課題研究報告書

分散型自己修復アルゴリズムを用いたネットワークの対話的な可視化に関する課題研究

仙田 一吉
学籍番号: S2030409

主指導教員 林 幸雄

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和５年３月

Abstract

Networks that are prevalent in modern society, such as electric power networks, communication networks, and social networking sites, can be abstractly represented as a set of nodes (points) and links (lines). These complex and large-scale networks share a common structure of being scale-free, characterized by a small number of hub nodes with many links and a large number of nodes with only a few links. However, these scale-free networks are also known to be highly vulnerable to attacks on their hub nodes, as the loss of just a few percent of all nodes can result in the loss of connectivity for the entire network.

Complex network science also focuses on resilience, which is the ability to maintain functionality after an attack or disruption. Resilience involves adapting to changes in the environment by reorganizing and restructuring structures and processes to enable sustainable development. In the case of scale-free networks, such as real-world networks, simply repairing a damaged network and restoring it to its original state is not enough to achieve a resilient network. The key is to have a self-healing method that can construct a more robust network after restoration, thus addressing the structural fragility inherent in scale-free networks. As previously stated, many of the networks that support modern society are highly vulnerable due to their scale-free structure. In order to achieve a resilient society that can withstand unpredictable crises such as disasters, it is essential to design systems that consider network fragmentation and recovery, such as in the case of attacks and self-healing.

However, the attack and repair processes of networks can be difficult for non-experts in network science to understand. Therefore, it is important for both scientific research and society to develop methods and tools that enable non-experts in network science to comprehend the attack and self-healing processes of networks. Research on identifying network structures that are resilient against attacks and on developing methods for building such networks has primarily been conducted through numerical simulations. As a result, it can be challenging for non-experts in network science, such as those managing power grids at electric companies, to understand the outcomes of these simulations.

Therefore, it is crucial for non-experts to be able to intuitively grasp network resilience events, such as attacks and repairs, through simple diagrams of the relationships between nodes, which can be easily navigated using basic actions like mouse operation. This research aims to enhance understanding of the attack and repair of complex networks with real-world scale-free structures through the use of interactive visualizations. This includes the process of node deletion and the addition of new links to nodes.

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　研究背景

電力、通信網、SNS といった現代社会に広く存在するネットワークはノード (点) とリンク (線) の集合として抽象表現することが可能である。これら一般的に複雑で大規模なネットワークはスケールフリー性という共通した構造を持ち、ハブノードと呼ばれる莫大なリンクを有する少数のノードと、ほんの少量のリンクを有する大多数のノードで構成されていることが知られている [1]。他方、スケールフリーネットワークは莫大なリンクを有するハブノードへの攻撃に対して非常に脆弱であることも明らかとなっており [2]、全ノードの数%にあたる程度のハブノードを失うだけでネットワーク全体の連結性を喪失してしまうことが知られている。インターネットを例にとると、DNS ルートサーバーがハブノードに相当する。

一方、攻撃後でもいかに機能を保つかと言ったレジリエンス (復活力) に関する研究も行われている。レジリエンスとは、攻撃のような環境の変化に対して構造・プロセスの組み替えや再構築を持って対応することで、持続的な発展を可能にする適応能力である [3]。現実のネットワークようなスケールフリー性を有するネットワークについて考えてみると、単に損傷したネットワークを修復して元のネットワークを復元できるだけでは、構造的に脆弱なスケールフリー構造が残り続けるためレジリエントなネットワークとは言えない。レジリエントなネットワークを実現するためには、復元後により頑健性が高いネットワークが構築できる自己修復法を有していることが鍵となる。

以上のように、現代社会を支えるネットワークの多くがスケールフリー構造ゆえ非常に脆弱であることから、災害等予測不能な危機に強いレジリエントな社会を実現するためには、攻撃や自己修復といったネットワークの分断や復活を考慮した上でシステム設計を行うことが望ましい。ところが、ネットワーク科学の非専門家にとってはネットワークの攻撃・修復過程は必ずしも理解しやすいとは言えず、ネットワーク科学の非専門家がネットワークの攻撃・自己修復過程を理解できるような手法・ツールの開発は、科学的研究のみならず社会的にも重要である。

1.2　研究目的

スケールフリー性を有するネットワークの連結性はハブに強く依存しており、多くのリンクを持つノードに対する攻撃 (ノード削除) に晒されると立ち所に全体の連結性を喪失する。このような攻撃に対する頑健性を有するネットワーク構造の解明や、その具体的な構築方法の研究が、数値シミュレーションを中心に行われている。一方、電力会社の送電網担当といったネットワーク科学の非専門家が数値シミュレーション結果を理解することは一般に困難であるため、研究成果を役立てたレジリエントなネットワーク構築を社会で実現するためには、非専門家が攻撃・修復というネットワークレジリエンスの事象をマウス操作の様な簡単な動作のみで、各ノードの関係図という形で直感的に理解できることが重要である。そこで本課題研究では、現実のスケールフリー構造を有する複雑ネットワークに対する攻撃及び修復、つまりノード削除とノードへの新リンク追加過程を、対話的なアプリケーションにより可視化して理解を促す。可視化によりネットワークにおける破壊状況を明らかにして頑健なネットワークに再構築するために修復が有効な箇所を示すとこができれば、災害や攻撃が発生する前に破壊状況の把握や修復計画の策定を事前に行うことでレジリエンな社会を実現することが可能になる。

1.3　本論の構成

本論文の構成を以下に示す。

第２章　本研究と関係がある従来研究について述べる。ネットワーク分析の基本的な用語、ネットワーク修復方法、ネットワークの可視化について述べる。

第３章　作成したネットワークの可視化ツールについて概要を述べる。

第４章　実データおよび、実世界を模したデータに対して、ネットワーク攻撃・修復の過程を可視化ツールにより視覚化した結果を示す。ネットワーク攻撃・修復による、次数分布といった指標の変化、ネットワーク図のそのもの変化を調査し、可視化ツールのネットワークシミュレーターとしての有用性を示す。

第５章　本研究の結果をまとめる。

第2章　関連研究

本課題研究では、ネットワークをノード集合 V=$\{v_1, v_2, ..., v_n\}$ と、その二点をつなぐリンク集合 E=$\{e_1, e_2, ..., e_m\}$ からなるグラフ G=(V,E) で表現する。リンクの両端が同じノードである自己ループや同一ノード間に複数リンクが存在する多重辺を許さない単純グラフを前提とする。また、リンクに重みや方向もないものとする。以下、ネットワーク分析の基本的な用語、ネットワーク可視化ツールに関連したネットワーク修復法および既存研究について説明を行う。

2.1　ネットワーク分析について

2.1.1　最大連結成分

ハブノードや多数のノードが除去行われると、スケールフリーネットワークは複数の孤立した連結成分に分断されやすい。連結成分とは、複数のノードがリンクで繋がった部分グラフであり、孤立した連結成分で最大のものを最大連結成分と呼ぶ。最大連結成分を用いてネットワークの連結性を測る。

$$C = \frac{S(q)}{N} \tag{2.1}$$

ここで、C は連結性、S(q) は最大連結成分のノード数、N はネットワーク全体のノード数である。

2.1.2　次数分布

次数 (degree) とは、ノードにつながるリンクの数を表しており、次数分布とはあるネットワークのノードの次数がどのように分布しているかを表している。

2.1.3　次数順攻撃

ネットワークに対する代表的な攻撃手法の一つで、次数の大きい順に、つまりハブノードを選択的に削除していく。スケールフリーネットワークはこのような攻撃手法に非常に脆弱で、わずか数%の除去率でネットワークがバラバラになって

しまうということが知られている [2, 4]。他に代表的なネットワーク攻撃手法としては、ランダム故障を模したランダム攻撃、ループを破壊して現在最も攻撃力が高いとされている BP(Belief Propagation) 攻撃 [5] が挙げられる。

2.1.4 人工ネットワーク

ネットワーク研究では、アルゴリズムの評価などに実世界のデータの他に、ネットワークの大きさや特徴を調整することのできる人工的ネットワークも使用する。人工的ネットワークの代表例として、実世界を模した BA（Barabási Albert) モデルがある。BA モデルがスケールフリー性をもつ理由は、ネットワーク成長と優先的選択のにある [6]。ネットワーク成長とは時間が経つにつれてノード数が大きくなり、ネットワークの規模が大きくなること、優先的選択とは成長によって新ノードからが次数が大きい既存ノードに優先的に接続することである。これらにより、ノード数増加に伴い高次数ノードはよりリンクを獲得し、ハブと呼ばれるリンクが集中したノードが発生する。

他にも、攻撃への耐性が強い玉葱状ネットワーク [7] を生成する IPA(Inverse Preferential Attachments) モデルもある [8]。ネットワーク成長は BA モデルと同様であるが、リンク追加の考え方が異なる。新ノードを加追する時、接続するノードを次数 k に対して k-β に比例した確率で選択する。これを一度に追加するリンクの回数に応じて実行する。

2.1.5 NetworkX

NetworkX[9] は、グラフ・ネットワーク分析のための Python ライブラリでり、ネットワーク分析のための数多くの関数やアルゴリズムが実装されている。また、Pandas のようなデータ操作ライブラリや、NumPy や SciPy といった数値解析ライブラリとの連携が容易である点が特徴である。欠点は数千ノードを超えるような大規模ネットワークを高速に処理することは困難であることである。

2.1.6 Dash

Dash[10] は Python ベースで記述可能な Web アプリケーションフレームワークであり、データ分析等に用いられるダッシュボード構築に用いられる。その特徴として、フロントエンドを Python コードで簡潔に実装できること、Ploty.py や Ploty Express を用いてデータをインタラクティブに可視化できることが挙げられる。また、ネットワーク図を描画する Dash Cytoscape と呼ばれるコンポーネントを有しており、リッチでインタラクティブなネットワーク図をブラウザ上で簡単に表示・操作することができる。

2.2 ネットワーク修復の研究

攻撃などで機能を喪失したネットワークを回復するため、これまで複数の方法が提案されているが大別すると、元のネットワーク構造を回復することを目的とする方法 [15] とより頑健に再構築する方法 [16] に分けられる。本節ではより頑健に再構築する方法について着目する。

近年の研究から頑健性とループの関係性が明らかとなってきた。例えば、最大連結成分の除去とループ除去に必要な最小ノード数が漸近的に等価であることが以下のように示されている [13]。ネットワーク G が持つ最大連結成分の大きさを、定数 C より小さくするために除去するノード数の最小比率を $\theta_{dis}(G, C)$ と、ネットワーク G を無ループ構造にするために除去する必要があるノード数の最小比率を $\theta_{dec}(G)$ とすると、

$$\theta dec(p_k) = \lim_{N\to\infty} E[\theta_{dec}(G)] \tag{2.2}$$

$$\theta_{dis}(p_k) = \lim_{N\to\infty} \lim_{C\to\infty} E[\theta_{dis}(G, C)] \tag{2.3}$$

とか書き表され、任意の次数分布 p_k では、$\theta_{dis}(p_k) \leq \theta_{dec}(p_k)$ となる。次数分布 q_k であるランダムグラフ集団における平均値を E[・] とする。また、次数の 2 乗の期待値が有限:$< k^2 >= \sum_k k^2 p_k < \infty$ ならば、$\theta_{dis}(p_k) = \theta_{dec}(p_k)$ であると示されてい [13]。

一方、インフルエンサーと呼ばれる情報伝搬に影響力のあるノード集合を除去すると連結性が失われることを示した研究もある [14]。ノード数 N=$n_1, n_2, ..., n_N$, リンク数 M のネットワークにおいて、あるリンクから別のリンクへ情報伝搬をメッセージ伝搬式で表現すると以下のようになる。

$$v_{i\to j} = n_j\{1 - \prod_{k\in\partial_{i/j}} (1 - v_{i\to j})\} \tag{2.4}$$

$\partial_{i/j}$ はノード j を含まないノード i の近隣ノード集合である。

$v_{i\to j} = 0$ とは連結性が失われ情報伝搬が途絶えた状態であるから、インフルエンサーを見つけ除去することは連結成分を除去することと言える。ネットワークからのノード除去率 q($0 \leq$ q ≤ 1) とし、どのようなノードを除去すると情報伝搬がなくなるかを考えると、式 (2.4) 右辺の解の安定性は線型近似したヤコビ行列の最大固有値 $\lambda(n; q) < 1$ となる場合である。つまり、最大固有値 $\lambda(n; q)$ が最小となるようなノードを除去することである。これらから、ネットワークを効率的に破壊するのようなノード除去とはループの削除と考えることができる。

この研究結果に着目して頑健性の高いネットワークを作るためにループ強化を行なったネットワークは現在最も高い頑健性を示す [17, 18]。

ところで、ループ強化に基づいたより頑健なネットワーク再構築するための修復法の研究が行われている。現在最も高い頑健性を示すネットワークに再構築する修復法もループ強化に基づいている [19, 20]。以下に、修復法について簡単に記

載する。前提として、修復に利用できるリンク数 M_h は次のように定義される。

$$M_h = r_h \sum_{i \in D_q}^{\sim} k_i \tag{2.5}$$

ここで、k_i はノード i の次数、D_q は攻撃を受けたノードの集合、$\sum^{\sim}$ は重複なしの被攻撃ノード次数の合計、r_h は制御パラメータであり範囲は $0 \leq r_h < 1$ である。

Step1: ノードが攻撃を検知すると、ネットワークは修復モードを開始する。各ノードは予め、ローカルマップと呼ばれる自分から 3 ホップ以内のノードリストを持っており、このローカルマップを用いて情報交換を行うことで、修復範囲を決定する。

Step2: 上の 3 ホップから拡張した修復範囲内において、攻撃でリンクを喪失したノード同士を接続し、輪を形成する。輪は最も単純なループである。もし修復リンク数 M_h が不足して完全な輪を形成できない場合、M_h 本の範囲で輪を途中まで作成する。

Step3: 輪を形成してもなお修復リンク数が残っている場合は、ループ強化のため各輪上でノードを接続する。

本課題研究では、ネットワークの修復法 [19, 20] としてこの方法を実装している。

図 2.1 に上で説明したネットワーク修復法の図を示す。赤色ノードは攻撃されたノード、赤色点線は喪失したリンク、青色ノードはリンクの一部が失われダメージを受けたノード、緑色と黄色の線はそれぞれリング形成とループ強化のための修復リンクを表している。攻撃後、青色ノードはローカルマップを使用して修復範囲を決定する。その際、修復範囲は赤色ノード、赤色点線リンクを含む 3 ホップ以内のノードである。修復リンクは、まず拡張した修復範囲内で輪を形成するように青色ノード同士を繋ぎ合わせる (緑色リンク)。なお修復リンク数が残っている場合は、ループを強化するため、青色ノードと緑色リンクで作った輪の中にさらにループを作るよう青色ノード同士を繋ぐようにリンクを接続する (黄色リンク)。

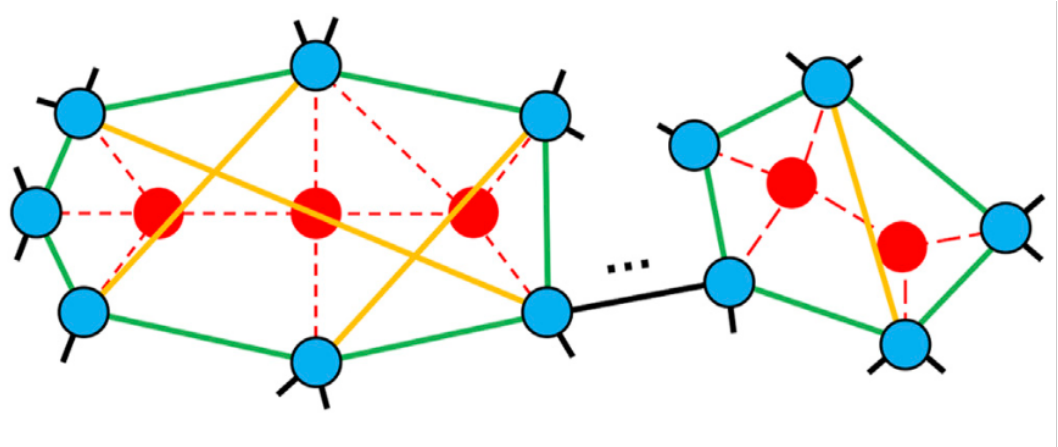

図 2.1: ネットワーク修復方法の説明 [20] から抜粋
赤色ノードは攻撃されたノード、赤色点線は喪失したリンク、青色ノードはリンクの一部が失われダメージを受けたノード、緑色と黄色の線はそれぞれリング形成とループ強化のための修復リンクを表している。

2.3 従来のネットワーク可視化研究

ネットワークデータを人間が理解しやすい形に変換し、描画することをネットワークの可視化という。ノード数が数千から数万程度であり、リンク密度が高くないネットワークにおいては、可視化によって構造を明らかにするだけで簡単な分析となることもある。ネットワークの可視化方法には様々なものが存在するが、本節ではノード間をリンクで結んだいわゆるグラフ構造を二次元平面上に描画する可視化について本課題研究と関連性がある可視化方法を紹介する。

グラフ構造を二次元平面上に描画する可視化研究は、ノードやリンクをどのように配置するかという描画手法や描画プラットフォームといった点から行われている。描画手法の研究は、可能な限り多くの描画制約を満たした上で見やすい可視化を得る様々な手法が提案されている Mi:2016。

描画手法の代表的なものにバネモデル (Spring Layout) と呼ばれるものがあり [22]、ネットワークの可視化の際にしばし用いられる。これはノード間を結ぶリンクは自然長となるようにノードに引力もしくは斥力を与え、またノード同士が重ならないよう自分とつながっていない全てのノードから距離に応じた斥力を受けるとしてして位置の移動を繰り返していき、最終的に安定状態になったもの描画する。しかしながら、このような手法はノードのすべてのペア間の斥力を正確に計算する必要があり描画速度に課題があるため、近年では斥力の計算を高速化する取り組みがなされている [21, 23]。

描画プラットフォームは、入手が容易であることから NetworkX[9] と Matplot[11] の組み合わせや Cytoscape[12] といったオープンソースソフトウェアが用いられる

ことが多い。これらは入力データを二次元平面上に描画する静的なネットワーク可視化を行うものである。ネットワークに限らず、一般に情報可視化の手順は、全体を俯瞰し、見えてきた部分構造を徐々に詳しく調査し，段階的に調査を詳細化するもの [24] であり、この観点に照らし合わせると拡大・縮小や情報の取捨選択を行うには都度描画を行う必要がある静的なネットワーク可視化は煩雑な作業が必要となる。このような用途には対話的なネットワーク可視化が有効であり、Web ブラウザベースのプラットフォームが提案されている [25]。Web ブラウザベースのメリットはマウス操作により動的なネットワーク可視化・分析が可能になること、また利用者の環境と実行環境を切り離すことができるところにある。また、単にネットワークを可視化するのみならず、次数順攻撃を受けた際のネットワーク変化を対話的に可視化した簡易なシミュレーションツールも未完成ながら検討されている [26]。本課題研究では、ネットワークの修復に着目し、攻撃を受けたネットワークがどのように頑健なネットワークに再構築されるのかを対話的に可視化するシミュレーションツールを実装する。

第3章　ネットワーク可視化ツール

本章では、本課題研究のネットワーク可視化ツールの構成・各機能を紹介する。また使用画面の例を示す。

3.1　ネットワーク可視化ツールの構成

本課題研究で作成するネットワーク可視化ツールは Web ブラウザ上で動作することとし、フロントエンドのフレームワークとして Dash を採用した。サーバーサイドの処理は主に Python で記述し、部分的に C++を用いる。Dash[10] は画面上から処理を受け付けると、callback の呼出という形で Python で記述された関数に処理を投げ、戻り値を受け取り表示を行う。本ツールでは、計算コストがかかり有効なライブラリが提供されていないネットワークの攻撃部分を C++で記述し、ctypes を用いて Python 上から関数呼出を行う。

表 3.1: 本ツールの開発言語

フロントエンド	HTML, CSS
サーバーサイド	Python, C++

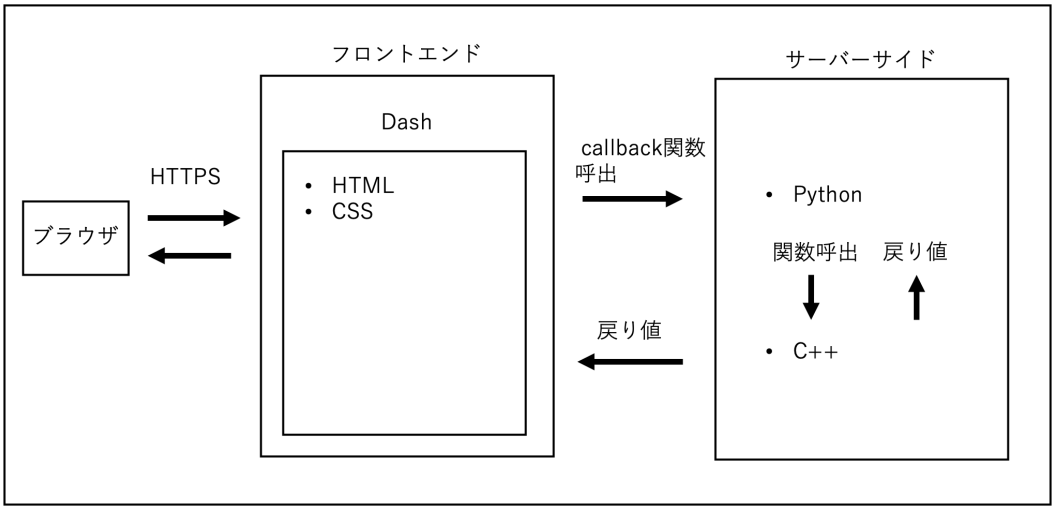

図 3.1: ネットワーク可視化ツールの構成

3.2　ネットワーク可視化ツールの機能

本節では、作成したネットワーク可視化ツールの機能と、想定する使用方法を実際の画面示しつつ説明する。

3.2.1　ネットワーク可視化ツールの画面

図 3.2 に本ツールの画面を示し、以降に画面の各機能について述べる。

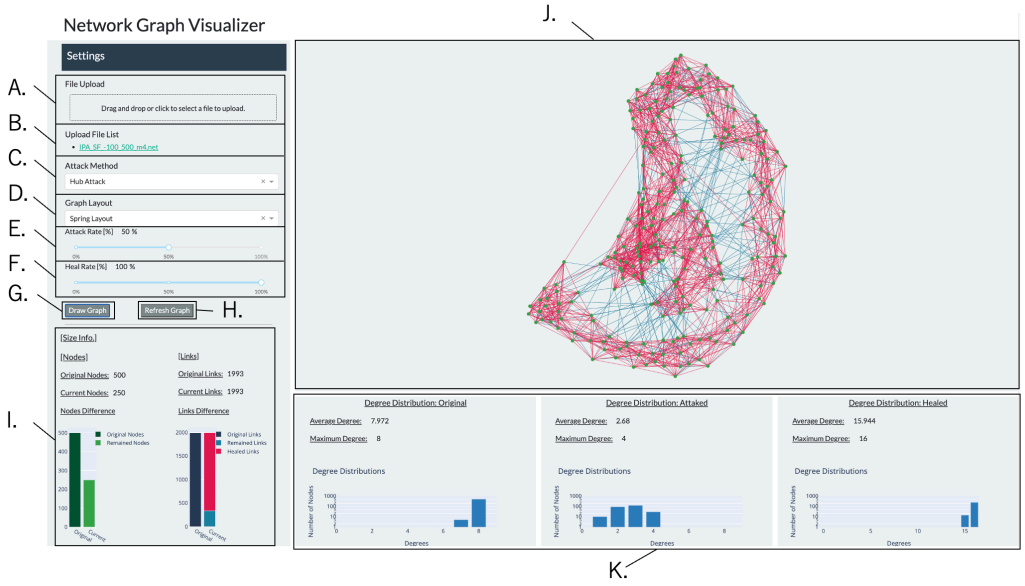

図 3.2: ネットワーク可視化ツールの画面

3.2.2　ネットワーク可視化ツールの画面機能

A. File Upload

ネットワークファイルをドラッグ&ドロップ、もしくは点線内をクリックしてディレクトリからネットワークファイルを指定することで、可視化対象のネットワークファイルをツールへアップロードすることができる。

B. Upload File List

アップロードしたファイルが表示される。

C. Attack Method

攻撃手法をリストから選択する (図 3.3)。本ツールでは、Hub Attack(次数順攻撃)、Random Attack(ランダム攻撃)、BP Attack(BP 攻撃) の 3 つから選択する。

D. Graph Layout

ネットワークの描画アルゴリズムをリストから選択する (図 3.4)。本ツールでは、Spring Layout(バネモデル)、Random Layout(ランダムモデル)、Kamada Kawai Layout(KK モデル)、Original Layout(ネットワークが持っている座標表示) の 4 つから選択する。

E. Attack Rate

ネットワークへの攻撃率を指定するスライダー。0 %から 100%の範囲を 1%単位で指定できる。

F. Heal Rate

ネットワークの修復率を指定するスライダー。0%から 100%の範囲を 1%単位で指定できる。

G. Draw Graph

A. から F. のパラメータを選択し、本ボタンを押すとネットワークグラフの計算・描画を開始する。

H. Refresh Graph

入力ファイルを変更する際には、本ボタンを押すことで、J. に表示された選択したパラメータ、ネットワークグラフをクリアする。

I. Size Info.

ネットワークの大きさに関する情報を表示する。

- Original Nodes・Links : 攻撃前のサイズ情報
- Current Nodes・Links : 攻撃/修復後のサイズ情報

J. ネットワークグラフの描画

A から F のパラメーターに基づいて計算されたネットワークグラフが描画される。緑色の点がノード、青い線がネットワーク攻撃前に存在したリンク、赤い線がネットワーク修復により新たに追加されたリンクを表す。

K. Degree Distribution

ネットワークの平均次数、最大次数、攻撃前、攻撃後、修復後の次数分布をヒストグラムで表示する。

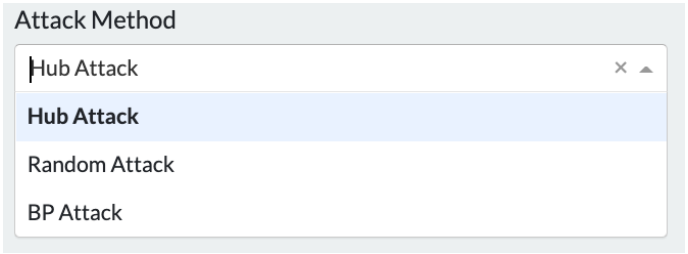

図 3.3: ネットワーク可視化ツールの攻撃手法画面

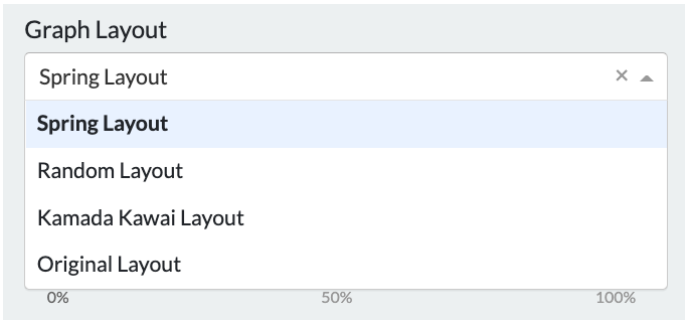

図 3.4: ネットワーク可視化ツールのレイアウト画面

3.2.3 ネットワーク可視化ツールの使用方法

ネットワーク可視化ツールの想定する使用方法について記載する。

Step1: 手元のネットワークファイルをネットワーク可視化ツールへアップロードし、図 3.2 の C. から F. のパラメータを設定、グラフの描画を行う。

Step2: 図 3.2 の I. から K. に表示されるネットワークの攻撃・修復結果を確認する。必要があれば、C. から F. のパラメータを変更し再度描画を行う。

Step3: Step2 を必要なだけ繰り返す。

Step4: ネットワークの可視化が終わったらツールを開いるブラウザを閉じる。もし、別のネットワーク可視化をこなう場合は図 3.2 の H.(Refresh Graph) をクリックし、一度画面をクリアした上で、Step1 に戻る。

第4章　ネットワーク可視化ツールの実験・評価

インフラネットワークの実データと、人工的ネットワークを作成したネットワークとを可視化ツールで可視化し、ネットワークが攻撃と修復にどのように変化するのかを評価する。評価は、主に頑健に修復を行なったネットワークを可視化するとどのように描画されるのか、そこからどのような示唆があるのかという観点から行う。

対象とするネットワークは、公開されている現実のインフラネットワーク、BA モデル、IPA モデルである。

攻撃手法は次数順攻撃、ネットワーク描画レイアウトはネットワークデータに座標が含まれていれば Original Layout、なければ Spring Layout(バネモデル) で描画する。

評価用端末は MacBook Air2022(CPU:Apple M2, メモリ:8GB) を用いた。

4.1　ネットワークの詳細

Step1: Cost266

アーキテクチャーやアルゴリズムを比較・評価を目的としたヨーロッパの光伝送ネットワークデータ [27]。

Step2: Janos-US-CA

アメリカおよびカナダの主要都市の通信ネットワークデータ [28]

Step3: BA モデル

ノード数 N=500、ノード追加時に生成するリンク数 m=2 で生成した BA モデルネットワーク [6] データ。

Step4: IPA モデル

ノード数 N=500、ノード追加時に生成するリンク数 m=2、ノード選択に関する変数 beta=-100 で生成した IPA モデルネットワーク [8] データ。

表 4.1 にはネットワークデータの基本特性を示す。

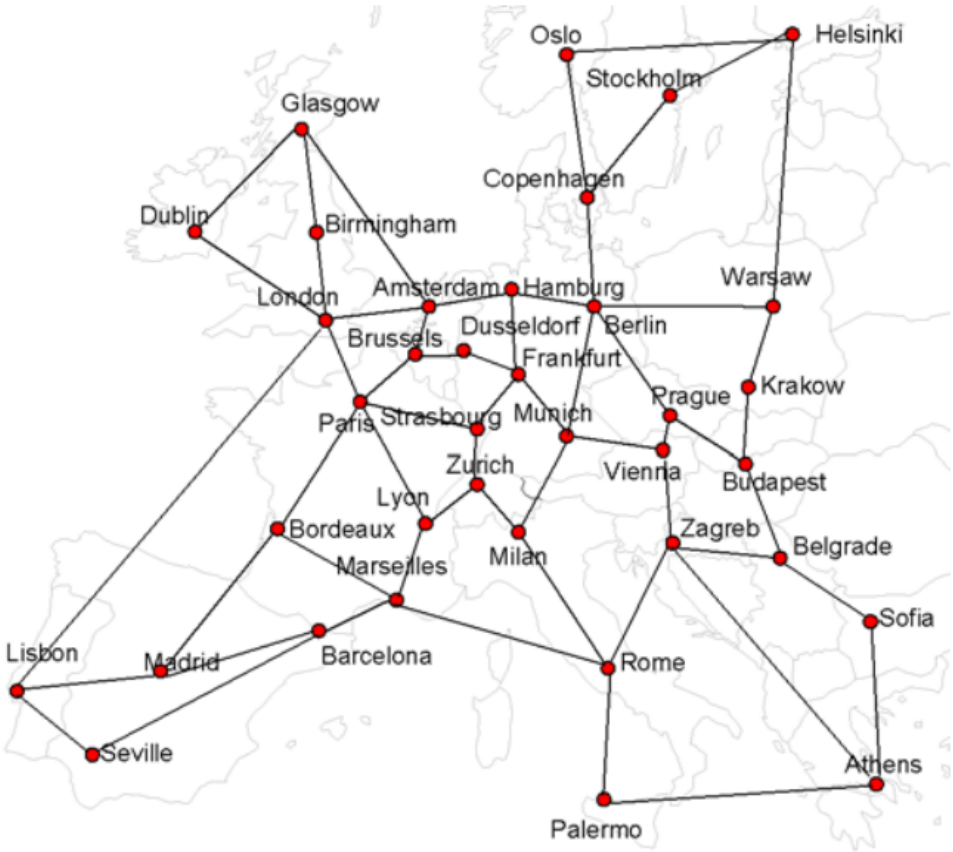

図 4.1: Cost266 のネットワークデータと地図を重ねた図 [28] から抜粋

表 4.1: ネットワークの基本特性

ネットワーク名	総ノード数	総リンク数	最大次数	最小次数	平均次数
Cost266	37	57	5	2	3.081
Janos-US-CA	37	61	5	2	3.128
BA モデル	500	1007	55	2	4.028
IPA モデル	500	1007	7	2	4.028

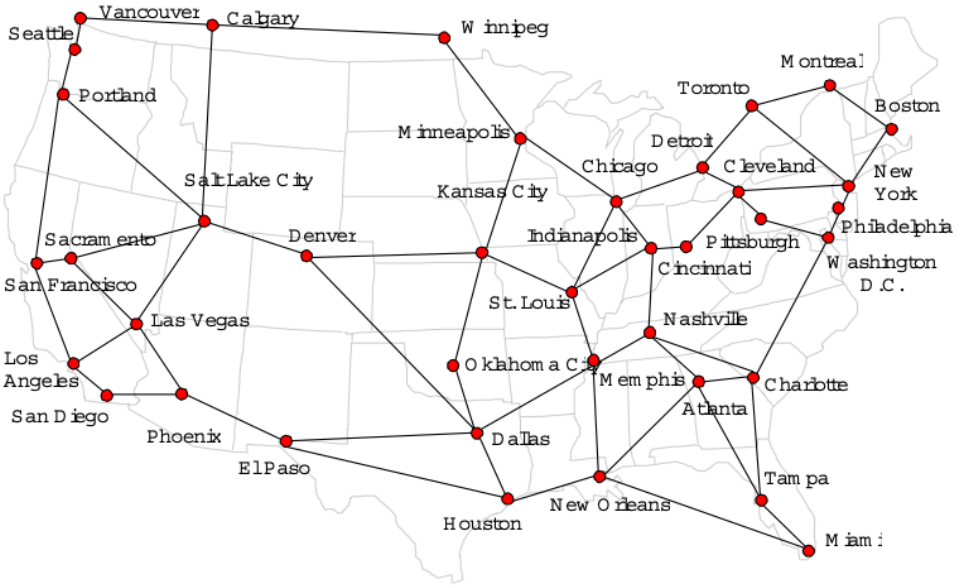

図 4.2: Janos-US-CA のネットワークデータと地図を重ねた図 [28] から抜粋

4.2 Cost266 の実験・評価

図 4.3 に、攻撃・修復を行なっていない Cost266 の可視化例を示す。

図 4.4 には、攻撃率 50%の状態を示す。Cost266 は攻撃率 50%で大きく崩れ、表示されているネットワークにはループが存在していないことが、図 4.4 から読み取ることができる。BA モデルが攻撃率 10%程度でも大きく崩れてしまうことを考えると、Cost266 は比較的頑健なネットワークといえ、これは都市間を結ぶ伝送ネットワークの特性が現れている。この状態からネットワークの修復を行う。

図 4.5 には攻撃率 50%、修復率 20%を示す。図 4.5 から修復リンクはバラバラに孤立した連結成分同士を輪を作るように接続されていることがわかる。

図 4.6 には攻撃率 50%、修復率 100%を示す。この状態ではグラフはすべてのノード同士が連結した完全グラフの状態となり、攻撃に対して強い頑健性を発揮するネットワークに再構築することができたと言える。

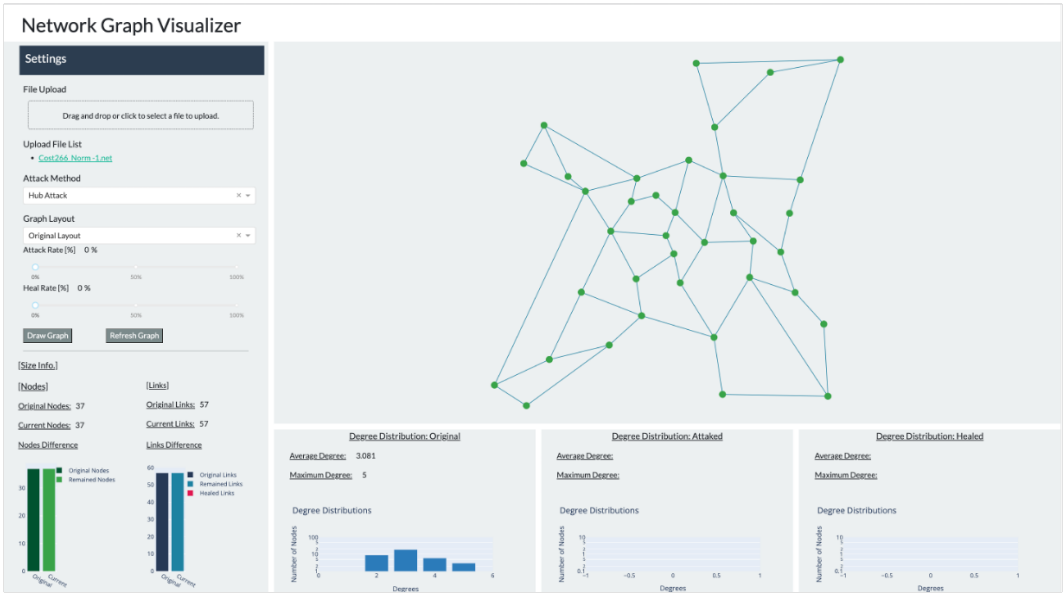

図 4.3: Cost266 のネットワーク図 (攻撃率 0%、修復率 0%)

図 4.4: Cost266 のネットワーク図 (攻撃率 50%、修復率 0%)

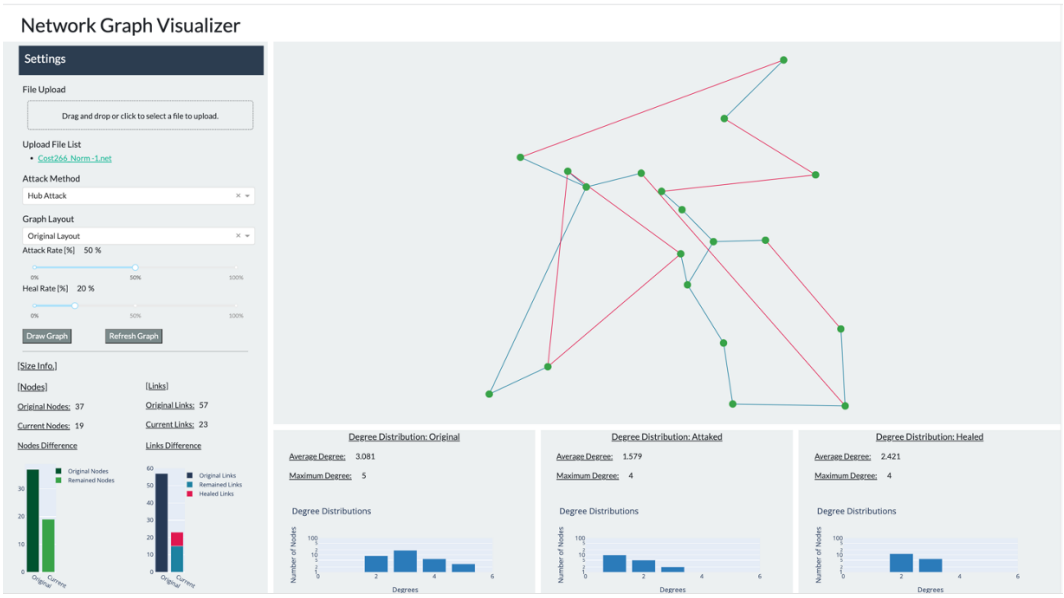

図 4.5: Cost266 のネットワーク図 (攻撃率 50%、修復率 20%)

図 4.6: Cost266 のネットワーク図 (攻撃率 50%、修復率 100%)

図 4.7: バネモデルで表示した Cost266 のネットワーク図 (攻撃率 50%、修復率 1020%)

4.3 Janos-US-CA の実験・評価

図 4.8 に、攻撃・修復を行なっていない Janos-US-CA の可視化例を示す。

図 4.9 には、攻撃率 20%の状態を示す。図 4.9 からアメリカ東海岸、西海岸、中西部を中心に攻撃を受けていることがわかる。次数順攻撃はリンクを多く持つノードを優先的に攻撃するため、大都市がある場所が最初に攻撃されることは直感と一致する。

Janos-US-CA は攻撃率 50%で大きく崩れ、表示されているネットワークにはループが存在していないことが図 4.10 から読み取ることができる。BA モデルが攻撃率 10%程度で崩壊してしまうことを考えると Janos-US-CA は比較的頑健なネットワークといえ、これは都市間を結ぶ伝送ネットワークの特性が現れている。この状態からネットワークの修復を行う。

図 4.11 には攻撃 50%、修復率 20%を示す。図 4.11 から修復リンクはバラバラに孤立した連結成分同士を輪を作るように接続されていることがわかる。

図 4.12 には攻撃率 50%、修復率 100%を示す。この状態では再生可能なリンクは全て使用している。次数分布に着目すると、全てのノードがほぼ同じ次数をとっていることがわかる。このように脆弱なハブ構造のない、攻撃に対して強い頑健性を発揮するネットワークに再構築できたことが視覚的にも理解できる。

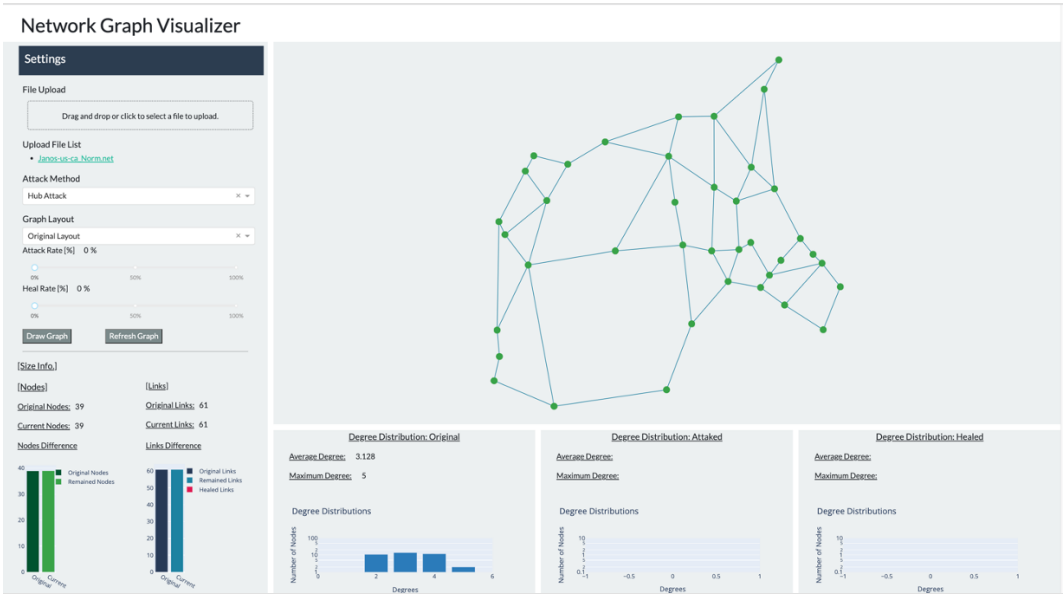

図 4.8: Janos-US-CA のネットワーク図 (攻撃率 0%、修復率 0%)

図 4.9: Janos-US-CA のネットワーク図 (攻撃率 20%、修復率 0%)

図 4.10: Janos-US-CA のネットワーク図 (攻撃率 50%、修復率 0%)

図 4.11: Janos-US-CA のネットワーク図 (攻撃率 50%、修復率 20%)

図 4.12: Janos-US-CA のネットワーク図 (攻撃率 50%、修復率 20%)

4.4 BA ネットワークの実験・評価

図 4.13 に、攻撃・修復を行なっていない BA モデルの可視化例を示す。

攻撃率 10%(図 4.14)、つまり次数の多い上位 10%のノードを取り除いただけで平均次数 1.5 程度とネットワークがほぼ崩壊し、攻撃率 20%(図 4.15) 程度で完全にループがなくなっていることが見て取れる。

攻撃率を 50%(図 4.16) まで上昇させるとノードがリンクを持たない孤立した状態となる。この状態からネットワークの修復を行う。

図 4.17 には攻撃率 50%、修復率 10%を示す。図 4.17 から修復リンクはバラバラに孤立した連結成分同士を輪を作るように接続されていることがわかる。

図 4.18 には攻撃率 50%、修復率 100%を示す。この状態では再生可能なリンクは全て使用している。次数分布に着目すると、全てのノードがほぼ同じ次数をとっていることがわかる。また、図から修復リンクはノード同士が多数の小さなループを作るように接続されている様がわかる。このように脆弱なハブ構造のない、攻撃に対して強い頑健性を発揮するネットワークに再構築できたことが視覚的にも理解できる。

一方、バネモデルの計算時間はノード数に大きく依存していることがわかった。表 4.2 には、リンク数は等しいがノード数が異なる二つのネットワーク、元の BA モデル (攻撃率 0%、修復率 0%) と攻撃・修復後の BA モデル (攻撃率 10%、修復率 100%)、をバネモデルで描画する時にかかる計算時間を示す。図 4.13 と図 4.19 から、ノード数は 10%しか変化していないのにも関わらず、計算時間は大幅に増

加していることから、ノードの位置を決定する斥力の計算量とその時間はノード数に大きく依存すると言える。

表 4.2: 元の BA モデル (攻撃率 0%、修復率 0%) と攻撃・修復後の BA モデル (攻撃率 10%、修復率 100%) をバネモデル座標計算時間 (100 回平均)

ネットワーク名	計算時間 [s]
BA モデル (攻撃率 0%、修復率 0%)	1.326
BA モデル (攻撃率 10%、修復率 100%)	0.317

図 4.13: BA モデルのネットワーク図 (攻撃率 0%、修復率 0%)

図 4.14: BA モデルのネットワーク図 (攻撃率 10%、修復率 0%)

図 4.15: BA モデルのネットワーク図 (攻撃率 20%、修復率 0%)

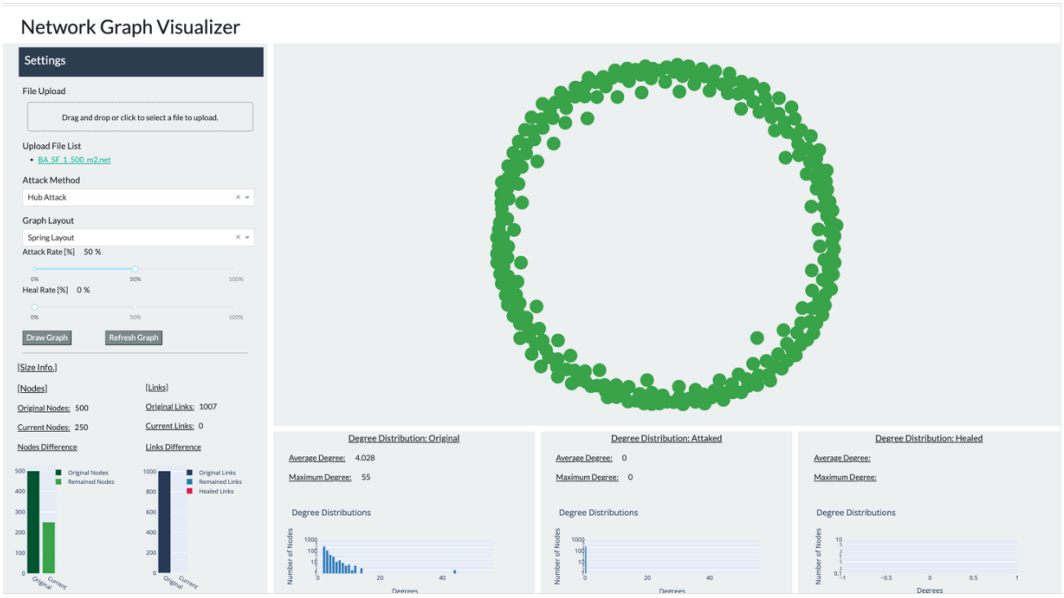

図 4.16: BA モデルのネットワーク図 (攻撃率 50%、修復率 0%)

図 4.17: BA モデルのネットワーク図 (攻撃率 50%、修復率 10%)

図 4.18: BA モデルのネットワーク図 (攻撃率 50%、修復率 100%)

図 4.19: BA モデルのネットワーク図 (攻撃率 10%、修復率 100%)

4.5 IPA ネットワークの実験・評価

図 4.20 に攻撃・修復を行なっていない IPA モデルの可視化を示す。

IPA モデルは攻撃率 40%(図 4.21) でも部分的にループを形成し続けていることが見て取れる。

攻撃率 60%(図 4.22) でループは消失し、攻撃率 80%(図 4.23) で各ノードは完全に孤立する。この状態からネットワークの修復を行う。

図 4.24 には攻撃率 80%、修復率 10%を示す。図 4.24 から修復リンクはバラバラに孤立した連結成分同士を輪を作るように接続されていることがわかる。

図 4.25 には攻撃率 80%、修復率 100%を示す。この状態では再生可能なリンクは全て使用している。次数分布に着目すると、全てのノードがほぼ同じ次数をとっていることがわかる。また、図から修復リンクはノード同士が多数の小さなループを作るように接続されている様がわかる。このように脆弱なハブ構造のない、攻撃に対して強い頑健性を発揮するネットワークに再構築できたことが視覚的にも理解することができる。

表 4.3 に、同一ノード数・リンク数である BA ネットワークと IPA ネットワークをバネモデルで描画するためににかかる時間を示す。2 つのネットワークは同一ノード数・リンク数、レイアウトであることから、2 つのネットワークの差分はネットワーク構造のみであるが、表示にかかる時間は有意差 p0.05 で IPA ネットワークの方が遅くなっている。IPA モデルは同じような次数のノードで構成されているため、ノードが座標上に均等に分布している。そのため、新しいノードが追加された際に安定状態になるまでのノード移動回数、つまり計算量が多いことによると考えられる。

表 4.3: BA モデル (攻撃率 0%、修復率 0%) と IPA モデル (攻撃率 0%、修復率 0%) をバネモデル座標計算時間 (100 回平均)

ネットワーク名	計算時間 [s]
BA モデル (攻撃率 0%、修復率 0%)	1.326
IPA モデル (攻撃率 0%、修復率 0%)	1.402

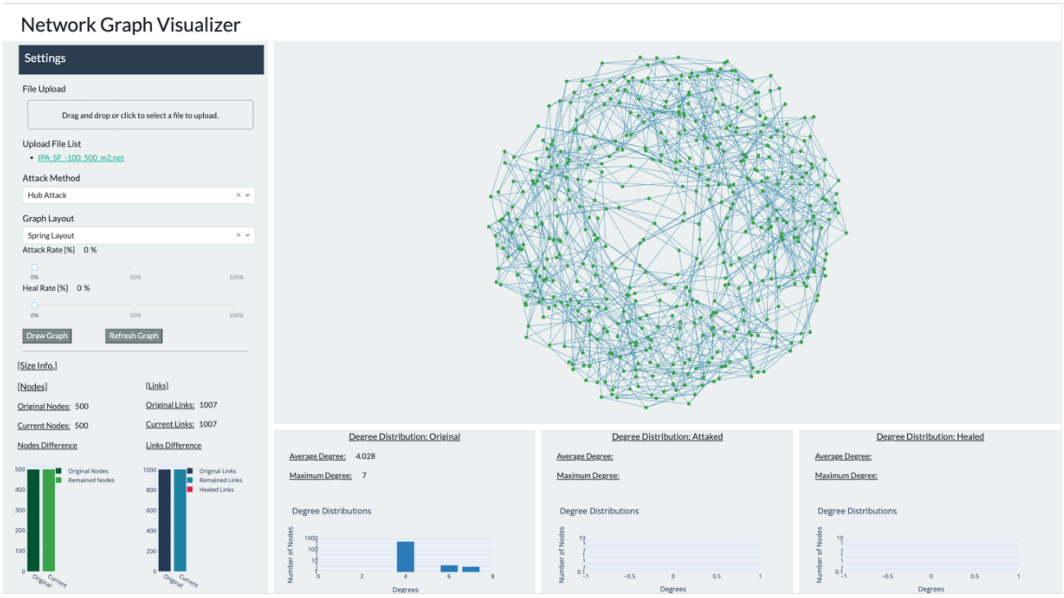

図 4.20: IPA モデルのネットワーク図 (攻撃率 0%、修復率 0%)

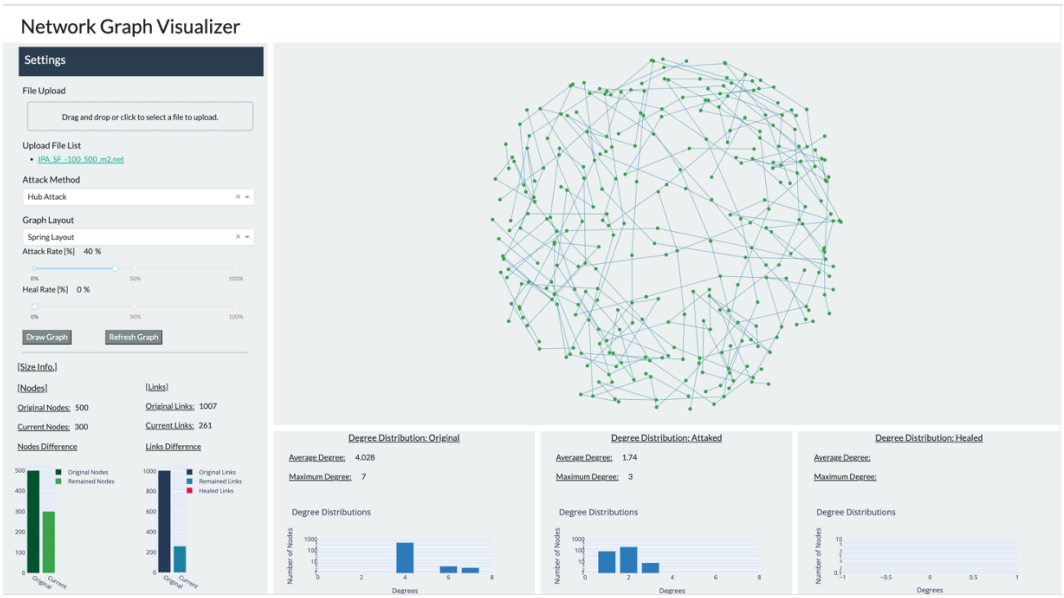

図 4.21: IPA モデルのネットワーク図 (攻撃率 40%、修復率 0%)

図 4.22: IPA モデルのネットワーク図 (攻撃率 50%、修復率 0%)

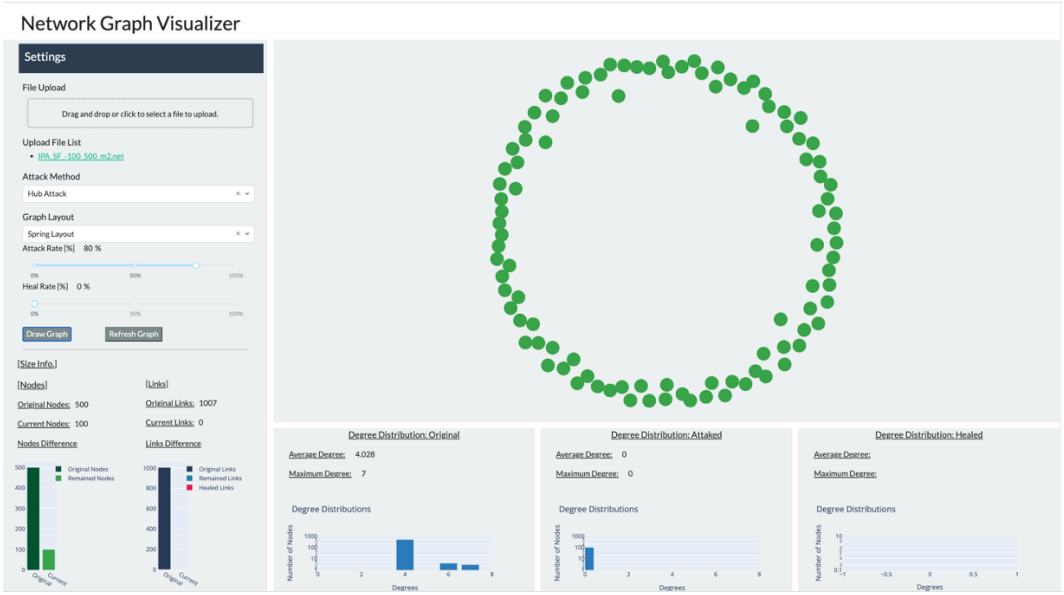

図 4.23: IPA モデルのネットワーク図 (攻撃率 80%、修復率 0%)

図 4.24: IPA モデルのネットワーク図 (攻撃率 80%、修復率 10%)

図 4.25: IPA モデルのネットワーク図 (攻撃率 80%、修復率 100%)

第5章　おわりに

本課題研究では、現実のネットワークや人工ネットワークに対して攻撃や修復を行い、それを対話的に可視化するというネットワークシミュレーターについて研究を行なった。複数のネットワークを様々な攻撃率・修復率で対話的に可視化して得られた結果を以下にまとめる。

- 複数のパラメータ (攻撃率、修復率) を操作する対話的な可視化は、従来の計算結果を都度グラフする方法より容易であり、ネットワークシミュレーターとして有用である
- 現実ネットワークに対しては、描画アリゴリズムを用いるよりも座標データを用いる方が直感的に攻撃による破壊状況や修復に伴う新たなネットワーク構造を理解することができる。一方、ノード数が数百を超えてくると、関係性を目視で全てを確認することは難しくなる
- 可視化にかかる時間は、攻撃・修復アルゴリズムよりも描画アリゴリズムの計算時間に影響される部分がある
- 同一ノード・リンク数でもネットワーク構造が異なると、描画アリゴリズムによっては計算時間に有意な差が発生する

今回、描画アルゴリズムの計算は、NetworkX という Python ライブラリで行ったが、数千ノード・数万リンクのようなネットワークの可視化には、対話的とは言えない時間が必要となることが予想される。ネットワーク攻撃同様、描画アルゴリズムについても、C++等の比較的高速に計算できる言語で、関数化することで計算時間を短縮するといった検討が今後必要であると考えられる。

参考文献

[1] A-L.Barabási, and R.Albert. "Emergence of scaling in random networks." Science 286.5439 (1999): 509-512.

[2] R.Albert, H.Jeong, and A-L.Barabási. "Error and attack tolerance of complex networks." Nature 406.6794 (2000): 378-382.

[3] C.Folke. "Resilience: The emergence of a perspective for social–ecological systems analyses." Global environmental change 16.3 (2006): 253-267.

[4] D.Callaway, et al. "Network robustness and fragility: Percolation on random graphs." Physical Review Letters 85.25 (2000): 5468.

[5] S.Mugisha, and Hai-Jun Zhou. "Identifying optimal targets of network attack by belief propagation." Physical Review E 94.1 (2016): 012305.

[6] A-L.Barabási, R.Albert, and H.Jeong. "Mean-field theory for scale-free random networks." Physica A: Statistical Mechanics and its Applications 272.1-2 (1999): 173-187.

[7] T.Tanizawa, S.Havlin, and H.E.Stanley. "Robustness of onionlike correlated networks against targeted attacks." Physical Review E 85.4 (2012): 046109.

[8] F.Liao, and Y.Hayashi. "Emergence of robust and efficient networks in a family of attachment models." Physica A: Statistical Mechanics and its Applications 599 (2022): 127427.

[9] NetworkX `https://networkx.org/`

[10] Dash `https://dash.plotly.com/`

[11] Matplotlib `https://matplotlib.org/`

[12] Cytoscape `https://cytoscape.org/`

[13] A.Braunstein, et al. "Network dismantling." Proceedings of the National Academy of Sciences 113.44 (2016): 12368-12373.

[14] F.Morone, and H.A.Makse. "Influence maximization in complex networks through optimal percolation." Nature 524.7563 (2015): 65-68.

[15] L.K.Gallos, and N.H.Fefferman. "Simple and efficient self-healing strategy for damaged complex networks." Physical Review E 92.5 (2015): 052806.

[16] M.Stippinger, and J.Kertész. "Enhancing resilience of interdependent networks by healing." Physica A: Statistical Mechanics and its Applications 416 (2014): 481-487.

[17] M.Chujyo, and Y.Hayashi. "A loop enhancement strategy for network robustness." Applied Network Science 6.1 (2021): 1-13.

[18] M.Chujyo, and Y.Hayashi. "Adding links on minimum degree and longest distance strategies for improving network robustness and efficiency." Plos one 17.10 (2022): e0276733.

[19] Y.Hayashi, A.Tanaka, and J.Matsukubo. "More tolerant reconstructed networks using self-healing against attacks in saving resource." Entropy 23.1 (2021): 102.

[20] J.Kim, and Y.Hayashi. "Distributed Self-Healing for Resilient Network Design in Local Resource Allocation Control." Frontiers in Physics (2022): 272.

[21] P.Mi, et al. "Interactive graph layout of a million nodes." Informatics. Vol. 3. No. 4. MDPI, 2016.

[22] T.M.J.Fruchterman, and E.M.Reingold. "Graph drawing by force - directed placement." Software: Practice and experience 21.11 (1991): 1129-1164.

[23] S.Zellmann, M.Weier, and I.Wald. "Accelerating force-directed graph drawing with rt cores." 2020 IEEE Visualization Conference (VIS). IEEE, 2020.

[24] B.Shneiderman. "The eyes have it: A task by data type taxonomy for information visualizations." The craft of information visualization. Morgan Kaufmann, 2003. 364-371.

[25] N.Ahmed, and R.Rossi. "Interactive visual graph analytics on the web." Proceedings of the International AAAI Conference on Web and Social Media. Vol. 9. No. 1. 2015.

[26] 渡辺聖史, 現実ネットワークにおける対話的な可視化に関する研究, 先端科学技術研究科, 北陸先端科学技術大学院大学, 2022.

[27] R.Inkret, A.Kuchar, and B.Mikac. "Extended final report of COST Action 266-advanced infrastructure for photonic networks." Faculty of Electrical Engineering and Computing, University of Zagreb (2003).

[28] J.Tapolcai. "Routing algorithms in survivable telecommunication networks." (2004).