review E*, vol. 80, no. 5, p. 056117, 2009.
- [93] J. Leskovec, K. J. Lang, and M. Mahoney, “Empirical comparison of algorithms for network community detection,” in *Proc. 19th International Conf. on World Wide Web (WWW’10)*, 2010, pp. 631–640.
- [94] J. Yang and J. Leskovec, “Defining and evaluating network communities based on ground-truth,” *Knowledge and Information Systems*, vol. 42, no. 1, pp. 181–213, 2015.
- [95] X. Huang, D. Chen, T. Ren, and D. Wang, “A survey of community detection methods in multilayer networks,” *Data Mining and Knowledge Discovery*, vol. 35, no. 1, pp. 1–45, 2021.
- [96] D. S. Callaway, M. E. J. Newman, S. H. Strogatz, and D. J. Watts, “Network robustness and fragility: Percolation on random graphs,” *Physical Review Letters*, vol. 85, no. 25, p. 5468, 2000.
- [97] J. Liu, M. Zhou, S. Wang, and P. Liu, “A comparative study of network robustness measures,” *Frontiers of Computer Science*, vol. 11, no. 4, pp. 568–584, 2017.
- [98] A. Zeng and W. Liu, “Enhancing network robustness against malicious attacks,” *Physical Review E*, vol. 85, no. 6, p. 066130, 2012.
- [99] F. Morone and H. A. Makse, “Influence maximization in complex networks

- through optimal percolation,” *Nature*, vol. 524, no. 7563, pp. 65–68, 2015.
- [100] P. Erdős and A. Rényi, “On random graphs I,” *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.
- [101] H. J. Herrmann, C. M. Schneider, A. A. Moreira, J. S. Andrade Jr, and S. Havlin, “Onion-like network topology enhances robustness against malicious attacks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2011, no. 01, p. P01027, 2011.
- [102] W. O. Kermack and A. G. McKendrick, “A contribution to the mathematical theory of epidemics,” *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 115, pp. 700–721, 1927.
- [103] K. Faust and S. Wasserman, “Blockmodels: Interpretation and evaluation,” *Social Networks*, vol. 14, no. 1-2, pp. 5–61, 1992.
- [104] Y. Chen, G. Paul, S. Havlin, F. Liljeros, and H. E. Stanley, “Finding a better immunization strategy,” *Physical Review Letters*, vol. 101, no. 5, p. 058701, 2008.
- [105] E. Ferrara, “What types of COVID-19 conspiracies are populated by twitter bots?” *First Monday*, vol. 25, no. 6, 2020.
- [106] D. Lazer, A. S. Pentland, L. Adamic, S. Aral, A. L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann *et al.*, “Life in the network: the coming age of computational social science,” *Science*, vol. 323, no. 5915, p. 721, 2009.
- [107] S. Asur and B. A. Huberman, “Predicting the future with social media,” in *Proceedings of 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 1, 2010, pp. 492–499.
- [108] M. S. Granovetter, “The strength of weak ties,” *American journal of sociology*, vol. 78, no. 6, pp. 1360–1380, 1973.
- [109] B. M. Waxman, “Routing of multipoint connections,” *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.
- [110] P. Goyal and E. Ferrara, “Graph embedding techniques, applications, and performance: A survey,” *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.
- [111] A. Grover and J. Leskovec, “node2vec: Scalable feature learning for networks,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD’16)*, 2016, pp. 855–864.
- [112] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in *Proceedings of International Conference on Learning Representations (ICLR’17)*, 2017.
- [113] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, “A comprehensive survey on graph neural networks,” *IEEE Transactions on Neural Networks and*

*Learning Systems*, vol. 32, no. 1, pp. 4–24, 2020.

[114] Y. Ma and J. Tang, *Deep Learning on Graphs*. Cambridge University Press, 2021.

[115] W. L. Hamilton, “Graph representation learning,” *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 14, no. 3, pp. 1–159.