

## Artisoc によるNF (Newman Fast) アルゴリズムコミュニティ抽出

末 木 俊 之\*

### NF (Newman Fast) Algorithm Community Extraction with Artisoc

Toshiyuki SUEKI\*

#### 要約

2012年度研究紀要<sup>(1)</sup>で使用した artisoc textbook<sup>(2)</sup>で作成したコミュニティ抽出シミュレーションは、一般的なパソコン環境で、ノード数200・リンク数2000程度のネットワークで1時間ほど計算時間がかかった。より大きなネットワークで気軽にコミュニティ抽出を体験できるシミュレーション作成を目指し、より高速な Newman Fast アルゴリズムに基づくシミュレーションを作成した。結果として、ノード数2000個のネットワークで2時間弱でコミュニティ抽出が可能となった。リンク数は計算時間に目立った影響はなく、リンクが多いネットワークでも気軽に使えるものである。

キーワード：Girvan-Newman コミュニティ抽出 モジュラリティ (modularity)切断法 結合法

#### 1. Newman Fast 法コミュニティ抽出概要

2012年度研究紀要<sup>(1)</sup>では、artisoc textbook<sup>(2)</sup>で作成した Girvan-Newman コミュニティ抽出シミュレーションを使いノード数200個ほどの『Barabasi-Albert モデル』のコミュニティ抽出を行ってみた。およそ総リンク数2000程度のネットワークでは、昨今の一般的なパソコン環境で1時間程の計算時間を必要とした。Girvan-Newman 法では、各計算ステップで全てのノードを起点とし、全てのリンクを辿り辺の媒介中心性を計算するため、リンク数の増加に伴い計算時間も急激に増加する。artisoc textbook はネットワークの表現、ネットワーク上でのシミュレーションをパソコン上で気軽に体験できる便利なツールであるが、もう少しノード数・リンク数の多いネットワークでコミュニティ抽出を短時間で行えるシミュレーションを使いたいと考えた。

Girvan-Newman コミュニティ抽出法より高速なコミュニティ抽出アルゴリズムとしては、Newman Fast アルゴリズムがある。『ネットワーク科学の道具箱 つながりに隠れた現象をひもとく』<sup>(3)</sup>などを参考にできる。Girvan-Newman コミュニティ抽出法は「切断法」と、Newman Fast 法は「結合法」と呼ばれ、お互い逆の発想によるアルゴリズムである。Girvan-Newman 法では、各計算ステップで媒介中心性最大値を示す辺を削除することにより最初は全体で1つのコミュニティであるネットワークをコミュニティに分割する。そしてQ値(モジュラリティ<sup>(4)</sup>)が最大値を示す時点のコミュニティ分割を最適なものとする方法である。Newman Fast 法では、最初にネットワーク上の個々のノードを、それぞれ自分自身のみのメンバー1ノードのコミュニティとする。最初はノードの数だけコミュニティがある。そして各計算ステップでコミュニティを

\*人間健康学部 健康栄養学科

2つ選んで結合する。その場合Q値(モジュラリティ)の増分 $\Delta Q$ 値が最大値を示すコミュニティのペアを選択する。 $\Delta Q$ 値がマイナス値となるステップでコミュニティ結合を止めるとその1つ前のコミュニティ分割が最適なものとなる。

コミュニティ*i*と*j*を結合した結果のQ値(モジュラリティ)の増分 $\Delta Q$ は、(式1.)で計算できる。

$$\Delta Q_{(i,j \rightarrow i+j)} = Q_{(i+j)} - (Q_i + Q_j) \quad (\text{式1.})$$

ここで、 $Q_i = e_{ii} - a_i^2$  及び  $Q_j = e_{jj} - a_j^2$ である。

また  $e_{ij} \stackrel{\text{def}}{=} n_{ij}/M$ 、 $a_i \stackrel{\text{def}}{=} \sum_j e_{ij}$  である。

$n_{ij}$ とは、リンクの始点ノードがコミュニティ*i*に属し、リンクの終点ノードがコミュニティ*j*に属しているリンクの数であり、 $M$ はグラフの次数(リンク総数)である。

$$Q_{(i+j)} = e_{ii} + e_{jj} + 2e_{ij} - \left(\frac{k_i + k_j}{M}\right)^2 = Q_i + Q_j + 2e_{ij} - 2a_i a_j \quad (\text{式2.})$$

ここで  $k_i$ 、 $k_j$  は、それぞれコミュニティ  $i$ 、 $j$ のリンク総数を意味している。

$$\left(\frac{k_i + k_j}{M}\right)^2 = \left(\frac{k_i^2}{M^2} + \frac{2k_i k_j}{M^2} + \frac{k_j^2}{M^2}\right) = a_i^2 + 2a_i a_j + a_j^2 \text{ と変形できるので}$$

$$Q_{(i+j)} = e_{ii} - a_i^2 + e_{jj} - a_j^2 + 2e_{ij} - 2a_i a_j = Q_i + Q_j + 2e_{ij} - 2a_i a_j \text{ となる。}$$

従って、最終的に(式1.)から

$$\Delta Q_{(i,j \rightarrow i+j)} = Q_i + Q_j + 2e_{ij} - 2a_i a_j - (Q_i + Q_j) = 2(e_{ij} - a_i a_j) \quad (\text{式3.})$$

が導かれる。

各計算ステップでは、コミュニティペア( $i, j$ )について、コミュニティ*i*からコミュニティ*j*へのリンク総数と、コミュニティ*i, j*それぞれの次数(総リンク数)のデータさえあれば、その2つのコミュニティを結合した場合のQ値(モジュラリティ)の増分 $\Delta Q$ 値を計算できる。この計算は、基本的に隣接行列あるノードが隣接しているノードが分かる情報を持つ行列があれば計算できる。使用メモリ容量は大きくなるが、計算処理自体は簡単である。最大 $\Delta Q$ 値を示すコミュニティペア( $i, j$ )における $\Delta Q$ 値がマイナスになったタイミングが計算処理を止めるタイミングとなるので、計算終了のタイミングは明確である。計算終了後はすぐに、コミュニティ分割結果を画面で確認できる。

2011年度研究紀要<sup>(5)</sup>で作成した Girvan-Newmanコミュニティ抽出シミュレーションでは、

各計算ステップでノードを表すエージェントとリンクを表すエージェントをリンク(つながり)を辿り何度も参照する計算処理があったが、今回の Newman Fastアルゴリズムでは、エージェントを参照する必要はない。単純に配列(隣接行列またはそれに類似したもの)上のデータを参照する計算だけで済む。このことから計算速度も格段に速くなることが予想される。

## 2. Newman Fast法 artisocシミュレーションの概要

シミュレーションにおいては、2010年度研究紀要<sup>(6)</sup>からの一連のシミュレーションと同様に、ノードは nodeエージェント、リンクは linkエージェントによって実現する。コミュニティ抽出計算完了後は、ノードを円内に配置して表示する。重要度の

高いノードをより円の中心に表示し、各 node エージェントに付与されるコミュニティ ID の値により、同一コミュニティ ID を持つノードは円内の同一扇型領域内に表示する。各ノードの媒介中心性指数は計算されないで、ノードの重要度は次数(リンクの数)で判定する。

コミュニティ抽出計算中には、node エージェント、link エージェントを参照する必要は一切ない。

計算に必要な情報は、各コミュニティの持つ全リンク総数、同一コミュニティに属するノード同士をつなぐリンク総数と、個々のノードがリンクするリンク先ノードに関する情報であるが、これらは artisoc で使える Universe 変数(グローバルに使う変数)を用いて、(表 1.) に示す 3 つの配列情報として保持して使用することにした。

変数名	型	用途
コミュニティ	二次元配列長整数型	<ul style="list-style-type: none"> <li>・コミュニティに関する情報を保存する二次元整数型変数。</li> <li>・一次元目の配列数は、現状5000。 二次元目の配列数は2。</li> <li>・Newman Fast法では、最初個々のノードを、自分自身のみがメンバーのコミュニティ所属とする。各ノードには別々のコミュニティIDが付与される。今回作成するシミュレーションで扱うモデルは、最大ノード数を4999(コミュニティIDが1~4999)と仮定して作成している。</li> <li>・コミュニティ(<math>i, 1</math>)...コミュニティID = <math>i</math> であるコミュニティに所属するノードのリンクの総数。</li> <li>・コミュニティ(<math>i, 0</math>)...コミュニティID = <math>i</math> であるコミュニティに所属するノード同士をつなぐリンク総数。</li> <li>・コミュニティ(<math>i, 1</math>) = 0 の場合は、コミュニティID = <math>i</math> であるコミュニティが存在しない(コミュニティ併合により消滅した)ことを意味する。</li> </ul>
コミュニティ間 リンク数	整数型	<ul style="list-style-type: none"> <li>・5000×5000の巨大な二次元配列。</li> <li>・コミュニティ間リンク数(<math>i, j</math>) は、コミュニティID = <math>i</math> に属するノードが始点でコミュニティID = <math>j</math> に属するノードが終点となるリンクの総数を保持する。</li> </ul>
ノードテーブル	整数型	<ul style="list-style-type: none"> <li>・5000×5000の巨大な二次元配列。 最初-99に初期化される。 隣接行列と類似の配列として使用。</li> <li>・ノードテーブル(<math>i, 0</math>)...ノードID = <math>i</math> であるノードの所属するコミュニティのコミュニティIDを保持する。</li> <li>・ノードテーブル(<math>i, j</math>) (<math>j=1\sim 4999</math>)...ノードID = <math>i</math> であるノードがリンクしているノードの所属するコミュニティのコミュニティIDを保持する。</li> </ul>

(表 1.) コミュニティとノードのリンクに関する情報を保持するための変数

今回作成したシミュレーションは最大数4999のノードを扱うことを想定して作成した。Newman Fast法では、初期状態では各ノードを自己1つの

ノードだけから構成されるコミュニティの所属とする。シミュレーション開始時に各ノードは、コミュニティ ID = ノード ID に初期化される。

シミュレーション開始時に、まず(表1). の3つの配列変数の初期化を行う。nodeエージェント、linkエージェントの保持する情報に基づき3つの配列変数に初期情報をセットする。

(図1.)はノードテーブルの初期状態の例を示す。ノードテーブル(x,0)には、ノード ID=xの所属するコミュニティ IDが格納される。初期状態では、個々のノードの所属するコミュニティのコミュニティ IDとノード IDは一致している。これは node エージェント情報をエクセルシート上で用意する時点で、コミュニティ ID=ノード IDとしてデータ作成しているためである。ノードテーブル(x,1) ~ノード

テーブル(x,4999)の要素は、ノード ID=xのノードがリンクしている相手ノードの所属するコミュニティのコミュニティ IDがセットされる。(図1.)の例では、各行の先頭列にはノードの所属するコミュニティ IDが書き込まれている。2列目以降の列は、ノードがリンクする相手ノードの所属するコミュニティ IDが書き込まれている。-99が書き込まれている列以降は、未使用の列(それ以上リンクするノードが無い)である。(図1.)でノード ID=2のノードは、2つのノードとリンクしていて、それぞれのノードの所属するコミュニティ IDは、5と10であることが分かる。

→リンク先ノードの所属するコミュニティの ID

配列要素番号	0	1	2	3	4	...	4997	4998	4999
ノード ID 0	-99	-99	-99	-99	-99	...	-99	-99	-99
1	1	4	-99	-99	-99	...	-99	-99	-99
2	2	5	10	-99	-99	...	-99	-99	-99
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
4997	4997	-99	-99	-99	-99	...	-99	-99	-99
4998	4998	-99	-99	-99	-99	...	-99	-99	-99
4999	4999	-99	-99	-99	-99	...	-99	-99	-99

(図1.) ノードテーブルの初期状態の例

Newman Fast法による各計算ステップでは、全てのコミュニティペアについて、コミュニティを仮結合した場合のQ値(モジュラリティ)の増分 $\Delta Q$ を計算し、 $\Delta Q$ 値が最大となるコミュニティペア(Com1,Com2)を見つける。このシミュレーションはノード間に必ず双方向リンクが存在することを仮定しているため、コミュニティ間リンク数(Com1,Com2)=コミュニティ間リンク数(Com2,Com1)となる。コミュニティペア探索では、(Com1,Com2)ペア(Com1=1~4999, Com2= Com1+1~4999)について計算を行うので、コミュニティ間リンク数変数も、コミュニティ間リンク数(Com1,Com2) (Com1=1~4999, Com2= Com1+1~4999)部分のみが参照されている。

$\Delta Q$ 値は、(式3.)に基づいて、コミュニティ

変数と、コミュニティ間リンク数変数上の情報のみを使用し簡単に計算できる。最大 $\Delta Q$ 値がマイナスになったら計算を終了する。 $\Delta Q$ 値が0より大で最大値を示すコミュニティペア(Com1,Com2)が見つかった場合には、コミュニティ ID= Com2のコミュニティをコミュニティ ID= Com1のコミュニティに併合する処理を行う。

コミュニティ併合処理は、以下の考え方に基づいて作成した。

(1) ノードテーブルについての更新処理

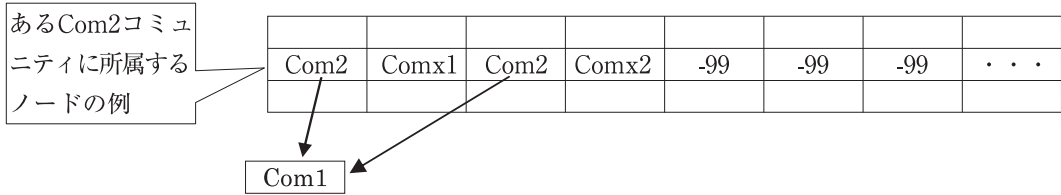
- a) ノード ID=xのノードが、コミュニティ ID= Com2のコミュニティに属しているならば、所属コミュニティを Com1に更新する。

具体的には、(図2.)に示すように、ノード

テーブル(x,0) = Com2ならば、Com1に更新する。

b) リンク先ノードの所属するコミュニティのIDが Com2ならば、Com1に更新する。

具体的には、(図2.)に示すように、ノードテーブル(x,y) = Com2 (y=1~4999)ならば、Com1に更新する。



(図2.) ノードテーブルの初期状態の例

さらに Com1コミュニティに併合されることにより消滅する Com2コミュニティに属するノードのリンク先ノードの所属するコミュニティIDに基づき以下の(2)、(3)の更新処理を実行する。具体的には、(図2.)の2行目の例のように、ノードテーブル(x,0) = Com2である行の2列目~5000列目、ノードテーブル(x,1) ~ノードテーブル(x,4999)のリンク先ノードコミュニティID情報に基づいて以下の(2)、(3)の処理を行う。

(2) コミュニティテーブルについての更新処理

a) Com2コミュニティの消滅処理  
コミュニティ (Com2,1) = 0とする。

b) コミュニティのリンク総数の増減については、リンク先ノードコミュニティIDに依らず、以下の処理を行う。

Com1のコミュニティリンク総数を1つ増やす。

Com2のコミュニティリンク総数は1つ減るが、Com2コミュニティはCom1コミュニティに併合されて消滅するので処理は不要である。

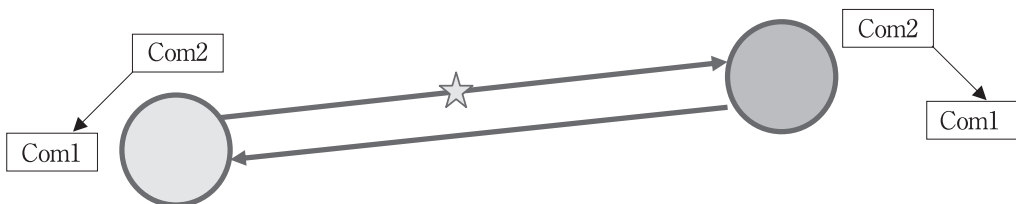
c) Com1・Com2のコミュニティのコミュニティ内リンク数の増減について

① リンク先ノードが Com2コミュニティ所属のケース

リンク先ノードも Com1コミュニティ所属に変わるので Com1のコミュニティ内リンク数が1つ増える。

この処理は(図3.)での☆印付き矢印のリンクに対応する処理である。

Com2のコミュニティ内リンク総数が1つ減るが、Com2のコミュニティはCom1のコミュニティに併合されて消滅するので処理は不要である。

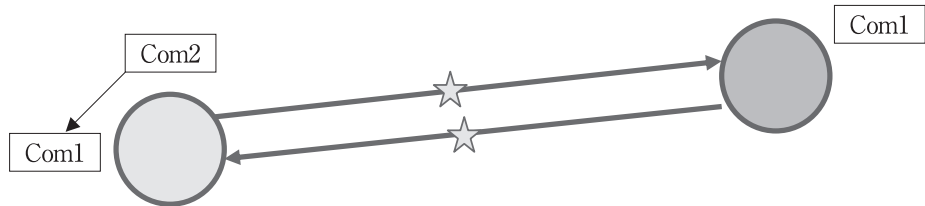


(図3.) Com1・Com2のコミュニティのコミュニティ内リンク数の増減について (ケース1)

② リンク先ノードが Com1コミュニティ所属のケース

Com1のコミュニティ内リンク数が2つ増える。

この処理は(図4.)での☆印付き矢印のリンクに対応する処理である。  
Com2のコミュニティ内リンク数について



(図4.) Com1・Com2のコミュニティのコミュニティ内リンク数の増減について(ケース2)

(3)コミュニティ間リンク数テーブルの更新処理について

①リンク先ノードがCom2コミュニティ所属(Comx = Com2)のケース

・以下2つの処理を行う必要がある。

Com2 ⇒ Comx (Com2) Com2からCom2へのリンク数が1つ減ることになる。ただし、Com2のコミュニティはCom1のコミュニティに併合されて消滅するので処理は不要となる。

Com1 ⇒ Com1 Com1からCom1へのリンク数が1つ増える。この処理は

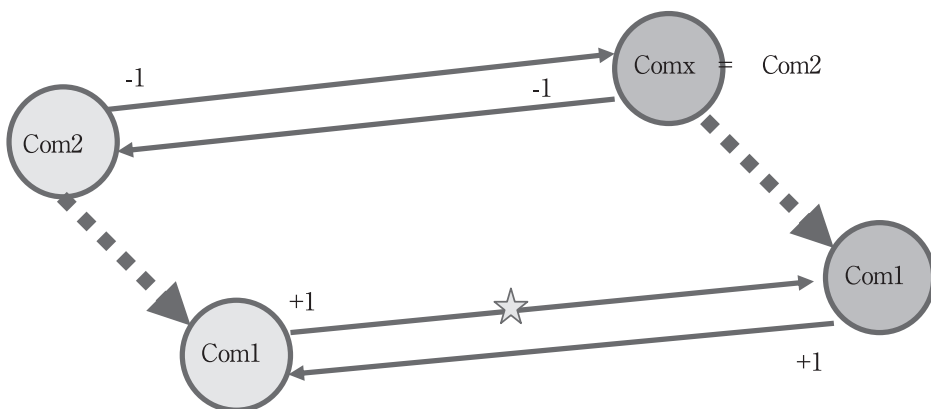
は、Com2のコミュニティがCom1のコミュニティに併合されて消滅するので処理は不要である。

(図5.)での☆印付き矢印のリンクに対応する処理である。

・以下2つの処理は、リンク相手先ノードの行うべき処理なので不要となる。

Comx (Com2) ⇒ Com2 Com2からCom2へのリンク数が1つ減ることになる。ただし、Com2のコミュニティはCom1のコミュニティに併合されて消滅するので処理は不要となる。

Com1 ⇒ Com1 Com1からCom1へのリンク数が1つ増える。



(図5.) コミュニティ間リンク数テーブルの更新処理について(ケース1)

②リンク先ノードがCom2コミュニティ所属でない(Comx ≠ Com2)ケース Com1 ⇒ Comx Com1からComxへのリンク数が1つ増える。

Comx ⇒ Com1 ComxからCom1へのリンク数が1つ増える。

この処理は(図6.)での☆印付き矢印のリンクに対応する処理である。

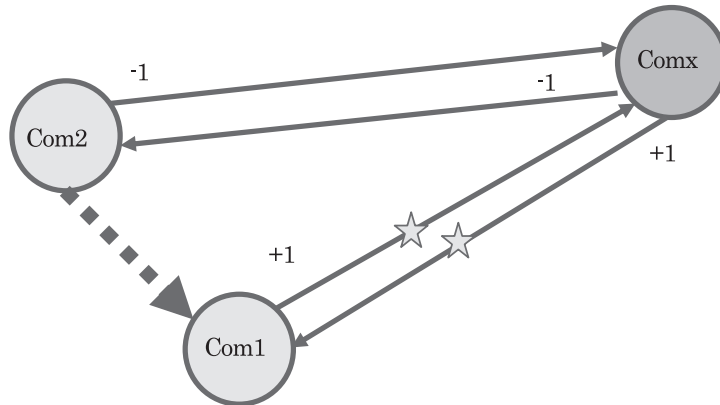


・以下2つの処理は、Com2のコミュニティはCom1のコミュニティに併合されて消滅するので処理は不要である。

Com2 ⇒ Comx Com2からComxへ

のリンク数が1つ減る。

Comx ⇒ Com2 ComxからCom2へのリンク数が1つ減る。



(図6.) コミュニティ間リンク数テーブルの更新処理について (ケース2)

### 3. Newman Fast法 artisocシミュレーション抽出結果

#### 3.1. ノード数1000、m0=0、m=1 BAモデルコミュニティ抽出結果

およその計算時間を知るため、まずノード数1000、m0=0、m=1 BAモデルでコミュニティ抽出を実行した。全ノード数1000、リンク総数1998のモデルである。

結果、計算時間14分、コミュニティ数=36、Q値=0.9207971735499266であった。

シミュレーション開始時に、まず(表1.)の3つの配列変数を初期化するが、この処理だけで約1分かかった。Artisocの5000×5000の巨大な Universe変数(グローバルに使う変数)にアクセスするのは時間がかかるようだ。

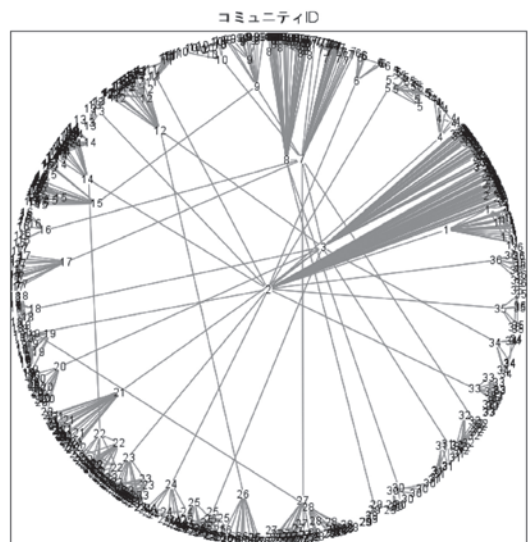
初期状態では、1000個のコミュニティが存在する。各計算ステップ最大ΔQ値を示すコミュニティペアを探し、その2つのコミュニティを併合する処理で1つずつコミュニティが減るので各ステップの計算所要時間は徐々に減少していった。

最初1000個のコミュニティが存在するステップでは、2秒ほど計算時間がかかった。

700個の程度のコミュニティが存在するステップでは、1秒ほどの計算時間に減少した。

400個の程度のコミュニティが存在するステップでは、0.5秒ほどになった。

コミュニティ抽出の結果として得られたネットワーク図として、各ノードの所属するコミュニティIDを表示する(図7.)を掲載する。



(図7.)ノード数1000、m0=0、m=1 BAモデルコミュニティ抽出結果(コミュニティID表示図)

### 3.2. ノード数1000、 $m_0=0$ 、 $m=3$

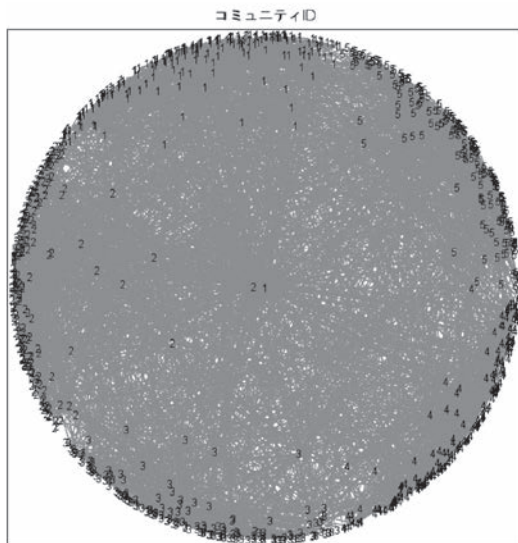
#### BAモデルコミュニティ抽出結果

つぎにノード数1000、 $m_0=0$ 、 $m=3$  BAモデルでコミュニティ抽出を実行した。全ノード数1000、リンク総数 5988のモデルである。

結果、計算時間13分、コミュニティ数=5、Q値= 0.36752211838506677であった。

3.1のモデルとノード数は同じで、リンク総数は約3倍だが計算時間はほとんどは変わらない。

コミュニティ抽出の結果として得られたネットワーク図として、各ノードの所属するコミュニティIDを表示する(図8.)を掲載する。



(図8.)ノード数1000、 $m_0=0$ 、 $m=3$  BAモデルコミュニティ抽出結果(コミュニティID表示図)

### 3.3. ノード数2000、 $m_0=0$ 、 $m=1$

#### BAモデルコミュニティ抽出結果

つぎにノード数2000、 $m_0=0$ 、 $m=1$  BAモデルでコミュニティ抽出を実行した。全ノード数2000、リンク総数 3998のモデルである。

結果、計算時間1時間46分、コミュニティ数=53、Q値= 0.946620133478449であった。

最初2000個のコミュニティが存在するステップでは、8秒ほどの計算時間がかかった。

1300個の程度のコミュニティが存在するステップでは、4秒ほどの計算時間に減少した。

1100個の程度のコミュニティが存在するステップでは、3秒ほどの計算時間に減少した。

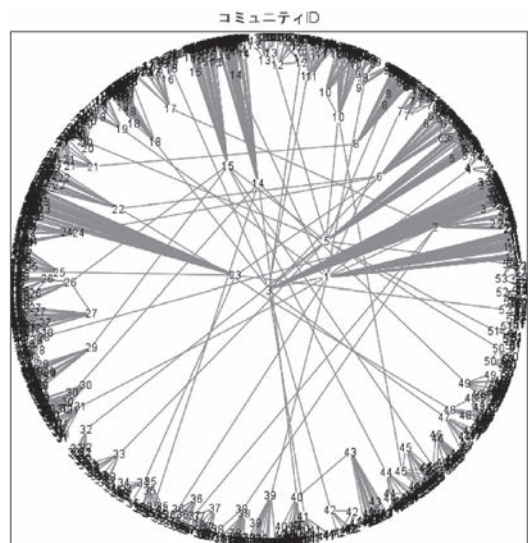
700個の程度のコミュニティが存在するステップでは、2秒ほどの計算時間に減少した。

500個の程度のコミュニティが存在するステップでは、1秒ほどの計算時間に減少した。

400個の程度のコミュニティが存在するステップでは、0.5秒ほどの計算時間に減少した。

コミュニティ抽出の結果として得られたネットワーク図として、各ノードの所属するコミュニティIDを表示する(図9.)を掲載する。

コミュニティ抽出の結果として得られたネットワーク図として、各ノードの所属するコミュニティIDを表示する(図9.)を掲載する。



(図9.)ノード数2000、 $m_0=0$ 、 $m=1$  BAモデルコミュニティ抽出結果(コミュニティID表示図)

## 4. 結び

今回は artisoc textbook(2)を使用して Newman Fast法コミュニティ抽出シミュレーションを作成した。シミュレーションの最初に全ての nodeエージェント、linkエージェント情



報を読み込み、その情報に基づいて3つのグローバル変数の初期化処理を行う。後の計算では全てグローバル変数データのみを使って計算できる。コミュニティペアから計算する $\Delta Q$ 値の最大値がマイナスになったら自動的に計算を終え、すぐに結果をネットワーク表示図で確認できる利点もあり Girvan-Newmanコミュニティ抽出シミュレーションと比べると作成は容易であった。nodeエージェント、linkエージェントはコミュニティ抽出が終わった後、ネットワークの表示で使われるだけである。

およその計算時間等を知るため、(表2.)に示した3つのBAモデル(Barabasi-Albertモデル)ネットワークを使いコミュニティ抽出を行ってみた。(表2.)1つ目のリンク数2000ほどのネットワークだと、Girvan-Newmanコミュニティ抽出シミュレーションでは、2時間ほど(1)の計算時間がかかった。

今回作成した Newman Fast法コミュニティ抽出シミュレーションでは約14分で、比較すると8分の1程度の短時間でコミュニティ抽出が可能となっている。また Girvan-Newmanコミュニティ抽出シミュレーションでは、計算時間はリンクの数に大きく依存していたが、今回の Newman Fast法コミュニティ抽出シミュレーションでは、ほとんどリンク数の増加による計算時間の増加が見られない。(表2.)2つ目のモデルでは、リンク数が6000ほどであるが、計算時間はほとんど1つ目のモデルと変わらない。Girvan-Newmanコミュニティ抽出シミュレーションでは数時間程度では計算が終わらないレベルのモデルである。おそらくリンク総数が万のオーダーのネットワークでもさほど計算時間は変わらないであろうことが想像できる。

m0	m	ノード数	リンク数	Q値	抽出されたコミュニティ数	コミュニティ抽出計算時間
0	1	1000	1998	0.9207971735499266	36	約14分
0	3	1000	5988	0.36752211838506677	5	約13分
0	1	2000	3998	0.946620133478449	53	1時間46分

(表2.) コミュニティ抽出結果

ただし、Newman Fast法コミュニティ抽出シミュレーションの場合は、ノード数の増加による計算時間の増大の影響が大きい。結合法では、初期にはノードの数だけコミュニティが存在する。そして計算の1ステップごとに選択された2つのコミュニティが併合され、コミュニティ数が1つ減少する。今回実行したモデルの計算終了時のコミュニティ数は、100以下の数なので、ほとんどノード数と同じ程度の計算ステップが実行されることになる。このことから計算時間は少なくともノード数に比例して増加するのは当然のことになる。

さらに $\Delta Q$ 値が最大となるコミュニティペアを見つけるための計算時間もノード数の増加に大きく影響される。ペアの数はノード数の二乗に比例する。各計算ステップで必ずこのペアを見つける計算が行われるので、1ステップごとの計算時間も

ノード数の増加によって増大する。今回のモデルのケースでは、2000個のコミュニティが存在するステップでは8秒程度、1000個のコミュニティが存在するステップでは、2秒程度の計算時間がかかった。2000個のコミュニティ数だと、コミュニティペアの数は $2000 \times 2000 = 4000000$ 、1000個のコミュニティ数だと、コミュニティペアの数は $1000 \times 1000 = 1000000$ となり、1000個から2000個にノード数が増えると約4倍計算時間が増加する。以上のことより、ノード数が1000から2000に増加すると、計算時間は少なくとも8倍程度増加することが想像される。実際に(表2.)の3つ目のモデルは1つ目のモデルの7~8倍程度計算時間がかかっている。

artisocで用意されている Universe変数(グローバルに使う変数)をアクセスするのは想像していた

より時間がかかるようだ。シミュレーションのスタート時、巨大な配列変数を初期化するだけで、1分程度の時間がかかっている。ノード数の増加に伴い使用する配列変数も巨大なものとなる。配列変数へのアクセスに時間がかからなければより多くのノードがあるモデルでもコミュニティ抽出が可能であろうが、昨今の一般的なパソコンを使って artisoc を用いてコミュニティ抽出とその後のネットワーク表示を気軽に体験するには、ノード数3000個のモデルあたりが限界であろうか。Newman Fast法コミュニティ抽出シミュレーションを使う場合は、リンク数の多さは計算時間に影響が少ない。Artisocでコミュニティ抽出を行う場合、ノード数3000程度のネットワークであれば、リンク数がかかなり多いモデルでも気軽に使えるであろうというのが結論である。

数計算とネットワーク表示について、駒沢女子  
大学研究紀要、p.375-394

#### [参考文献・出典]

- 1) 末木俊之(2012) Artisocによる BA (Barabasi-Albert) モデルコミュニティ抽出、駒沢女子大学研究紀要、p.279-293
- 2) 山影進(2007) 人工社会構築指南 artisocによるマルチエージェント・シミュレーション入門、有限会社書籍工房早山、東京
- 3) 林幸雄・大久保潤・藤原義久・上林憲行・小野直亮・湯田聴夫・相馬亘・佐藤一憲著(2007) ネットワーク科学の工具箱 つながりに隠れた現象をひもとく、p138-139、近代科学社、東京
- 4) モジュラリティ (modularity) は、この数値が大きいほど同じ属性を持つグループ(コミュニティ)に属するノード同士をつなぐリンクの割合が、異なる属性を持つノード同士をつなぐリンクの割合よりも多いこと示す。
- 5) 末木俊之(2011) Artisocで作成する Girvan-Newmanコミュニティ抽出シミュレーション、駒沢女子大学研究紀要、p.279-293
- 6) 末木俊之(2010) Artisocによる媒介中心性指