14. Network モデル

2020/1/27

1　はじめに

今回は、ネットワーク (グラフ) をプログラムとして実装するとともに、ランダムネットワークを構成しましょう。今回は、無向ネットワーク、つまり辺に向きの無いものを考えましょう。

シミュレーションの準備として、NetBeans を開き、サンプルプログラムを取得します。また、必要に応じて、ライブラリとして、微分方程式の際に使用した MyLib を登録します。

`https://github.com/modeling-and-simulation-saga/Network`

2　Node クラスと Edge クラス

ネットワーク (グラフ) は、頂点 (Node) を向きの無い辺 (Edge) で結んだものです。これらを表すクラスが `network` パッケージ内の `Node` クラスと `Edge` クラスです。構成後のネットワークの探索などを行えるためには、 頂点に接続している辺と、辺の両端の頂点が分かる必要があります。

表 1　`Node` クラスのコンストラクタ

コンストラクタと説明
`public Node(String label)` ラベル `label` を与えて初期化する。

表 2　`Node` クラスのメソッド (一部)

修飾子と型	メソッドと説明
`void`	`addEdge(Edge edge)` 頂点に辺 `edge` を接続する。
`List<Edge>`	`getEdges()` 頂点に接続している辺の一覧を返す。
`boolean`	`isNeighbour(Node node)` 指定した頂点 `node` が隣接頂点ならば `true` を返す。
`List<Node>`	`neighbours()` 隣接頂点の一覧を返す。
`void`	`removeEdge(Edge edge)` 頂点から辺 `edge` を切り離す

表 3　Edge クラスのコンストラクタ

コンストラクタと説明
public Edge(String label) ラベル label を与えて初期化する。

表 4　Edge クラスのメソッド (一部)

修飾子と型	メソッドと説明
void	addEnd(Node node) 辺に node を付ける。
List<Node>	getEnds() 辺の端点の頂点の一覧を得る。
Node	getEnd(Node node) 辺の node と反対側の頂点を得る。
boolean	hasEnd(Node node) node が辺の端点であれば true を返す。

3　AbstractNetwork クラス

ネットワークを構成するための基本的なクラスが、network パッケージ内の AbstractNetwork です。このクラスは具体的なネットワーク構造を有しない抽象クラスとして定義しておきます。

表 5　AbstractNetwork クラスのコンストラクタ

コンストラクタと説明
public AbstractNetwork(String label) ラベル label を与えて初期化する。

表 6　AbstractNetwork クラスのメソッド (一部)

修飾子と型	メソッドと説明
void	addNode(Node node) ネットワークに頂点 node を追加する。
Edge	connectNodes(Node n1, Node n2) 二つの頂点 n1 と n2 を結ぶ新たな辺を生成し、その辺を返す。ラベルは自動的に生成する。
Edge	connectNodes(Node n1, Node n2, String label) 二つの頂点 n1 と n2 を結ぶ新たな辺をラベル label を指定して生成し、その辺を返す。
abstract void	createNetwork() 実際にネットワークを構成する抽象メソッド。

クラス AbstractNetwork の main には、簡単な例として 3 頂点の完全ネットワーク (図 1) を記載しています。通常、抽象クラスのインスタンスを生成することはできません。しかし、ソースファイル 3.1 に示すように、抽象クラス AbstractNetwork のインスタンス network を生成する際に、抽象メソッド createNetwork を実装することで、インスタンスを作成することができます。この中では、最初に頂点を 3 個生成した後、

`connectNodes` メソッドを用いて二つの頂点を結び付けています。最後に、pajek というネットワークを描画するアプリケーション用のデータをファイルに出力しています（ソースコード 3.2）。

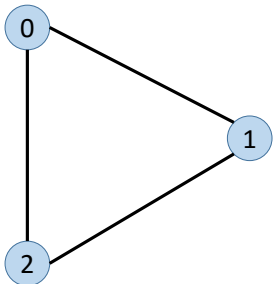

図 1　3 頂点の完全ネットワーク

ソースコード 3.1　`AbstractNetwork` クラスの `main` メソッドに記載された、3 頂点の完全ネットワークの例

```
1     static public void main(String args[]) throws IOException {
2         //抽象クラスのインスタンスを生成
3         AbstractNetwork network = new AbstractNetwork("testNetwork") {
4             //実装の無いメソッドを具体的に記述
5             @Override
6             public void createNetwork() {
7                 int n = 3;
8                 for (int i = 0; i < n; i++) {
9                     addNode(new Node(String.valueOf(i)));}
10                connectNodes(nodes.get(0), nodes.get(1));
11                connectNodes(nodes.get(1), nodes.get(2));
12                connectNodes(nodes.get(2), nodes.get(0));
13            }
14        };
15        network.createNetwork();
16        //pajek 用データを出力
17        NetworkFile.outputPajekData(network.getLabel() + ".net", network);
18    }
```

ソースコード 3.2　`AbstractNetworkMain` の出力。3 頂点の完全ネットワークの例

```
1 *Vertices 3
2 1 "0"
3 2 "1"
4 3 "2"
5 *Edges
6 1 2
7 1 3
8 2 3
```

ソースコード 3.2 の 1 行目は、頂点が 3 つあることを示しています。2 行目から 4 行目では、三つの頂点に名前を付けています。Pajek の中では、頂点の番号は 1 から始まることに注意します。6 行目以降は、頂点間を結ぶ辺の情報です。6 行目では、頂点番号 1 と 2 が結ばれていることを表しています。

課題 1　クラス `AbstractNetwork` の `main` を書き換えて、図 2 のネットワークを生成しなさい。結果は、生成された `.net` ファイルの中身により確認しなさい。

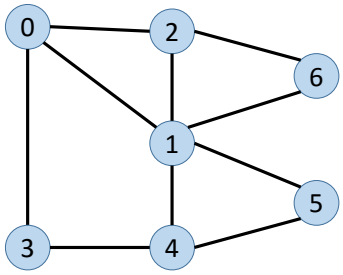

図 2　サンプルネットワーク

4　Erdös-Rényi のランダムネットワーク

講義で扱った Erdös-Rényi のランダムネットワークを実際に構成しましょう。頂点数を N、辺の総数を L とします。また、頂点数 N の完全グラフにおける辺の数との比を p とします。

$$L = p\frac{N\,(N-1)}{2} \tag{4.1}$$

ランダムネットワークは、でたらめに頂点の組を L 個選び辺を生成することで構成することができます。

表 7　`ER` クラスのコンストラクタ

コンストラクタと説明
`public ER(int n, int numEdges)` 頂点数 n と辺の数 `numEdges` を与えて初期化する。

ランダムネットワークのクラス `ER` を抽象クラス `AbstractNetwork` の拡張として定義します。配布した `networkModels` パッケージの下に含まれています。ネットワークを構成するには、メソッド `createNetwork` を用います。このメソッドは、さらに二つの `private` メソッドを呼びます。メソッド `createNodes` は、`n` 個の頂点を生成します。頂点はリスト `nodes` に保存されます。メソッド `createEdges` は、リスト `nodes` からでたらめに二つの頂点を選び辺を生成することを `numEdges` 回繰り返します。

課題 2　メソッド `createEdges` を実際に記述しなさい。実行すると `er.net` ファイルが生成されます。概ねでたらめに辺が生成されていることを確認しなさい。

課題 3　`analysis` パッケージ内の `GiantCluster` クラスを実行すると、p の値を変えながら ER ネットワークを構成し、講義で説明した最大クラスタサイズを計算します。その結果は `GiantClusterSize.txt` に書き込まれます。p と最大クラスタサイズの関係を作図しなさい。