D-041

大規模社会ネットワークからのクラスタ構造の抽出

鶴見 敏行　　脇田 建

東京工業大学大学院情報理工学研究科
E-mail: {tsurumi2, wakita}@is.titech.ac.jp

概 要

Clauset，Newman，Moore はネットワークをボトムアップかつ貪欲に解析する手法 (CNM アルゴリズム) を提案し、50 万ノード程度までの社会ネットワーク解析を可能としたが、それ以上の規模については実用的な時間内での解析は困難であった。そこで CNM アルゴリズムの合併の過程を観察した。その結果合併するクラスタサイズの不均衡が計算コストに大きく影響していることが明らかとなった。この観測から、合併するクラスタサイズの均衡がアルゴリズムの速度向上につながると考えた。本稿では、合併時のクラスタのサイズを考慮することにより合併比率を向上させる 3 種類の手法を提案する。提案手法の実験データセットとして、国内最大級のソーシャルネットワーキングサービスより 2006 年 10 月に取得した 550 万ユーザーの友人関係のネットワークを使用した。提案した 3 つの手法を用いたところ、CNM アルゴリズムに比べ劇的なスケーラビリティの向上がみられた。もっとも速度向上がみられた手法では、100 万ノードに対して 5 分、400 万ノードに対しては 35 分程度で解析する事に成功した。また別の手法では、50 万ノードに対して 50 分（CNM アルゴリズムより 7 倍早い）で解析でき、モジュール性の向上にも成功した。

1　はじめに

クラスタ構造の抽出は、複雑な社会ネットワークからその固有の性質を知るうえでの最初の重要なステップであり、今までに数々の研究がなされている。[5, 11, 3, 7, 10, 6, 13, 2, 1, 9]

われわれは、Clauset，Newman，Moore によって提案されたアルゴリズム（CNM アルゴリズム）[2] を実装し、Social Networking Service(SNS) より取得したネットワークのさまざまな部分集合に対しての解析を試みた。50 万ユーザー以下での解析は可能だった。しかし、さらに大きなネットワークでは実用的な時間での解析は不可能であった。

そこで CNM アルゴリズムの合併の過程を観察した。その結果合併するクラスタサイズの不均衡が計算コストに大きく影響していることが明らかとなった。この観測から、合併するクラスタサイズの均衡がアルゴリズムの速度向上につながると考えた。

本稿では合併する合併の不均衡を抑えるために、クラスタサイズの比である**合併比率**を定義し、合併するクラスタの選定の基準として従来用いられてきた**モジュール性**に合併比率を加味するヒューリスティックを提案する。この提案を用いて実験を行った。その結果合併の不均衡を抑えることができ、数百万ノードのネットワークに対する解析も可能となった。

これ以後の本稿の構成は以下の通りである。2 節で関連研究について述べる。3 節で CNM アルゴリズムを紹介し、その問題点を明らかにする。4 節では CNM アルゴリズムの欠点を改善するヒューリスティックを提案し、5 節では提案手法を評価する。最後に 6 節で本稿をまとめる。

2　関連研究

クラスタを抽出する手法は大きく分けて二つある。一つは、グラフよりいくつかのノードを**種**として選び、それを含むクラスタを返す手法 [5, 11, 3, 7] である。もう一つは、ネットワーク全体をクラスタに分割する手法 [10, 6, 13, 2, 1, 9] である。

前者の手法は主にウェブの解析に使われている。ウェブの解析は、各ウェブページをノード、ハイパーリンクを**エッジ**とした有向グラフとしてウェブをとらえ、解析する。Kleinberg が提案した *HITS*[5, 4] は、ウェブの適当な部分グラフの中からハイパーリンクで密に結合された**オーソリティ**および**ハブ**を抽出するアルゴリズムである。オーソリティとは、多くのページからハイパーリンクを張られている著名なページである。ハブとは、リンク集およびブックマークなど、多くの著名なページへハイパーリンクを張っているページである。HITS は、簡単な反復計算によって各ウェブページのオーソリティおよびハブのスコアを計算する。

後者の手法としてさまざまな提案がなされている。Kumar らによる *Trawling*[6] は特に知られている。彼らは、2 億ページ以上の大規模なウェブのアーカイブに対してこの手法を適用し、10 万個を越えるクラスタの**コア**を発見した。コアとはオーソリティとハブから成るサイズの小さい**完全 *2* 部グラフ**であり，ほとんどのクラスタがコアを含むという仮定に基づいている。Clauset らにより提案されたアルゴリズム（CNM アルゴリズム）[2] は、本研究と関連が深いため、次節で説明する。

3　CNM アルゴリズム

Newman と Girvan は、グラフ G に対するクラスタリング C が与えられたときに、そのクラスタリングの優劣を評価するための指標として**モジュール性**$Q(G, C)$ を定義し [8]、モジュール性を指標として貪欲にクラスタを合併する手法を提案した。のちに Clauset，Newman，Moore らはそれにデータ構造上の改善を施した案を提案した (CNM アルゴリズム)[2]。

CNM アルゴリズムでは、まずそれぞれのノードを独立したクラスタとして表す。そして、全クラスタ対 (c_i, c_j) について、それらを合併したときのモジュール性の上昇

度 ($\Delta Q^{C}_{c_i,c_j}$) を計算する。ここで $\Delta Q^{C}_{c_i,c_j}$ の定義は以下の通りである。

$$\Delta Q^{\mathcal{C}}_{c_i,c_j} = Q(G, \mathcal{C}\backslash\{c_i, c_j\} \cup \{c_i, \cup c_j\}) - Q(G, \mathcal{C}).$$

CNM アルゴリズムは、最大値を保持する $\Delta Q^{C}_{c_i,c_j}$ を反復的に合併し新しいクラスタとする。この反復は $\Delta Q^{C}_{c_i,c_j}$ が正である限り続く。クラスタの合併とともにクラスタ数は単調に減少するため、このアルゴリズムはいずれ停止する。

アルゴリズム 1 CNM アルゴリズム [2] の概略

$\mathcal{C} := \{v \in V | \{v\}\}$;

function join(c_i, c_j) {
  return $\mathcal{C}\backslash\{c_i, c_j\} \cup \{c_i, \cup c_j\}$;
}

procedure updateDeltaQ() {
  $\forall c_i, c_j \in \mathcal{C}$.
    $\Delta Q^{\mathcal{C}}_{c_i,c_j} := Q(G, \mathrm{join}(c_i, c_j)) - Q(G, \mathcal{C})$;
}

while (**true**) {
  updateDeltaQ();
  $\Delta Q^{\mathcal{C}}_{c_i,c_j}$ が最大である $(c_i, c_j) \in \mathcal{C}^2$.
  if ($\max(\Delta Q^{\mathcal{C}}_{c_i,c_j} < 0)$ **break**;

  $\mathcal{C} := \mathrm{join}(c_i, c_j)$;
}

アルゴリズム 1 に、CNM アルゴリズムの概要を与えた。updateDeltaQ の操作の費用が高いように思えるが、$\Delta Q^{\mathcal{C}}_{c_i,c_j}$ の再計算を要するのは c_i または c_j のいずれかが直前に合併した場合だけであり、その計算はその時点における値を用いて $\Delta Q^{C}_{c_i,c_j}$ の値から差分的に計算できる。このため、実際は効率的に計算することができる。また、$\Delta Q^{\mathcal{C}}_{c_i,c_j}$ の最大値を与えるクラスタ対もプライオリティ・キューを利用することで簡単に見つけることができる。

Clauset らは、このアルゴリズムを用いて Amazon.com から得た数十万ノード規模のネットワークなどを解析した。

われわれは Clauset らの手法を Java によって実装した。データセットとして、わが国の代表的な SNS である mixi(http://mixi.jp/) の友人間関係（マイミクシィへの参照）のネットワークを用いた[1]。実験環境として、Intel Xeon 2.80GHz，L2 cache = 2MB，メモリ = 4GB を用いた。

実験の結果 [2] で示唆される高効率に反して予想外に解析時間がかかり、1 週間で全体の 10%程度の解析しかすることができなかった。

[1] 2005 年 11 月に取得した mixi より取得した約 100 万ユーザーからなるネットワーク

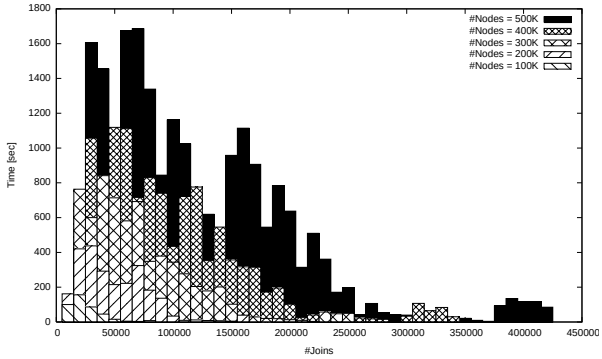

図 1　ネットワークの解析に要した時間の時系列遷移。

計算コストのボトルネックを調べるために、mixi ネットワークの様々な部分集合に対して解析を試みた。mixi はそれぞれのユーザーに対して、登録した順番に ID ナンバーを 1 から順番に与える。$U = \{1, 2, \ldots\}$ はユーザー ID の集合。$F \subset U \times U$ は友人関係の集合とすると、mixi ネットワークは、$G_{mixi} = (U, F)$ のようなグラフで表せる。このグラフの部分集合を以下のように定義する。

$$G^{n}_{mixi} = (U(n), F \cap (U(n) \times U(n)))$$
$$\text{ただし } U(n) = \{u \in U | u \leq n\}$$

図 1 は mixi ネットワークの様々な部分集合 $G^{N}_{mixi}(N = 100K, 200K, 300K, 400K, 500K)$ に対する、クラスタ抽出が完了するまでの解析に要した時間の時系列遷移である。横軸は累積の合併の回数、縦軸は 10,000 回の合併に要する時間を秒で表している。それぞれのデータセットでは、解析の前半で合併に要する時間がかかり、後半で劇的に減少していることが分かる。

[2] は CNM アルゴリズムの計算量が $O(md \log n)$ と述べている。ここで n はノード数、m は全エッジ数、d はデンドログラムの高さである。また、疎な社会ネットワークにおいて、m を n、d を $\log n$ と近似でき、計算量は $O(n \log^2 n)$ となると予想している。一方、われわれの実験した実行時間の結果である図 3 のデータに対して $f(x) = ax^b$ を最小二乗近似した結果、$a = 1.51 \cdot 10^{-8} \pm 2.0610^{-8}, b = 2.13 \pm 0.104$ と 2 次のオーダーで上昇することがわかった。これより $d \approx \log n$ の性質が少なくとも mixi ネットワークでは現れないことがわかった。

われわれはデンドログラムがどのように成長するかを調べるため、合併履歴を観察したところ、少数の巨大クラスタが、多数の小さなクラスタと合併することにより急速に成長していることが観測でき、この現象のために不均衡なデンドログラムが作られていることが分かった。

図 2 は G^{500K}_{mixi} の解析プロセスにおける各合併ごとの**合併比率**を、片対数グラフにプロットした結果である。合併比率の定義を以下に示す。

$$ratio(c_i, c_j) = \min(|c_i|/|c_j|, |c_j|/|c_i|).$$

図 2 は n 回目の合併を、$c_k := \mathrm{join}(c_i, c_j)$ とすると、$(n, ratio(c_i, c_j))$ にプロットしている。$|c|$ はクラスタの

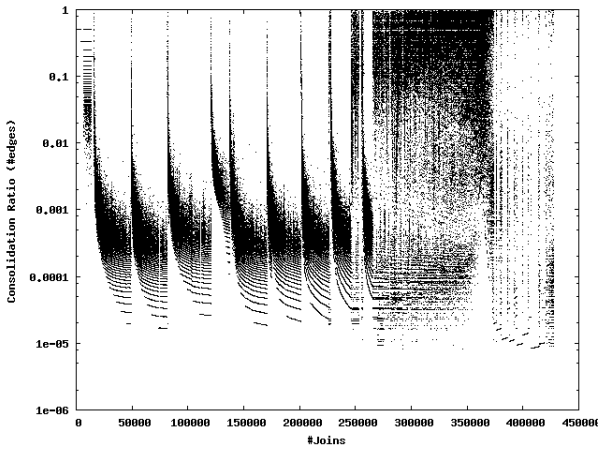

図 2　CNM アルゴリズムにおけるクラスターの合併比率の推移 (片対数)。

サイズである。この図より、解析の前半で、8 つの巨大なクラスタの成長を観測できた。われわれはこの偏って成長したクラスタが CNM アルゴリズムを mixi ネットワークに適用したときの主な計算速度劣化の原因と考えた。

不均衡な合併はデンドログラムの高さを成長させ、$d \approx \log n$ ではなく、$d \approx n$ となり $O(n^2 \log n)$ となる。

4　ヒューリスティック

前節では CNM アルゴリズムの問題点を指摘した。本節では CNM アルゴリズムの問題点を改善し、速度を劇的に向上させる 3 つのヒューリスティックを提案する。

アルゴリズム 2 提案アルゴリズムの概略

$ratio(c_i, c_j) \equiv \min(s(c_i)/s(c_j), s(c_j)/s(c_i))$;
while (**true**) {
　updateDeltaQ();
　$\Delta Q^{\mathcal{C}}_{c_i,c_j} \cdot ratio(c_i, c_j)$ が最大となる $(c_i, c_j) \in \mathcal{C}^2$.
　if $(\max(\Delta Q^{\mathcal{C}}_{c_i,c_j} < 0)$ **break**;
　$\mathcal{C} := \mathrm{join}(c_i, c_j)$;
}

提案するアルゴリズムをアルゴリズム 2 に掲げる。全体の構造は CNM アルゴリズムとほぼ同じである。異なるのは合併するクラスタの対を決める戦略である。CNM アルゴリズムでは $\Delta Q^{C}_{c_i,c_j}$ を用いていた。本アルゴリズムではそれに加えて合併比率 $(ratio(c_i, c_j))$ を用いる。このヒューリスティックは、合併の均衡を保ち、平均したクラスタの成長を促すことにより、合併の不均衡によるアルゴリズムの性能の劣化を抑えるように設計した。

さて、本稿ではここまでクラスタサイズ $(|c_i|)$ を定義してこなかった。ここで、3 種類のクラスタサイズの定義を行うことにより、3 種類のヒューリスティック HE, HN, HE' を定義する。以下にそれぞれのヒューリスティックのクラスタサイズ $|c_i|$ の定義を示す。

HE　：c_i の隣接クラスタ数

HN　：c_i 内のノード数

表 1　実行時間 (秒)

	200K	400K	600K	800K	1M
CNM	2,530	11,800	NA	NA	NA
HE	129	408	814	1470	2170
HE'	511	2,130	4,090	7,410	10,400
HN	25.7	70.0	123	190	268

表 2　モジュール性の比較

	100K	400K	700K	1M
CNM	0.496	0.478	NA	NA
HE	0.413	0.373	0.355	0.339
HE'	0.535	0.529	0.520	0.514
HN	0.421	0.382	0.358	0.351

HE'　：HE と CNM アルゴリズムを混同させた定義[2]

5　評価

本節では、CNM アルゴリズムと前節で紹介した 3 種類のヒューリスティック (HE、HE'、HN) を用いたアルゴリズムの評価を行う。実験環境として、Intel Xeon 2.80GHz、メモリ 4GB を用い、Linux(Red Hat Linux version 2.6.16) 上で実験した。4 種類のアルゴリズムの実装言語には Java(Java 5.0, Java HotSpot Server VM build 1.5.0_06 b-05) を用いた。また、Xeon は CPU を複数内蔵しているが、単一スレッドのみの実装を行い、並列化は行わなかった。

4 種類のアルゴリズムを用い $G^n_{mixi}, (n \in \{50K, 100K, \ldots, 1000K\})$ を解析・比較した。解析時間を表 1、図 3 に示す。CNM アルゴリズムが解析できた最大のデータセットは G^{500K}_{mixi} であり 5.9 時間の解析時間を要した。最も解析時間が短かったアルゴリズムは HN で、G^{1000K}_{mixi} に対して 5 分以内で解析することができた。HE と HE' は G^{1000K}_{mixi} に対してそれぞれ、36 分と 3 時間の解析時間だったが、HN よりも低速であるものの、データのサイズを考慮すれば十分に実用的である。

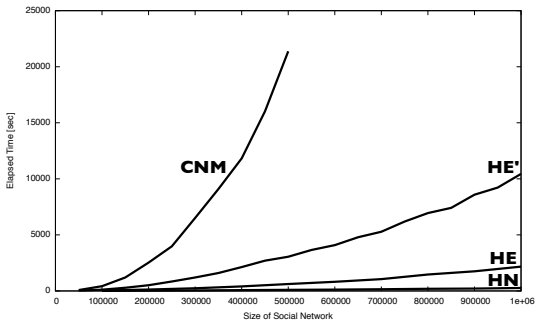

図 3　実行時間比較

次にモジュール性の評価だが、モジュール性の向上を最優先とする CNM アルゴリズムと比較し、われわれの手法では合併比率をも考慮したために、モジュール性が

[2]詳細は [12] をご覧ください。

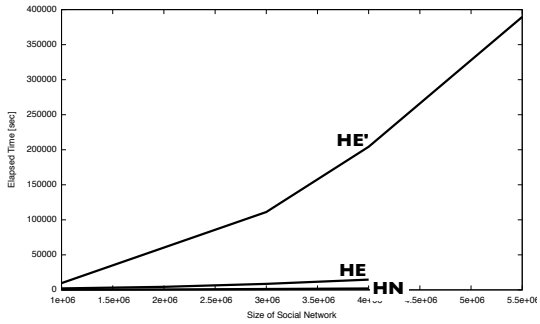

図 4　スケーラビリティ

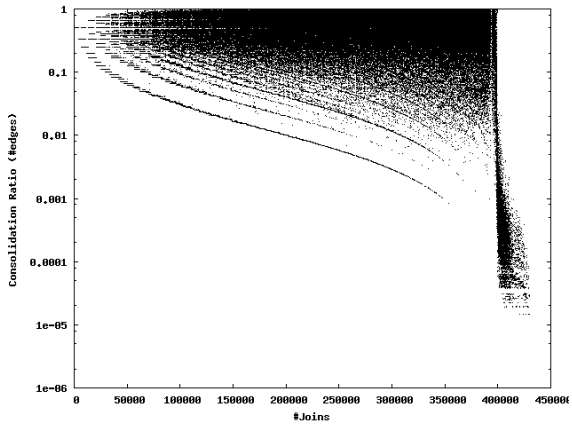

図 5　HN におけるクラスターの合併比率の推移 (片対数)。

悪化することを予想していた。しかし、表 2 に見るように、HE' では CNM アルゴリズムよりも若干であるがモジュール性が上回った。この結果より、HE' アルゴリズムでは G_{mixi}^{500K} において、CNM アルゴリズムの 7 倍の速度性能を持ち、モジュール性においても 8 – 11% の向上が見られ、速度面・モジュール性の両面で CNM アルゴリズムを上回る事ができた。

HN アルゴリズムは、速度面において HE よりも若干の性能を有すが、モジュール性においては CNM アルゴリズムの 21% – 28% 劣る結果となった。

図 5 に HN の G_{mixi}^{500K} の解析プロセスにおける各合併ごとの合併比率を示す。この図からわかるように、CNM アルゴリズムと比べ劇的に合併比率の偏りを抑えることができた。

われわれはさらに 2006 年 10 月に取得した約 550 万ユーザーからなる mixi ネットワークに提案するヒューリスティックを適用した。図 4 はそのときの実行時間である。この図より、HE と HN はほぼ線形で実行時間が増加していることがわかる。一方 HE' は、他の 2 つよりも遅いが、それでもなお 550 万ノードに対しても実用的な時間での解析が可能であった。われわれのアルゴリズムのスケーラビリティは、物理メモリサイズにより制約されており、HE と HN は物理メモリ不足のため 550 万ノードが実行できなかった。

6　まとめ

本研究では従来の Clauset, Newman, Moore が提案したアルゴリズムのボトルネックが合併の不均衡にあることを明らかにした。そして、3 種類のヒューリスティックを提案し、ボトルネックを取り除くことによって 500 万ノード以上のネットワークの解析も可能にした。

今後の課題としては、データ構造の改善によりより大規模なネットワークに対する解析が可能になると考える。また、並列化を施すことにより、数億ノード規模のネットワークに対する解析にも対応できると考えられる。

謝辞　本研究の一部は文部科学省科学研究費助成金（18300041 号）の援助を受けています。本稿の草稿について貴重なコメントをいただいた越田港さんに感謝します。

参考文献

[1] D. Cai, X. He, J. R. Wen, and W. Y. Ma. Block-level link analysis. Technical Report MSR-TR-2004-50, Microsoft Research, 2004.

[2] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, Vol. 70, p. 066111, 2004.

[3] J. Dean and M. R. Henzinger. Finding related pages in the world wide web. In *Proc. the 8th intern. conf. World Wide Web*, pp. 1467–1479, 1999.

[4] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *Proc. the 9th conf. Hypertext and hypermedia: links, objects, time and space—structure in hypermedia systems*, pp. 225–234, 1998.

[5] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. the 9th symp. on Discrete algorithms*, pp. 668–677, 1998.

[6] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *Proc. the 8th inter. conf. on World Wide Web*, pp. 1481–1493, 1999.

[7] J. C. Miller, G. Rae, and F. Schaefer. Modifications of kleinberg's hits algorithm using matrix exponentiation and web log records. In *Proc. the 24th intern. conf. Research and development in information retrieval*, pp. 444–445, 2001.

[8] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, Vol. 69, p. 026113, 2004.

[9] M. Onsjo and O. Watanabe. Simple algorithms for graph partition problems. Technical Report C-212, Department of. Mathematical & Computing Science, Tokyo Institute of Technology, 2005.

[10] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[11] M. Toyoda and M. Kitsuregawa. Creating a web community chart for navigating related communities. In *Proc. the 12th conf. Hypertext and Hypermedia*, pp. 103–112, 2001.

[12] Ken Wakita and Toshiyuki Tsurumi. Finding community structure in mega-scale social networks. In *Proc. the 16th intern. conf. World Wide Web*, p. 1275, 2007.

[13] F. Wu and B. A. Huberman. Finding communities in linear time: A physics approach, 2003.