# コミュニティ構造を考慮した属性付きグラフ汎用生成機構

前川　政司† George H. L. Fletcher†† 鬼塚　真†

† 大阪大学大学院情報科学研究科　〒 565–0871 大阪府吹田市山田丘 1–5
†† Eindhoven University of Technology, 5612 AZ Eindhoven, Netherlands
E-mail: †{maekawa.seiji,onizuka}@ist.osaka-u.ac.jp, ††g.h.l.fletcher@tue.nl

**あらまし**　多くの実世界のグラフでは, ノードは属性を持っている. そのため, トポロジーと属性の両方を利用して, グラフからコミュニティを推定する属性付きグラフクラスタリングの需要が増してきている. それらの手法を評価するために正解データ付きのグラフデータが必要であるが, 多種多様な属性付きグラフデータを十分に収集することは困難である. 本稿では, コミュニティ構造を考慮したスケーラブルな属性付きグラフ汎用生成機構を提案する. 本機構では, ノード次数, コミュニティサイズ, 属性の値に対して, 様々な分布を仮定した, 汎用なグラフ生成が可能である. スケーラビリティについて実験を行い, エッジ数および属性数に関して線形時間でグラフが生成できることを示した. 生成したグラフに対して, 既存のクラスタリング手法を適用し, アルゴリズムを定量的に評価できることを示した.

**キーワード**　属性付きグラフ, ベンチマーク, コミュニティ

## 1 はじめに

グラフはノードとそれらの関係を示すための基本的なデータ構造である. 多くの領域でグラフデータは現れており, 例えばソーシャルネットワーク [5], タンパク質の結合 [4], 交通計画 [7], コンピュータビジョン [9], そして遺伝子表現 [13] などが挙げられる. グラフ処理は研究者たちの注目を集めており, 特にグラフクラスタリングは機械学習およびデータマイニングで最も広く用いられている技術の 1 つである.

実世界のグラフデータでは多くの場合, ノードが属性を持っている. 実際に, グラフデータベースは属性付きグラフにも対応している [6], [23]. そして, 属性付きグラフのためのクラスタリング手法が注目されてきている [8], [24], [1], [21].

研究者たちは自身の設計したクラスタリング手法を評価するために, 大規模かつ多様なデータセットを必要としている. 一方で属性付きグラフについては, 正解クラスタが既知でかつ, 大規模なデータの収集は非常に困難である. 実際に, グラフデータのアーカイブとして広く知られている SNAP [16] において, ノードが属性を持ち, かつコミュニティ構造の正解データが与えられているデータは公開されていない. また, 現在公開されている正解データが付属した属性付きグラフ [1] の大半は小規模なものである. そのため, グラフのサイズや特性を自由に設定できるグラフ生成機構の必要性が高い. しかし, 属性付きグラフに対応した既存のグラフ生成機構 [2], [15] は, ノード次数やクラスタサイズ, 属性の値に対して, 様々な分布を指定することができない. 例えば [15] では, クラスタサイズに関して確率分布を指定することができず, また属性の値に関しては正規分布のみを仮定している.

本稿では, コミュニティ構造を考慮した属性付きグラフ汎用生成機構である acMark を提案する. acMark では, ユーザが指定する任意の分布に従ってノード次数を決定し, そのノード次数に基づいてエッジを生成する. そのため, ノード次数分布を制御することが可能である. また, ユーザが指定した分布をディリクレ分布の入力ベクトルとして利用するため, コミュニティサイズに関してもユーザが分布を選択することができる. 属性の値については, 同じクラスタ内のノードは似た属性値を持ち, かつユーザ指定可能な分布に従うように生成される. 属性ごとに分布を選択できるため, 多様な属性を持つグラフを生成可能である. 計算量に関して, 生成するグラフのエッジ数に対して線形時間で処理が終了するため, acMark は大規模グラフの生成が可能である. そのため, 大規模でかつ, 様々な特性を持つ属性付きグラフを生成することができる.

実験では, まず実際に生成したグラフのプロパティが設定した分布に従っているかを確認した. スケーラビリティの検証のために, 様々なエッジ数, 属性数で実験を行い, いずれのパラメータに対しても線形時間でグラフを生成できることを示した. またパラメータを調整することによって, クラスタ分離度を制御することが可能であることも示した. 最後に生成したグラフに対して, 既存のクラスタリング手法を適用し, それぞれの手法を定量的に評価できることを示した.

本稿の構成は以下の通りである. 2 章で事前準備, 3 章

1：https://linqs.soe.ucsc.edu/

で関連研究について述べる．4 章では, 提案手法について述べ, 5 章で実験結果を示し, それについて考察する．6 章で結論を述べる．

## 2 事前準備

属性付きグラフは $G = (V, E, A)$ と表すことができ，ノード集合 $V = \{1, 2, \ldots, n\}$，エッジ集合 $E = \{(i, j)\} \subseteq [n] \times [n]$，およびノードを属性空間に射影する関数 $A = \{dom_{A_1}(v), dom_{A_2}(v), \ldots, dom_{A_d}(v)\}$ からなる．トポロジーに基づいたコミュニティ構造は $\boldsymbol{C}^T \in \{1, ..., k_1\}^n$ と表され, 属性に基づいたコミュニティ構造は $\boldsymbol{C}^A \in \{1, ..., k_2\}^n$ と表される．現在, 全ての既存のグラフ生成機構は $\boldsymbol{C}^T$ ついてのみ考慮している．

### 2.1 グラフプロパティ

次に生成したグラフが満たすべき, 実世界の属性付きグラフが持つプロパティについて述べる．属性付きグラフが満たすべきプロパティは, トポロジーについての構造プロパティと属性についての属性プロパティに分けることができる．

#### 2.1.1 構造プロパティ

実世界のグラフは広く知られたプロパティ [20] を有しており, 人工グラフもそれらのプロパティを持つべきである．例えば, ノード次数およびコミュニティサイズはそれぞれべき乗則に従うことが多いことが知られている．

#### 2.1.2 属性プロパティ

属性の値は大きく 2 つのタイプに分けることができる．1 つ目はカテゴリー値 (e.g. 会議, 大学, 街など) であり, 2 つ目は数値 (e.g. 価格, タイムスタンプなど) である．カテゴリー値に関しては, 1-hot vector によって, バイナリで表現することができる．実世界の数値をとる属性のうちの多くは, べき乗則や正規分布に従うことが知られている [18].

## 3 関連研究

本章では, 提案手法と関連の深い既存のグラフ生成機構を説明する．ここでは, グラフクラスタリング手法から考えられるグラフ生成手順についても言及する．各生成機構の特徴をまとめたものを表 1 に示す．

### 3.1 LFR-benchmark

Lancichinetti らによって提案された LFR-benchmark [14] は, 人工グラフデータの生成に広く用いられている生成機構である．この生成機構はグラフ構造にのみ注目しており, ノード属性の生成は行わない．LFR-benchmark の特徴はノード次数とコミュニティサイズの分布を考慮に入れることができる点である．実行時間については, 各ノードについてエッジを生成するペアを探すために, 生成するエッジ数だけの試行回数が必要になるため, $O(m)$ となる．ここでの $m$ はグラフが持つエッジ数を表す．

表 1: 各手法のグラフプロパティ．表中の PL はべき乗分布, ND は正規分布を意味している．提案手法 acMark は, ノード次数とコミュニティサイズに関して, 本質的には任意の分布をサポートすることができる．× は該当するプロパティについて仮定がなされていないことを意味する．$*$ が付いている手法は, クラスタリング手法の逆演算を想定した生成機構である．

| generator | node degree | community size | attribute |
|---|---|---|---|
| LFR | PL | PL | × |
| ANC | PL | × | $\mathbb{R}$(ND) |
| PAICAN* | PL | × | $\{0, 1\}$ |
| NAGC* | × | × | $\{0, 1\}$ |
| acMark | arbitrary | arbitrary | arbitrary |

### 3.2 ANC

Largeron らによって提案された ANC [15] は, コミュニティ構造とノード属性の両方を持つグラフを生成する．最初にノードの属性を正規分布に基づいて生成し, それらの属性値からサンプリングした集合に対して k-medoids [12] を適用し, クラスタコアを得る．そのため, 構造プロパティを考慮しないクラスタしか生成できない問題点がある．一方でノード次数に関しては, べき乗則に従うようにエッジが生成される．

### 3.3 PAICAN

Bojchevski らによって提案された PAICAN [3] は, 属性付きグラフクラスタリング手法であり, 2 つの側面がある．1 つ目はベイジアンアプローチによって属性付きグラフからクラスタを抽出することであり, 2 つ目は属性グラフから部分的な異常値も含めた異常値検出を行うことである [2]．PAICAN は生成モデルでもあり, 目的関数の式から隣接行列および属性行列を構成するための計算式が定義されている．そのため, PAICAN の逆演算を考えたとき, クラスタやノード次数などの必要な変数を用意すれば, 隣接行列および属性行列を目的関数の式の計算から得ることができる可能性がある [3]．グラフを生成するときに, 隣接行列の要素全てに対して, エッジの生成確率を計算する必要があるため, PAICAN の逆演算の時間計算量は $O(n^2)$ となる．

### 3.4 NAGC

Maekawa らによって提案された NAGC [17] は属性付きグラフのためのクラスタリング手法であり, グラフか

2：我々の取り組みでは, 異常値について考慮していないので, 異常値がない場合を考える．

3：実際には制約充足問題であり, 与えられた条件を可能な限り満たした近似解を得ることしかできない．

らノードごとのクラスタ割合を抽出する．クラスタ割合とは, ノードと各クラスタの関係の強さを表すものである．この手法は行列分解手法に非線形関数および中間層の概念を導入することにより, 既存手法では捉えられなかった構造と属性の間の複雑な関係を考慮したクラスタリングを可能にする．この手法では, 目的関数の式においてクラスタ割合と 隣接行列および属性行列の関係が明示的に定義されている．そのため, NAGC の逆演算を考えたとき, クラスタ割合などの必要な変数を用意し, 目的関数の式に基づいて, 隣接行列と属性行列を計算することが可能である．NAGC の逆演算では, 隣接行列の全ての要素に対してエッジ生成確率を計算する必要があるため, 時間計算量は $O(n^2)$ となる．

### 3.5 SBM

SBM(Stochastic Block Models) [22] はエッジの重みを考慮しないグラフを表すために設計されており, 隣接行列の各要素 $\boldsymbol{S}_{ij}$ はベルヌーイ分布に従うことを仮定している．Karrar らは SBM をマルチグラフに拡張した手法 [10] を提案した．マルチグラフとは, 同じノード対で複数のエッジの存在できるように, 通常のグラフを拡張したものである．ここでは, $\boldsymbol{S}_{ij}$ はそれぞれ独立で, ポアソン分布に従う．ポアソン分布は, 距離や量などの特定の間隔の中で起きるイベントの数を表すために用いられる分布である．スパース条件下では, ポアソンモデルは元のベルヌーイモデル [22] と類似しているが, ポアソン分布は統計的に解析がより容易である．

## 4 提案手法

私たちはコミュニティ構造を考慮した属性付きグラフ汎用生成機構, acMark を提案する．acMark は, グラフプロパティおよび属性プロパティに対して, 多様な分布 [4] を指定することを可能にするだけでなく, トポロジーと属性の間に, 非線形な関係を考慮することも可能である．表 2 に本章で用いる変数とその定義を示す．また提案するグラフ生成機構の入力を表 3 にまとめる．

提案手法では, まずユーザがグラフのサイズや属性数, クラスタ数および各グラフプロパティが従う分布とそのパラメータを入力として与える．それらの入力に基づいて, 構造と属性のクラスタ割合 $\boldsymbol{U}, \boldsymbol{V}$ およびクラスタ転移行列 $\boldsymbol{H}$ を生成する．構造のクラスタ割合 $\boldsymbol{U}$ はノードと構造クラスタの関係の強さを表す行列であり, 属性のクラスタ割合 $\boldsymbol{V}$ は属性クラスタと属性の関係の強さを表す行列である．転移行列 $\boldsymbol{H}$ とは, 構造クラスタ $\boldsymbol{U}$ と属性クラスタ $\boldsymbol{V}$ をつなぐための行列であり, 構造と属性がそれぞれ異なったクラスタを持つことを可能にする．その後それらの行列を用いて, 属性付きグラフを構成する隣接行列 $\boldsymbol{S}$ および属性行列 $\boldsymbol{X}$, そしてクラスタ割当 $\boldsymbol{C}$ を出力する．提案手法の概要図を図 1 に示す．

[4]：実際には, いくつかの分布 (べき乗分布, ユニフォーム分布, 正規分布) が実装されている．

表 2: 変数の定義.

| 変数 | 説明 |
|---|---|
| $\boldsymbol{S} \in \mathbb{R}_{+}^{n \times n}$ | 隣接行列 |
| $\boldsymbol{X} \in \mathbb{R}_{+}^{n \times d}$ | 属性行列 |
| $\boldsymbol{U} \in \mathbb{R}_{+}^{n \times k_1}$ | 構造のためのクラスタプロポーション |
| $\boldsymbol{V} \in \mathbb{R}_{+}^{d \times k_2}$ | 属性のためのクラスタプロポーション |
| $\boldsymbol{H} \in \mathbb{R}_{+}^{k_1 \times k_2}$ | クラスタ転移行列 |
| $\boldsymbol{\theta} \in \mathbb{R}_{+}^{n}$ | ノード次数リスト |
| $\boldsymbol{\chi} \in \mathbb{R}_{+}^{k_1}$ | コミュニティサイズリスト |
| $\boldsymbol{C} \in \mathbb{N}^{n}$ | クラスタ割当リスト |
| $\boldsymbol{M} \in \mathbb{N}^{*}$ | エッジ生成のための候補ノードリスト |

表 3: グラフ生成機構の入力とそれらの説明. $\phi, \delta, \sigma$ の添字の意味を以下に示す: $d$ はノード次数, $c$ はコミュニティサイズ, $V$ は属性のためのクラスタプロポーション, $H$ は構造と属性のクラスタ間の転移行列をそれぞれ示す.

| 入力 | 説明 |
|---|---|
| $n \in \mathbb{N}$ | ノード数 |
| $m \in \mathbb{N}$ | エッジ数 |
| $d \in \mathbb{N}$ | 属性数 |
| $k_1 \in \mathbb{N}$ | 構造のためのクラスタ数 |
| $k_2 \in \mathbb{N}$ | 属性のためのクラスタ数 |
| $\theta_{min}, \theta_{max} \in \mathbb{N}$ | ノード次数の最小値と最大値 |
| $\alpha \in \mathbb{R}$ | クラスタ間と内でのエッジ数の割合を調整するパラメータ |
| $\beta \in \mathbb{R}$ | 属性クラスタプロポーションのためのディリクレ分布のパラメータ |
| $\gamma \in \mathbb{R}$ | クラスタ転移行列のためのディリクレ分布のパラメータ |
| $\phi_d, \phi_c, \phi_V, \phi_H \in \mathbb{R}$ | べき乗分布で用いるパラメータ |
| $\delta_d, \delta_c, \delta_V, \delta_H \in \mathbb{R}$ | 一様分布の値域 |
| $\sigma_d, \sigma_c, \sigma_V, \sigma_H \in \mathbb{R}$ | 正規分布の分散 |
| $f_S$ | 隣接行列を構築するための関数 |
| $f_X$ | 属性行列を構築するための関数 |
| $r \in \mathbb{N}$ | エッジ生成ステップの反復回数 |
| $att_{ber} \in \mathbb{R}$ | 離散値をとる属性の割合 |
| $att_{pow} \in \mathbb{R}$ | べき乗分布に従う属性の割合 |
| $att_{uni} \in \mathbb{R}$ | 一様分布に従う属性の割合 |
| $att_{nor} \in \mathbb{R}$ | 正規分布に従う属性の割合 |
| $\phi_{att\_min}, \phi_{att\_max} \in \mathbb{R}$ | 属性値のためのべき乗分布のパラメータ |
| $\delta_{att} \in \mathbb{R}$ | 属性値のための一様分布の値域 |
| $\sigma_{att\_min}, \sigma_{att\_max} \in \mathbb{R}$ | 属性値のための正規分布の分散 |

### 4.1 acMark アルゴリズム

提案手法のアルゴリズムの全体像を Algorithm 1 に示し, Algorithm 2,3 において, エッジ生成ステップと属性

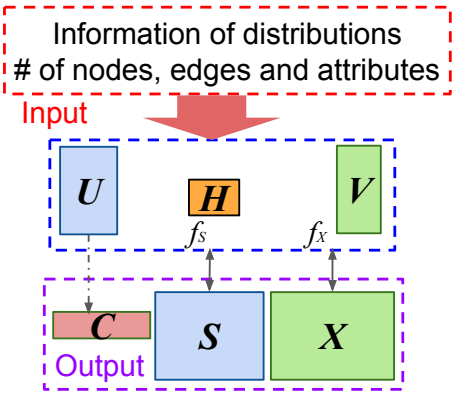



図 1: 提案法の概要図. $\boldsymbol{U}$ の行は各ノードのクラスタプロポーションを表し, $\boldsymbol{H}$ は構造と属性のクラスタを繋ぐ配分を表す. また $\boldsymbol{V}$ の行は各属性のクラスタプロポーションを表す. $\boldsymbol{C}$ はクラスタ割当, $\boldsymbol{S}$ は隣接行列, $\boldsymbol{X}$ は属性行列をそれぞれ表す. $f_S$ は $\boldsymbol{S}$ を生成するための関数を表し, $f_X$ は $\boldsymbol{X}$ を生成するための関数を表す.

生成ステップをそれぞれ示す.

**Algorithm 1** Graph generation steps of our method

**Require:** $n, (m \text{ or } (\theta_{min} \text{ and } \theta_{max})), d, k_1, k_2, \alpha, \beta, \gamma,$
$(\phi_d, \delta_d \text{ or } \sigma_d), (\phi_c, \delta_c \text{ or } \sigma_c), f_S, f_X, r, (\phi_V, \delta_V \text{ or } \sigma_V),$
$(\phi_H, \delta_H \text{ or } \sigma_H), att_{ber}, att_{pow}, att_{uni}, att_{nor},$
$\phi_{att\_min}, \phi_{att\_max}, \delta_{att}, \sigma_{att\_min}, \sigma_{att\_max}$

**Ensure:** $\boldsymbol{S}, \boldsymbol{X}, \boldsymbol{C}$

1: # Cluster assignment
2: $\boldsymbol{\chi} \leftarrow (power\,law(\phi_c), uniform(\delta_c), normal(\sigma_c))$
3: $\xi \leftarrow Dirichlet(\alpha\boldsymbol{\chi}/\sum\boldsymbol{\chi})$
4: **for** $i = 1$ to $n$ **do**
5: $\quad \boldsymbol{U}_i \leftarrow mutinomial(\xi)$
6: $\quad \boldsymbol{C}_i \leftarrow \arg\max_j(\boldsymbol{U}_{ij})$
7: **end for**
8: # Edge construction
9: Edge_construction() # Algorithm2
10: # Attribute generation
11: $\boldsymbol{\pi} \leftarrow (power\,law(\phi_V), uniform(\delta_V), normal(\sigma_V))$
12: $\boldsymbol{\omega} \leftarrow Dirichlet(\beta\boldsymbol{\pi}/\sum\boldsymbol{\pi})$
13: **for** $a = 1$ to $d$ **do**
14: $\quad \boldsymbol{V}_a \leftarrow multinomial(\boldsymbol{\omega})$
15: **end for**
16: $\boldsymbol{\tau} \leftarrow (power\,law(\phi_H), uniform(\delta_H), normal(\sigma_H))$
17: $\boldsymbol{\psi} \leftarrow Dirichlet(\gamma\boldsymbol{\tau}/\sum\boldsymbol{\tau})$
18: **for** $b = 1$ to $k_1$ **do**
19: $\quad \boldsymbol{H}_b \leftarrow multinomial(\boldsymbol{\psi})$
20: **end for**
21: Attribute_generation() # Algorithm3

まず, ノードをクラスタに割り当てるクラスタ割当ステップが Algorithm 1 の 1–7 行目で行われる. コミュニティサイズのリスト $\boldsymbol{\chi}$ はユーザ指定可能な分布によって導出される (2 行目). その後, 各ノードについて $\alpha\boldsymbol{\chi}$ に基づいたディリクレ分布から, 多項分布が導かれる (3 行目). ここでの $\alpha$ はクラスタ間と内のエッジの割合を調整するパラメータであり, $\alpha$ が小さいとき, グラフはクラスタ内エッジを持つ割合が大きい. この多項分布はノードのクラスタ割合 $\boldsymbol{U}$ を表し, クラスタ割当 $\boldsymbol{C}$ は $\boldsymbol{U}$ から得られる [5].

**Algorithm 2** Edge construction step

1: # Edge construction
2: $\boldsymbol{\theta} \leftarrow (power\,law(\phi_d), uniform(\delta_d), normal(\sigma_d))$
3: **for** $i = 1$ to $n$ **do**
4: $\quad counter \leftarrow 0$
5: $\quad$ **while** $counter < r$ and $\sum_l^n S_{il} < \theta_i$ **do**
6: $\quad\quad$ # Candidate selection
7: $\quad\quad \boldsymbol{M} \leftarrow \{\}$
8: $\quad\quad$ **for** $j = 1$ to $\theta_i$ **do**
9: $\quad\quad\quad \boldsymbol{M} \leftarrow \boldsymbol{M} \bigcup Rand_{int}(1, n)$
10: $\quad\quad$ **end for**
11: $\quad\quad \boldsymbol{M} \leftarrow unique(\boldsymbol{M})$
12: $\quad\quad$ # Edge Generation
13: $\quad\quad$ **for** $j \in \boldsymbol{M}$ **do**
14: $\quad\quad\quad$ **if** $S_{ij} == 0$ and $\sum_l^n S_{il} < \theta_i$ and $\sum_h^n S_{hj} < \theta_j$ **then**
15: $\quad\quad\quad\quad S_{ij} \leftarrow f_S(\boldsymbol{U})$ (e.g. $Poisson(< \boldsymbol{U}_i, \boldsymbol{U}_j >)$)
16: $\quad\quad\quad\quad S_{ji} \leftarrow S_{ij}$
17: $\quad\quad\quad$ **end if**
18: $\quad\quad$ **end for**
19: $\quad\quad counter \leftarrow counter + 1$
20: $\quad$ **end while**
21: **end for**

**Algorithm 3** Attribute generation step

1: # Attribute generation
2: $\boldsymbol{X} \leftarrow f_X(\boldsymbol{U}, \boldsymbol{H}, \boldsymbol{V})$ (e.g. $(power\ law, uniform, normal)(\boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\top)$ for numerical values, and $Bernoulli(\boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\top)$ for categorical values)

次に Algorithm 2 に示すエッジ生成ステップについて述べる. ユーザが指定した分布からノード次数が決定される. エッジはノード次数に基づいて生成され, このプロセスはエッジの生成確率を計算する候補集合をランダムに生成する *Candidate selection* と, 実際にエッジの生成確率を計算し, それに基づいてエッジを生成する *Edge generation* に分けることができる. *Candidate selection* では, 全てのノードからランダムにエッジを接続する先のノード候補の集合 $\boldsymbol{M}$ が選ばれる (7–11 行目). *Edge generation* では, ノード $i$ とノード $j$ のエッジの生成確率を対応するクラスタ割合の内積 $< \boldsymbol{U}_i, \boldsymbol{U}_j >$ で表し, その値に対してポアソン分布を適用することによってエッジを生成する (13–17 行目). この手続きは全てのノード次数を適切に満たせない場合があるので, ノード次数の

5 : 本研究では, 各ノードは $\boldsymbol{U}$ の対応する行の最大値を持つ列, すなわちクラスタに割り当てられる.

みの制約では反復が終わらない可能性がある．そのため，$r$ 回のループを終了したときに，ループから抜けるように設計されている (18–21 行目)．

属性クラスタ割合 $\boldsymbol{V}$ およびクラスタ転移行列 $\boldsymbol{H}$ は，ユーザが指定した分布およびクラスタの純度を調整する $\beta, \gamma$ を入力ベクトルとしたディリクレ分布によって，それぞれ生成される (Algorithm 1 の 11–20 行目)．そして，属性行列 $\boldsymbol{X}$ は行列 $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{H}$ から計算される．Algorithm 3 の例では，$\boldsymbol{H}$ は $\boldsymbol{U}$ と $\boldsymbol{V}$ の間の転移行列であるため，$\boldsymbol{X}$ は $\boldsymbol{UHV}^\top$ で表現されるものを挙げているが，NAGC と同様に関数 $f$ を導入して，$\boldsymbol{U}$ と $\boldsymbol{V}$ の間に非線形な関係を仮定することも可能である．

### 4.2 属 性 生 成

提案手法では，属性の値のため複数の分布を選択することができ，連続値と離散値の両方を属性の値に用いることができる．実世界では，多くの現象がべき乗則や正規分布に従うことがよく知られている [18]．

$att_{ber}, att_{pow}, att_{uni}, att_{nor}$ によって，複数の分布に従う属性の数を調整することができる．離散値をとる属性数は $d \times att_{ber}$ で表すことができ，これらの属性に関して，$\boldsymbol{X}$ の対応する列にポアソン分布を適用する．連続値をとる属性についても同様に，$att_{pow}, att_{uni}, att_{nor}$ を用いて表される属性数に対応する $\boldsymbol{X}$ の列に，べき乗則，一様分布，正規分布を適用する．またクラスタの情報から独立した属性を生成するために，$1-(att_{ber}+att_{pow}+att_{uni}+att_{nor})$ の割合の属性はランダムに生成される．この場合，複数の分布の中からランダムに従う分布が選択される．$\phi_{att\_min}, \phi_{att\_max}, \delta_{att}, \sigma_{att\_min}, \sigma_{att\_max}$ はランダム生成のときの分布のパラメータである．各属性は最小値が 0 となり，最大値が 1 となるように正規化される．

### 4.3 計 算 量

ここでは，提案したグラフ生成機構の空間計算量と時間計算量の両方について議論する．グラフに関する研究の多くの場面で，我々が検討するのはスパースグラフ [20] であり，$m$ は $n^2$ に比べて非常に小さいと考えることができる．

a） 空間計算量

提案手法の中で，最もサイズの大きい行列は隣接行列 $\boldsymbol{S}$ であり，そのサイズは $n \times n$ である．スパース条件下では，隣接行列のほとんどの要素が 0 であるため，メモリのほとんどはそれらの 0 を保持するために使われる．この状態を避けるために，隣接リストを用いた他の表現方法があり，その方法ではエッジ数だけ要素を保持すれば良い．他の行列サイズは表 2 に示す通りであるため，空間計算量は $O(m+nd)$ となる．

b） 時間計算量

Algorithm 1,2,3 に示す 3 つのステップについてそれぞれ考える．*Cluster assignment* では，クラスタ割合とクラスタ割当がノードごとに行われ，このステップは $O(nk_1)$ を必要とする．次に，*Edge construction* は *Candidate selection* と *Edge Generation* に分けられる．*Candidate selection* では，ノード次数の長さを持つ候補リストが生成される．そして，*Edge Generation* では，エッジは候補リストの中のノードとのクラスタポーションに基づいて生成される．これらの 2 つのステップが各ノードに対して実行されるので，*Edge construction* は $O(nk_1 r\theta_{Avg})$ となる．さらに，$c$ を定数として $r = c \times k_1$ と $m = n\theta_{Avg}$ であることから，時間計算量は $O(mk_1^2)$ となる．最後に *Attribute generation* では，$\boldsymbol{V}$ と $\boldsymbol{H}$ は各属性と各クラスタ割当に対してそれぞれ生成される．属性行列 $\boldsymbol{X}$ のための計算量は，$k = min(k_1, k_2)$ としたとき $O(ndk)$ である．それゆえ，合計の時間計算量は $O(mk_1^2 + ndk)$ である．通常は $m \gg k_1$ であることから，計算時間はエッジ数に対して線形に決定される．

## 5 実　　験

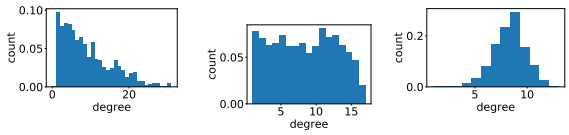
本実験では，4 つの目標がある．最初の目標は，ノード次数，コミュニティサイズおよび属性の値に対して，異なる分布を指定したときに得られる人工データを可視化することで，指定通りにグラフが生成されているかを確認することである．2 つ目の目標は，多様なエッジ数，属性数，クラスタ数に対してグラフを生成し，acMark のスケーラビリティを明らかにすることである．3 つ目の目標として，パラメータを調整することによって，クラスタの分離度を変更できることを示す．最後に，acMark にクラスタリング手法を適用し，既存のクラスタリング手法の特性を明らかにする．また本実験では，$f_S$ および $f_X$ は Algorithm2,3 で例に示すものを利用した．

### 5.1 多様なグラフプロパティでの可視化

ここでは，acMark で $n = 1000, m = 4000, k_1 = 5$ としてパラメータを設定して，グラフを生成した．ノード次数に関して，べき乗則，一様分布，正規分布をそれぞれ適用した 3 つのグラフのノード次数のヒストグラムを図 2 に示す．これらの図からノード次数が指定した分布に従うようにグラフが生成されていることがわかる．次に，コミュニティサイズに関して，べき乗則，一様分布，正規分布をそれぞれ適用した 3 つのグラフを図 3 に示す．この図から，指定した分布に従ってコミュニティが生成されていることがわかる．

### 5.2 スケーラビリティ

ここでは，多様なパラメータの値に対してグラフを生成し，実行時間とメモリ消費の変化を示す．特に記述がなければ，各パラメータは以下に示す値に設定す




(a) べき乗則に従ってノード次数を生成 ($\phi_d = 3$).　(b) 一様分布に従ってノード次数を生成.　(c) 正規分布に従ってノード次数を生成 ($\sigma_d = 0.1$).

図 2: 提案手法が生成したノード次数に関して 3 つの分布に従うグラフのノード次数分布のヒストグラム.

(a) コミュニティサイズはべき乗則に従う ($\phi_c = 2$).　(b) コミュニティサイズは一様分布に従う.　(c) コミュニティサイズは正規分布に従う ($\sigma_c = 0.1$).

図 3: 提案手法が生成したコミュニティサイズに関して 3 つの分布に従うグラフの可視化. ノードの色のはそのノードのクラスタ割当先を示す. これら実験では, $n = 1000, m = 4000$ としている. ノード次数はべき乗則に従う ($\phi_d = 3$).

る: $n = 10000, m = 100000, d = 100, k_1 = 10, k_2 = 10, \alpha = 1/k_1, \phi = 3, \psi = 2, \sigma_d = 0.1, \sigma_c = 0.1, r = 10 * k_1, att_{poi} = 0.0, att_{pow} = 0.4, att_{uni} = 0.1, att_{nor} = 0.4$.

#### 5.2.1 エッジ数

エッジ数 $m$ は $\{10^4, 10^5, 10^6, 10^7\}$ の中から選ばれ, スパース条件下を維持するために $n = m/10$ とする. 図 4a が示すように, 実行時間はエッジ数に関して線形に増加している. また図 4b に注目すると, 実際に生成されたエッジが $m$ に対して小さいことがわかる. これはエッジ生成アルゴリズムが, 確率分布から決定したノード次数を完全に満たすことができないことから生じている. $r$ をより大きな値に設定したとき, 実際のエッジ数は $m$ に近づく. $\boldsymbol{S}$ のサイズがエッジ数に対して, 線形に増加していくことが図 4c からわかる. これらの実験から時間および空間計算量の両方がエッジ数に関して線形にスケールすることが示された.

#### 5.2.2 属性数

属性数 $d$ は $\{10^2, 10^3, 10^4\}$ の中から選ばれる. 図 5 において, 属性に関する提案手法のスケーラビリティを示す. 図 5a は, 実行時間が属性数に対して線形に増加していることを示している. また図 5b において属性行列のサイズが線形に増加することを示している.

### 5.3 クラスタ分離度

図 6 が示すように, $\alpha$ が小さいときクラスタ内エッジが増加することがわかる. これはディリクレ分布の特徴の 1 つであり, $\alpha$ をより小さくすれば, 各クラスタが他のクラスタより分離する. $\beta$ と $\gamma$ についても同様に実験を行う. 図 7 が, これらのパラメータが属性のクラスタ分離度を調整することができることを示す. $\beta$ と $\gamma$ が大きいとき, クラスタがより重なり合っていることがわかる. ディリクレ分布の特徴から, 0.1 などの小さな値に $\beta$ を設定したとき, 各構造に関するクラスタは非常に少ない数の属性に関するクラスタとのみ関係を持つ.

### 5.4 クラスタリング手法の適用

本節では, acMark が生成する人工グラフデータによって, クラスタリング手法の特性を明らかにできることを示す. まず, acMark によってグラフデータを生成し, それらのデータに対して, 属性付きグラフクラスタリング手法である NAGC, グラフクラスタリング手法である METIS [11], 属性に関するクラスタリング手法である k-means をそれぞれ適用し, その結果を示す. グラフ生成は確率分布に基づくため, 同じパラメータで 5 つのグラフを作成する. また, 初期値依存がある手法を用いているため, それぞれ 5 回実行する. そのため, 合計 25 回実行し, その平均と標準偏差を結果として示すものとする.

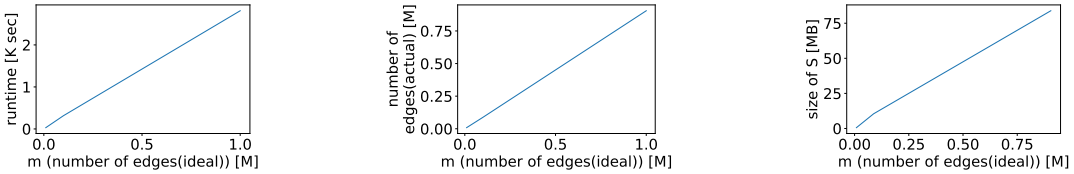
#### 5.4.1 データセット

実験に用いる人工グラフの詳細を表 4 に示す. acMark1 では, 属性のうちの 9 割が正規分布に従う. acMark2 では, 属性はべき乗則, 一様分布, 正規分布に 1 割ずつ従い, 残りの 7 割はクラスタとは独立してランダムに生成される. acMark3 は属性の従う分布について, acMark2 と同じ条件を持つ一方で, $\alpha$ が小さいためクラスタの分離度が高いグラフである.

#### 5.4.2 クラスタリング性能

ここでは 4 つの指標を用いて評価する. クラスタリング結果と正解データとの比較のために, Normalized Mutual Information (NMI) と ARI [25] を用いた. また構造の評価にモジュラリティ [19] を用い, 属性の評価に情報エントロピーを用いた. モジュラリティは, クラスタ内でエッジが密であり, クラスタ間でエッジが疎であれば, 値が高くなる指標である.

生成したデータにクラスタリング手法を適用した結果を表 5 に示す. まず, クラスタリング性能を示す NMI と ARI について議論する. acMark1 は, その属性の 9 割が正規分布に従うため, 属性と正解クラスタの関係が強いデータと言える. クラスタリング結果を見ると, NAGC と k-means が良い性能を示していることから, これらの手法は属性の情報をクラスタリング結果に組み込めていることがわかる. 次に acMark2 では, acMark1 に比べて属性と正解クラスタの関係が弱くなっている. そのため, このデータの正解クラスタを抽出するには構造と属性をうまく組み合わせる必要があると言える. NAGC




(a) 実行時間. (b) 実際に生成されたエッジ数と $m$ の比較. (c) $\boldsymbol{S}$ のサイズ.

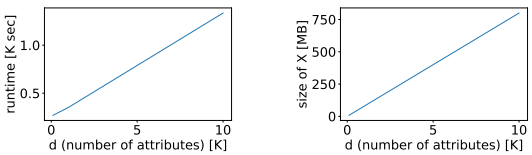
図 4: エッジ数に関するスケーラビリティの実験. その他のパラメータは以下のように設定する: $n = m/10$, $d = 100$, $k_1 = 10$, $k_2 = 10$.

表 4: 生成したデータセット

| ID | $n$ | $m$ | $d$ | $k_1$ | $k_2$ | $\alpha$ | $\beta$ | $\gamma$ | $att_{pow}$ | $att_{uni}$ | $att_{nor}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| acMark1 | 1000 | 4000 | 10 | 5 | 5 | 0.2 | 10 | 1 | 0.0 | 0.0 | 0.9 |
| acMark2 | 1000 | 4000 | 10 | 5 | 5 | 0.2 | 10 | 1 | 0.1 | 0.1 | 0.1 |
| acMark3 | 1000 | 4000 | 10 | 5 | 5 | 0.1 | 10 | 1 | 0.1 | 0.1 | 0.1 |

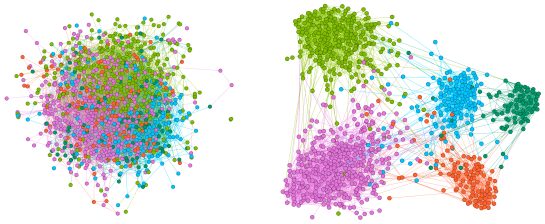
表 5: クラスタリング性能. Oracle は正解データをクラスタリング結果として評価したものである. 括弧内は標準偏差を意味する.

| データセット | 手法 | 入力タイプ | NMI | ARI | モジュラリティ | エントロピー |
|---|---|---|---|---|---|---|
| acMark1 | Oracle | | 1.000(±0.000) | 1.000(±0.000) | 0.571(±0.030) | −0.787(±0.045) |
| | NAGC | Topology, Attribute | 0.751(±0.000) | 0.805(±0.002) | 0.579(±0.000) | −0.622(±0.000) |
| | METIS | Topology | 0.553(±0.043) | 0.512(±0.060) | 0.534(±0.033) | −0.588(±0.068) |
| | k-means | Attribute | 0.620(±0.028) | 0.547(±0.041) | 0.348(±0.048) | −0.768(±0.062) |
| acMark2 | Oracle | | 1.000(±0.000) | 1.000(±0.000) | 0.495(±0.094) | −0.450(±0.147) |
| | NAGC | Topology, Attribute | 0.662(±0.008) | 0.741(±0.001) | 0.524(±0.002) | −0.320(±0.006) |
| | METIS | Topology | 0.475(±0.080) | 0.393(±0.135) | 0.497(±0.046) | −0.268(±0.085) |
| | k-means | Attribute | 0.321(±0.063) | 0.209(±0.056) | 0.167(±0.026) | −0.394(±0.047) |
| acMark3 | Oracle | | 1.000(±0.000) | 1.000(±0.000) | 0.653(±0.021) | −0.412(±0.048) |
| | NAGC | Topology, Attribute | 0.834(±0.014) | 0.890(±0.010) | 0.599(±0.005) | −0.321(±0.008) |
| | METIS | Topology | 0.669(±0.044) | 0.606(±0.053) | 0.611(±0.030) | −0.290(±0.046) |
| | k-means | Attribute | 0.406(±0.099) | 0.278(±0.088) | 0.182(±0.052) | −0.346(±0.063) |




(a) 実行時間. (b) $\boldsymbol{X}$ のサイズ

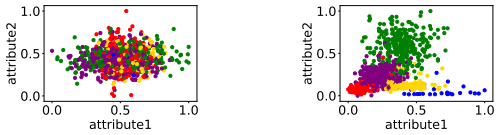
図 5: 属性数に関しての実行時間とメモリ消費の変化. その他のパラメータは以下のように設定する:$n = 10000$, $m = 10000$, $k_1 = 10$, $k_2 = 10$.



(a) $\alpha = 1.0$. (b) $\alpha = 0.1$

図 6: 構造に関するクラスタ分離度. これらの実験では, $n = 1000, m = 4000, k_1 = 5$ とし, ノードの色はそのノードのクラスタ割当先を表す.

が最も良いクラスタリング精度を示していることから, NAGC が構造と属性の両方を考慮できていることがわかる. acMark3 では, $\alpha$ の変化によって, クラスタの分離度が高まっている. そのため, いずれの手法でもクラスタリング精度が acMark2 に比べ高くなっている. Oracle のモジュラリティに注目すると, acMark3 でのスコアは他の 2 つのデータより高くなっており, 構造に関してクラスタの分離度が高くなっていることがわかる. 次にエントロピーに注目すると, Oracle の acMark1 と acMark2 での結果の差異から, ランダムに生成される属性が増えると, エントロピーの評価が悪化していることがわかる. また acMark2 と acMark3 との差異から, クラスタの分離度が上がると, エントロピーが良くなっていることがわかる. いずれの結果もデータ生成時のパラメータから推察できる特性と一致している. このことから, 様々なデータでクラスタリング手法を実験することによって, それらの手法の特性を明らかにできることを示した.




(a) $\beta = 100, \gamma = 100$.　　(b) $\beta = 1, \gamma = 1$

図 7: 属性に関するクラスタの分離度．これらの実験では, $n = 1000, k_1 = 5$ とし, ノードの色はそのノードのクラスタ割当先を表す.

## 6 おわりに

コミュニティ構造を考慮した属性付きグラフ汎用生成機構である acMark を提案した．acMark は生成するグラフのノード次数, コミュニティサイズ, 属性の値それぞれに任意の分布を仮定することが可能であり, 既存手法に比べ多様なグラフを生成することができる．また時間計算量および空間計算量ともに, エッジ数と属性数に関して線形であるため, 効率的なグラフ生成を実現した．acMark によって生成した属性付きグラフに対し, 実際にクラスタリング手法を適用することによって, それらのグラフが期待した特徴を保持して生成されることを示した.

### 文　献


[1] Leman Akoglu, Hanghang Tong, Brendan Meeder, and Christos Faloutsos. PICS: Parameter-free identification of cohesive subgroups in large attributed graphs. In *Proceedings of SIAM SDM*, 2012.

[2] Oualid Benyahia, Christine Largeron, Baptiste Jeudy, and Osmar R Zaïane. Dancer: Dynamic attributed network with community structure generator. In *ECML PKDD*. Springer, 2016.

[3] Aleksandar Bojchevski and Stephan Günnemann. Bayesian Robust Attributed Graph Clustering: Joint Learning of Partial Anomalies and Group Structure. In *AAAI*, 2018.

[4] Sylvain Brohee and Jacques Van Helden. Evaluation of clustering algorithms for protein-protein interaction networks. *BMC bioinformatics*, 2006.

[5] Santo Fortunato. Community detection in graphs. *Physics reports*, 2010.

[6] Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. Cypher: An Evolving Query Language for Property Graphs. In *Proceedings of SIGMOD*, 2018.

[7] Betsy George, Sangho Kim, and Shashi Shekhar. Spatio-temporal network databases and routing algorithms: A summary of results. In *International Symposium on Spatial and Temporal Databases*, 2007.

[8] Zhichao Huang, Yunming Ye, Xutao Li, Feng Liu, and Huajie Chen. Joint weighted nonnegative matrix factorization for mining attributed graphs. In *Proceedings of PAKDD*, 2017.

[9] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-RNN: Deep learning on spatio-temporal graphs. In *Proceedings of CVPR*, 2016.

[10] Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical review E*, 2011.

[11] George Karypis and Vipin Kumar. Multilevelk-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed computing*, 1998.

[12] Leonard Kaufmann and Peter Rousseeuw. Clustering by means of medoids. *Data Analysis based on the L1-Norm and Related Methods*, pp. 405–416, 01 1987.

[13] Brian Kulis, Sugato Basu, Inderjit Dhillon, and Raymond Mooney. Semi-supervised graph clustering: a kernel approach. *Machine learning*, 2009.

[14] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 2008.

[15] Christine Largeron, Pierre-Nicolas Mougel, Reihaneh Rabbany, and Osmar R Zaïane. Generating attributed networks with communities. *PloS one*, 2015.

[16] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[17] Seiji Maekawa, Koh Takeuch, and Makoto Onizuka. Non-linear Attributed Graph Clustering by Symmetric NMF with PU Learning. *arXiv preprint arXiv:1810.00946*, 2018.

[18] Mark EJ Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 2005.

[19] Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 2006.

[20] Mark EJ Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[21] M. Parimala and D. Lopez. Graph clustering based on Structural Attribute Neighborhood Similarity (SANS). In *Proceedings of IEEE ICECCT*, 2015.

[22] Patrick O Perry and Patrick J Wolfe. Null models for network data. *arXiv preprint arXiv:1201.5871*, 2012.

[23] Martin Sevenich, Sungpack Hong, Oskar van Rest, Zhe Wu, Jayanta Banerjee, and Hassan Chafi. Using Domain-specific Languages for Analytic Graph Databases. *PVLDB*, 2016.

[24] Zhiqiang Xu, Yiping Ke, Yi Wang, Hong Cheng, and James Cheng. A model-based approach to attributed graph clustering. In *SIGMOD*. ACM, 2012.

[25] Ka Yee Yeung and Walter L Ruzzo. Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data. *Bioinformatics*, 2001.