大規模グラフに対するクラスタリングの高速化に関する研究

Efficient Clustering Algorithms for Large-scale Graphs

塩川 浩昭 ♥

Hiroaki SHIOKAWA

グラフクラスタリングはグラフの中に存在するコミュニティ構造を理解する上で重要な要素技術である．しかしながら，既存のグラフクラスタリング手法の多くは，グラフに含まれる全てのエッジに対して計算を行うため，大規模なグラフに適用した際に膨大な計算時間を要する．本研究ではこの問題を解決するために，3 つの高速化アルゴリズムを提案する．本稿で提案するアルゴリズムでは，実世界のグラフが持つ統計的な性質を捉えることでクラスタリングに要する計算時間を削減する．本稿では実データを用いた評価実験を行い，3 つの提案手法が従来手法と比較して高速かつ，分析精度を劣化させない事を確認した．

Graph clustering is one of the most important techniques in various research areas such as data mining, social science, marketing and so on. The problem of the graph clustering is to find clusters which are densely connected within the cluster and sparsely connected inter clusters, and this problem has been studies for some decades in many fields. Since, the size of graphs are becoming significantly larger due to the big data era, efficient clustering algorithms that can handle large-scale graphs are highly demanded. In order to achieve efficient clustering for large-scale graphs, we investigate three novel algorithms by handling statistical properties of real-world graphs. In the experiments over real-world datasets, we verified that the algorithms achieve not only efficient clustering but also high accurate clustering results for the graphs.

1. はじめに

グラフ構造はデータをノードとエッジで表現した基本的なデータ構造であり，情報推薦や情報検索，科学データ分析などの様々な分野で利用されている．特に近年では，数億ノードから構成される大規模なグラフ構造が登場し，このようなデータに対する高速な解析処理技術への需要が高まっている．例えば，Facebook では2012 年に1ヶ月当たりのアクティブユーザ数が10 億人，またTwitter では一日当たりの投稿数が3 億4000 万を突破したと報告されている[1]．このように，大規模グラフは現実に存在し今後も規模をさらに増大させていくことが考えられ，大規模なグラフ構造に対する高速な解析手法は必要不可欠な技術となってくると言える．

♥ 正会員　日本電信電話株式会社　ソフトウェアイノベーションセンタ　shiokawa.hiroaki@lab.ntt.co.jp

グラフ構造解析のひとつとして，クラスタリングが挙げられる．グラフ構造にはクラスタと呼ばれる相互に密な接続を有する部分ノード集合が存在する．例えば，Web グラフではトピックの近いページ集合が互いにリンクすることで，トピックの類似したページ群がコミュニティを形成する傾向にある．グラフ構造中のクラスタは互いに共通した性質を持つことから，グラフ構造の理解や様々なアプリケーションに利用され，重要な要素技術となっている．

この背景からこれまで，最小カットに基づく手法[2]やクリーク検知に基づく手法[3]など数多くのクラスタリング手法が提案されてきた．しかしながら，既存のクラスタリング手法の多くは全てのノードとエッジに対して計算を行う必要があり，大規模なグラフからクラスタを抽出するために膨大な計算時間が必要となるという課題がある．

そこで本稿では，大規模なグラフに対する高速なクラスタリングを実現する手法を提案する．本稿では現実世界のグラフが持つ高いクラスタ性[4]や次数のべき乗則[4]などの特性をクラスタリングに利用することで計算対象となるノードとエッジを削減していく．本稿では特に幅広く利用されているモジュラリティクラスタリングと構造的類似度によるクラスタリングを対象として，以下の 3 手法を提案する．

1. **モジュラリティクラスタリングの高速化 ：**
   ノードの逐次集約方式を導入した階層凝集型のモジュラリティクラスタリング手法を提案する．本手法は実世界のグラフが持つ高いクラスタ性と次数のべき乗則を捉え，処理を高速化する．
2. **モジュラリティクラスタリングのデータ並列化：**
   複数のデータに対する演算を CPU の 1 サイクルで同時実行する CPU の拡張命令である SIMD 命令を活用し，モジュラリティクラスタリングのデータ並列化手法を提案する．
3. **構造的類似度によるクラスタリングの高速化：**
   構造的類似度に基づくクラスタリング手法に対して directly two-hop away reachable(DTAR)という概念を導入し，クラスタ性の高い部分グラフへの計算を削減する手法を提案する．

本稿では次節以降で上記 3 つのアルゴリズムについて概説し，その性能評価について簡潔に紹介する．

2. モジュラリティクラスタリングの高速化

モジュラリティ[5]とは，Newman と Girvan により提案されたクラスタリング結果の質を評価する指標である．モジュラリティはクラスタ内に含まれたノード間のエッジが密であり，クラスタ間に存在するエッジが疎となる程良い値を示すように設計されている．具体的には以下の定義で与えられる．

[定義 1]（モジュラリティQ）：e_{uv}をクラスタu, v間に接続するエッジの重みの総和，a_uをクラスタuに接続するエッジの重みの総和，mをグラフに含まれる全てのエッジの重みの総和としたとき，モジュラリティは以下のように定義される．

$$Q = \sum_u \left\{ \frac{e_{uu}}{2m} - \left(\frac{a_u}{2m} \right)^2 \right\}$$

定義 1 で与えられるQを最大化するノードの組合せの発見は NP-hard であるため，既存のモジュラリティクラスタリングでは貪欲法に基づき探索的にQを最大化するノードの組合せを決定する．例えば state-of-the-art な手法のひとつで

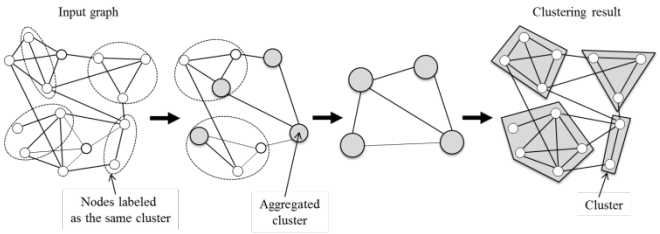

図 1. ノード逐次集約法

表 1. データセット

	dblp	ljve	uk-2005	webbase	uk-2007
ノード数	326K	5.36M	39.4M	118M	105M
エッジ数	1,61M	79.0M	936M	1.01B	3.73B

ある BGLL[6]は，ランダムでノードuを選択し，定義 2 に示すモジュラリティゲインを各エッジに対して計算する．

[定義 2]（モジュラリティゲインΔQ_{uv}）: ΔQ_{uv}はクラスタu, vを同一クラスタにした際の，モジュラリティの増加量とするとき，$\Delta Q_{uv} = 2\left\{\frac{e_{uv}}{2m} - \left(\frac{a_u}{2m}\right)\left(\frac{a_v}{2m}\right)\right\}$と定義される．

その後，ノードuの隣接ノード集合からモジュラリティゲインを最大化するノードvを選択し，ノードu, vを同一のクラスタにラベル付けする．この処理をモジュラリティが向上する限り繰り返す．この貪欲法に基づくモジュラリティクラスタリングは高いモジュラリティを示すクラスタを生成することが実験的に報告されている[6]が，依然として全てのノードとエッジを繰り返し計算する必要があり，大規模なグラフにおいて計算時間が膨大になるとう課題が存在する．

2.1 提案手法概要

提案手法では前述の課題を解決するために，実世界のグラフが持つ高いクラスタ性とべき乗則に従う次数分布に着目する．提案手法は次に示す 3 つのアプローチから構成される：(1) 同一クラスタにラベル付されたノードペアの逐次的な集約による計算対象ノード/エッジの削減，(2) 所属クラスタが自明となるノードの検出による計算対象ノード/エッジの削減，(3) ノードの選択順序最適化による計算対象エッジの削減．各アプローチの詳細については文献[7]を参照されたい．既存手法では全てのノードとエッジに対して繰り返しモジュラリティゲインの計算を行っていた．これに対し，提案手法では図 1 に示す様にノードの逐次集約を行うことで計算対象ノード/エッジの数を次第に減らし，クラスタを決定づける要因となるノードとエッジについてのみ計算するように動作する．また，ノードの逐次集約とノード選択順序の最適化により，高いクラスタ性を示すグラフにおいて極めて効果的に計算コストを削減することができる．

2.2 評価実験

評価実験の概要について述べる．本実験では表 1 に示す 5 つのデータセットを用いる．本実験は Linux 2.6.18 server, Intel Xeon CPU L5640 2.27GHz，144GB RAM を搭載する計算機上で行った．また，本実験では提案手法を C++で実装し，既存手法である CNM[8]および BGLL[6]と比較を行った．

図 2 に実行速度の評価結果を示す．本実験では提案手法を 2 つのタイプに分けて評価を行った．ひとつは，2.1 節で示した 3 つのアプローチ全てを用いた Proposed，もう一方はアプローチのうち(1)の逐次集約のみを用いた Proposed-Limited で

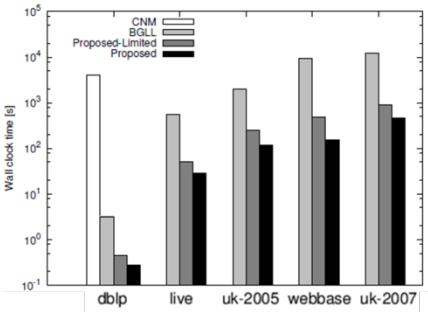

図 2. 実行速度の比較

表 2. モジュラリティ値の比較

	dblp	ljve	uk-2005	webbase	uk-2007
Proposed	**0.90**	**0.74**	**0.98**	**0.98**	**0.97**
BGLL	0.88	**0.74**	0.97	0.96	**0.97**
CNM	0.82	N/A	N/A	N/A	N/A

ある．CNM は dblp を除くデータセットにおいて 24 時間以上の計算時間を要したため図 2 から除外した．図 2 に示した様に提案手法 Proposed は CNM，BGLL と比較して高速に動作していることが確認できる．特に 1 億ノードを超える規模のグラフである webbase では state-of-the-art な手法である BGLL に対して約 60 倍高速であることが確認できる．また，図 2 において Proposed と Proposed-Limited を比較すると，前者は後者に対して約 3 倍程度高速であることが分かる．これはアプローチの(1)に示したノードの逐次集約が高速化に対して最も大きな貢献を果たしていることを示唆している．

次にクラスタリング精度の評価を行う．本評価では，クラスタリング結果の精度を示す指標として定義 1 のモジュラリティを用いた．表 2 に各手法のモジュラリティの値を示す．表 2 に示したように，提案手法は BGLL とほぼ同程度のモジュラリティ値を示すことが確認できる．モジュラリティという指標の性質上，クラスタリング結果が BGLL と一致することを保証するものではないことに注意されたい．

3. モジュラリティクラスタリングのデータ並列化

前節で述べたとおり，BGLL に代表される既存のモジュラリティクラスタリング手法は全てのノードとエッジに対してモジュラリティゲインを繰り返し計算する必要がある．そこで本説では，モジュラリティゲインの計算をデータ並列化することでクラスタを高速に求める手法を提案する．

3.1 提案手法概要

提案手法の概要を以下に述べる．提案手法ではまず，前節で述べたモジュラリティゲインの計算処理を定義 3 に示すような相対モジュラリティゲインに変換する．

[定義 3]（相対モジュラリティゲイン$\Delta Q'_{uv}$）: クラスタu, v間における相対モジュラリティゲイン$\Delta Q'_{uv}$を以下に定義する．

$$\Delta Q'_{uv} = 2\mathrm{me}_{uv} - a_u a_v$$

本手法は定義 2 の代わりに定義 3 を用いることで，モジュラリティゲインの計算に要する CPU 命令数を削減する．また，定義 2 で与えられるモジュラリティゲインの大小関係は，定義 3 に変換した際も保持されることが証明されている[9]．次に，データ並列化の効果を向上させるために，グラフのデ

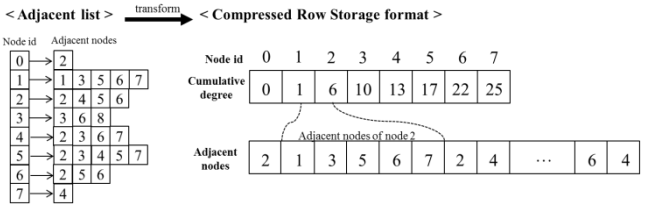

図 3. CRS format の例

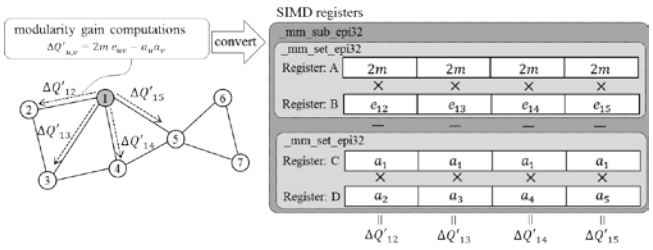

図 4. SIMD 命令を用いた相対モジュラリティゲイン計算

ータフォーマットを変換する．具体的には，提案手法では，通常用いられる隣接リスト表現の代わりに Compressed Row Storage (CRS) [10]と呼ばれるフォーマットを導入する．CRS とは疎行列を少ないメモリ領域に格納するためのデータフォーマットであり，疎行列のうち，被零要素を図 3 の様にメモリ空間上の連続した領域に配置する．提案手法ではこのフォーマットを用い，連続領域をシーケンシャルに走査する様に BGLL を置き換えることにより, CPU プリフェッチの効率を高めデータの読み込みにかかるコストの削減と後段のデータ並列化の効率の向上を図る．最後に，Streaming SIMD Extensions（SSE）を用いて相対モジュラリティゲインの計算をデータ並列化する．SSE とは x86 CPU の拡張命令である．SSE は同一の演算を SIMD レジスタ上に存在する複数のデータに対して同時に実行する．本手法では，SSE を相対モジュラリティゲイン計算に用いることで，複数のノードペアに対する計算を 1 回の CPU 命令の実行で求める．具体的には，あるノードの隣接ノード集合に対して，相対モジュラリティゲインを計算する際に，図 4 の様に SIMD レジスタ上に該当するデータを展開する．その後図 4 に示した 3 種類の SIMD 命令を用いて相対モジュラリティゲインを計算する．より詳細な方式については文献[9]を参照されたい．

3.2 評価実験

提案手法の評価実験について述べる．本実験で用いたデータセットおよび実験環境は 2.2 節と同様である．実験に用いた SIMD レジスタサイズは 128bit, 計算用のデータ型は 32bit であり，最大で 4 並列の計算が実行できる環境を構築した．

図 5 に実行速度の評価結果を示す．本実験では提案手法を 3 つのタイプに分けて評価を行った．図 5 の Cache+SIMD は BGLL に対して CRS を用いたデータ読み込みコストの削減と SIMD によるモジュラリティゲイン計算のデータ並列化を行った手法である．これに対し, Cache only および SIMD only は BGLL に対してそれぞれ CRS および SIMD のみを導入した手法である．図 5 に示したように Cache+SIMD, Cache only, および SIMD only は BGLL と比較してより高速に動作していることが確認できる．具体的には，Cache+SIMD，Cache only および SIMD only は BGLL に対してそれぞれ約 20 倍および約 5 倍，1.9 倍の高速化を実現している．

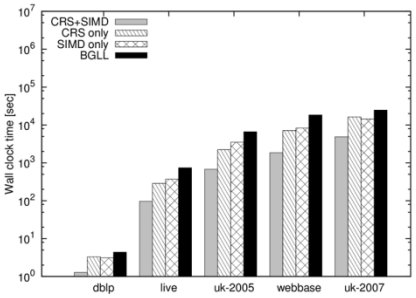

図 5. 実行速度の比較

4. 構造的類似度によるクラスタリングの高速化

モジュラリティという指標はグラフの規模が大きくなるとともにクラスタの精度が低下するという課題が指摘されている[11]．この課題を解決する手法として近年利用が広まっている手法のひとつが Xu らにより提案された構造的類似度によるグラフクラスタリング SCAN[12]である．SCAN は以下に定義される構造的類似度(structural similarity)を全てのエッジに対して計算する事で，ノード間の接続が密な部分グラフをクラスタとして抽出する．

[定義 4]（構造的類似度σ(u,v)）：ノードu,v間の構造的類似度は$\sigma(u,v)=\frac{|\Gamma(u)\cap\Gamma(v)|}{\sqrt{|\Gamma(u)||\Gamma(v)|}}$で定義される．$\Gamma(u)=\{v\in\mathbb{V}:(u,v)\in\mathbb{E}\}\cup\{u\}$，$\mathbb{V}$はノード集合，$\mathbb{E}$はエッジ集合である．

ノード間の接続が密なところのみをクラスタとして抽出することで，モジュラリティの持つ Resolution limit を解決している．また，SCAN の特徴のひとつとして，クラスタのみならず，クラスタ間に接続するハブと呼ばれる特殊なノードや通常ノイズとして扱われるような外れ値と呼ばれるノードもクラスタと併せて抽出することが可能である．

SCAN はモジュラリティに基づく手法と比較して, クラスタ精度が高くなることが報告されている[12]．しかし構造的類似度を全てのエッジに対して計算する必要があり，クラスタリングには膨大な計算時間を要するため，大規模なグラフに対するクラスタリングが難しいという課題を有している．

4.1 提案手法概要

本研究の目標は SCAN に対してクラスタの精度を劣化させずにクラスタリングを高速化することである．提案手法の概要を以下に示す．前節で述べたとおり，SCAN は全てのエッジに対して構造的類似度を計算するため，計算時間が膨大になっている．そこで提案手法では，実世界のグラフの持つ高いクラスタ性に着目する．クラスタ性の高いグラフでは，互いに 2 ホップ離れるノードu,vが存在した時に，それらの隣接ノード集合の積集合Γ(u) ∩ Γ(v)に含まれるノード数が多くなると考えられる．そこで提案手法では，この性質を利用した 2phase clustering 方式を提案する．

本方式は，図 6 に示した様に(1)local clustering phase と(2) cluster refinement phase から構成される．(1) local clustering phase では互いに 2 ホップ離れたノードu,vについてのみを探索し，構造的類似度を計算する．本手法では，2 ホップ離れたノードを探索する際に directly two-hop away reachable(DTAR)という新たな概念を導入している．その後，2 ホップ離れたノードが隣接ノードとのみ構築する local cluster を抽出する．この際に，複数の local cluster を接続するノードを bridge と呼び，保存しておく．続いて(2)

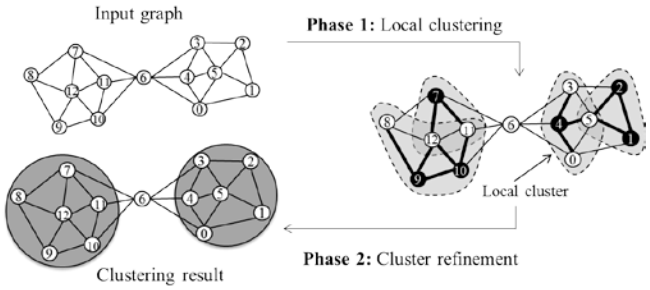

図 6. 2 phase clustering 概要

表3. データセット

	road	cnr	google	skitter
ノード数	137M	324K	875K	1.69M
エッジ数	3.84M	5.47M	5.10M	11.0M

cluster refinement phase では，前 phase で獲得した local cluster と bridge から最終的に出力するクラスタを構築する．具体的には, bridge を順に走査していき, 複数の local cluster に隣接する bridge が存在した場合には，隣接する local cluster を同一のクラスタに統合する．提案手法では，上記の 2 phase clustering に加え，構造的類似度の計算量自体を削減する類似度共有方式も提案しているが，説明は割愛する．

4.2 評価実験

評価実験の概要について述べる．本実験では表3に示す5つのデータセットを用いる．本実験はLinux 2.6.18 server，Intel Xeon CPU L5640 2.27GHz，144GB RAMを搭載する計算機上で行った．また，本実験では提案手法をC++で実装し，エッジサンプリングを用いたナイーブな手法SCAN*および既存手法であるSCAN[12]，gSkeletonClu[13]と比較を行った．

図7に実行速度の比較を示す. gSkeletonClu に関しては, skitter においてクラスタリングに 24 時間以上の計算を要したため図 7 からは除外している．また，本実験では類似度のしきい値となるパラメータϵを 0.2，0.4，0.6，0.8 と変化させて評価を行ったが，提案手法を除き実行速度に大きな変化が確認できなかったため図 7 からは除外した．

図 7 に示した様に提案手法は, いずれのデータセットにおいても高速に動作していることが確認できる．具体的には，提案手法はベースラインである SCAN と比較して平均で 20 倍程度高速にクラスタリングを実行している．

5. まとめと今後の課題

本稿では大規模グラフを対象としたグラフクラスタリングの高速化に取り組んだ．本稿で提案する 3 つのクラスタリング手法は実世界のグラフが持つクラスタ性や次数のべき乗則などの統計的な特性を捉え，クラスタの精度に影響を与えないノードやエッジを計算対象から除外する．これにより，本稿で提案した各手法は，既存手法と比較して精度を同程度に保ちつつ，高速な処理を実現している．

今後の研究課題として以下の 3 つが挙げられる. 一つ目は，時間経過とともに構造を変化させるグラフ(time-evolving graph)への対応である．二つ目は，クラスタリング手法のパラメタフリー化である．そして，三つ目 は，グラフクラスタリングの分散並列化である．

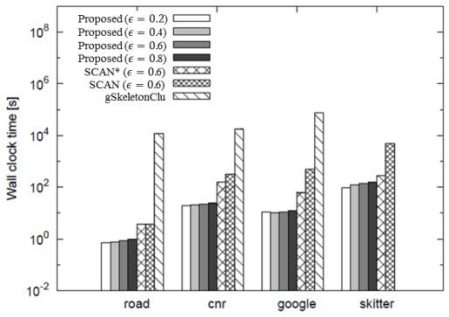

図 7. 実行速度の比較

[文献]

[1] C. Sibona and J. H. Choi: “Factors Affecting End-User Satisfaction on Facebook”, In *Proc. ICWSM,* 2012.
[2] J. Shi and J. Malik: “Normalized Cuts and Image Segmentation”, *IEEE TPAMI, 22(8),* pp888-905, 2000.
[3] J. Wang and J. Cheng: “Truss Decomposition in Massive Networks”, *PVLDB, 5(9),* pp812-823, 2012.
[4] D. J. Watts: “Small Worlds: The Dynamics of Networks Between Order and Randomness”, *Princeton Studies in Complexity*, Princeton University Press, 1999.
[5] M. E. J. Newman and M. Girvan: “Finding and Evaluating Community Structure in Networks”, *Phys. Rev. E, 69:026113* Feb., 2004.
[6] V. D. Blondel, J. L. Guillaume, R. Lambiotte and E. Lefebre: “Fast Unfolding of Communities in Large Networks”, *Journal of Statistical Mechanics: Theory and Experiment, P10008*, 2008.
[7] H. Shiokawa, Y. Fujiwara, and M. Onizuka: “Fast Algorithm for Modularity-based Graph Clustering”, In *Proc. AAAI, pp.1170-1176,* 2013.
[8] A. Clauset, M. E. J. Newman, and C. Moore: “Finding Community Structure in Very Large Networks”, *Phys. Rev. E, 70:066111*, Dec. 2004.
[9] H. Shiokawa, T. Yamamuro, Y. Fujiwara, and M. Onizuka: “Parallel Approach for Modularity-based Graph Clustering with SIMD Instruction”, *DBSJ Journal, Vol.12, No.1*, pp.91-96, 2013.
[10] J. Demmel, J. Dongarra, A. Ruhe, and H. Vorst: “Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide”, SIAM, 2000.
[11] S. Fortunato and M. Barthélemy: “Resolution Limit in Community Detection”, In *Proc. the National Academy of Sciences, 104(1)*, pp.36-41, 2007.
[12] X. Xu, N. Yuruk, Z. Geng, and T. A. J. Schweiger: “A Structural Clustering Algorithm for Networks”, In *Proc. KDD*, pp. 824-833, 2007.
[13] J. Huang, H. Sun, Q. Song, H. Deng, and J. Han: “Revealing Density-Based Clustering Structure from the Core-Connected Tree of a Network”, *IEEE TKDE, 25(8)*, pp.1876-1889, 2013.

塩川 浩昭 Hiroaki SHIOKAWA
2009 年筑波大学第三学群情報学類卒業．2011 年同大学院システム情報工学研究科博士前期課程修了．同年，日本電信電話株式会社入社．2015 年筑波大学大学院システム情報工学研究科博士後期課程修了．博士（工学）．大規模データ分析の研究開発に従事．日本データベース学会，ACM 各会員．