知 能 と 複 雑 系 128−3
( 2 0 0 2 . 5 . 2 3 )

# Apriori-based Graph Mining アルゴリズムの高速化

西村 芳男* , 鷲尾 隆* , 吉田 哲也* , 元田 浩*, 猪口 明博†

大阪大学産業科学研究所*, 日本アイ・ビー・エム (株) 東京基礎研究所†

**和文抄録:** Apriori-based Graph Mining (AGM) アルゴリズムは Apriori アルゴリズムをグラフ構造データに拡張したアルゴリズムであり，グラフ構造データから特徴的なパターンを抽出する．提案手法は，AGM アルゴリズムの多頻度グラフの候補生成において異なった条件を使用し，その候補生成の計算時間を短縮する．提案手法を実装したシステムを構築し，人工データによる性能評価と提案手法の特徴分析を行う．さらに，提案手法を実際のデータに適用し，AGM アルゴリズム全体の計算時間も高速化できることを示す．

# A Faster Apriori-based Graph Algorithm

Yoshio Nishimura* , Takashi Washio* , Tetsuya Yoshida* , Hiroshi Motoda*,
Akihiro Inokuchi†

I.S.I.R.,Osaka University* , IBM Japan, Ltd. Tokyo Research Laboratory†

**Abstract:** Apriori-based Graph Mining (AGM) algorithm derives subgraph patterns efficiency which frequently appear in database consisting of graph structure data. In this paper, we propose a new and faster algorithm of AGM achieved by adding a condition to generate candidate frequent graphs. Simulation experiments were carried out to evaluate the performance of the proposed new AGM algorithm over the synthesized data and real world chemical data. Efficient reduction of the computation time by the proposed method has been confirmed.

## 1 まえがき

グラフは普遍的な表現方法の一つで，さまざまな分野において利用され，データベースに膨大なグラフ構造データが蓄積されるようになってきた．例えば，医学や化学の分野における化学物質の分子構造，パターン認識のパターン，通信のトラフィックなどさまざまな分野でグラフ構造データは広く利用されている．このようなグラフ構造データから特徴的なパターンを抽出するさまざまな手法が提案されている [Yoshida 95] [Motoda 97] [Dehaspe 98] [Kuramochi 01] [Raedt 01]．これらの手法の多くは計算時間の観点から Greedy 探索を用いるものや，抽出されるグラフ構造に制限のあるものが多い．一方で，抽出されるグラフ構造に制限がなく，完全探索を行う AGM アルゴリズムが提案されており [Inokuchi 00]，その効率化も行われている [猪口 01]．本稿では，AGM アルゴリズムの多頻度グラフの候補生成に新たな条件を付加して，その候補生成の計算時間を短縮する手法の提案を行う．

## 2 AGM アルゴリズム

### 2.1 定義

AGM アルゴリズム [Inokuchi 00] で扱うグラフは頂点，辺にラベルを持ち，以下のように定義される．

連絡先：大阪大学産業科学研究所
〒 567-0047 大阪府茨木市美穂ヶ丘 8-1
e-mail: nishimura@ar.sanken.osaka-u.ac.jp

**定義 1 (ラベル付きグラフ)** 頂点の集合 $V(G)$，辺の集合 $E(G)$，頂点のラベル集合 $L_V(V(G))$，辺のラベル集合 $L_E(E(G))$ が

$$\begin{aligned} V(G) &= \{\ v_1, v_2, ..., v_k\ \}\ , \\ E(G) &= \{\ e_h = (v_i, v_j) | v_i, v_j \in V(G), i \neq j\}\ , \\ L_V(V(G)) &= \{lb(v_i) | \forall v_i \in V(G)\ \}\ , \\ L_E(E(G)) &= \{lb(e_h) | \forall e_h \in E(G)\ \} \end{aligned}$$

と与えられたとき，グラフ $G$ は

$$G = (V(G), E(G), L_V(V(G)), L_E(E(G)))$$

と表現される．ここで，頂点の数 $|V(G)| = k$ をグラフ $G$ の大きさとする．$lb(v_i)$ および $lb(e_h)$ はそれぞれ 頂点 $v_i$ のラベル，辺 $e_h$ のラベルである．頂点ラベル $lb(v_i)$ および辺ラベル $lb(e_h)$ にはそれぞれ $num(lb(v_i))$,$num(lb(e_h))$ によって自然数を以下のように割り当てる．

**定義 2 (ラベル間の順序関係)** グラフ構造データベースが与えられたとき，それに含まれる各ラベル $lb_i$ の頂点の数を $avg(lb_i)$ とする．$avg(lb_i)$ が少ないものから自然数を昇順に割り当てると，生成されるグラフ数は少なくなる *[猪口 01]*．つまり，

$$\begin{aligned} &if\ avg(lb_i) < avg(lb_j)\ \ then\ num(lb_i) < num(lb_j) \\ &\qquad for\ i, j = 1, ..., |L_V(V(G))|, i \neq j \end{aligned}$$

とする．辺のラベル $lb_i$ に割り当てる自然数も同様に，

$$\begin{aligned} &if\ avg(lb_i) < avg(lb_j)\ \ then\ num(lb_i) < num(lb_j) \\ &\qquad for\ i, j = 1, ..., |L_E(E(G))|, i \neq j \end{aligned}$$

とする．

**定義 3 (隣接行列)** 大きさ $k$ のグラフ $G = (V(G), E(G), L_V(V(G)), L_E(E(G)))$ が与えられたとき，隣接行列 $X_k$ の $(i,j)$ 要素 $x_{i,j}$ は

$$x_{i,j} = \begin{cases} num(lb(e_h)) & if\ e_h = (v_i, v_j) \in E(G) \\ 0 & if\ (v_i, v_j) \notin E(G) \end{cases}$$

で与えられる．さらに，グラフ $G$ は頂点ラベルに割り当てられた自然数によって，

$$num(lb(v_i)) \leq num(lb(v_{i+1}))\ for\ i = 1, ..., k-1$$

の条件を満たすように行と列をソートする．

**定義 4 (グラフのコード)** 隣接行列のコード $code(X_k)$ は隣接行列 $X_k$ の要素 $x_{i,j}$ を並べたものであり，有向グラフの場合には，

$$code(X_k) = x_{1,2}x_{2,1}x_{1,3}x_{3,1}x_{2,3}x_{3,2}\cdots x_{k-1,k}x_{k,k-1}$$

と定義する．無向グラフの場合は $x_{i,j} = x_{j,i}$ であるため冗長なものを省いて，

$$code(X_k) = x_{1,2}x_{1,3}x_{2,3}x_{1,4}\cdots x_{k-2,k}x_{k-1,k}$$

と定義する．さらに頂点ラベルを含めたグラフのコード $CODE(G(X_k))$ を

$$CODE(G(X_k)) = num(lb(v_1))\cdots num(lb(v_k))code(X_k)$$

と定義する．

**定義 5 (正準形)** グラフ $G(X_k)$ と同じ構造をもつグラフの集合を $\Gamma(G(X_k))$ としたとき，$\Gamma(G(X_k))$ の中で最小の $CODE$ を持つグラフ $g$ を正準形 *(canonical form)* とする．

$$g \quad w.r.t \quad CODE(g) = \min_{\forall g_i \in \Gamma(G(X_k))} CODE(g_i)$$

**定義 6 (誘導部分グラフ)** グラフ $G = (V(G), E(G), L_V(G), L_E(G))$ が与えられたとき，$G$ の誘導部分グラフ $G_s = (V(G_s), E(G_s), L_V(G_s), L_E(G_s))$ は以下の条件を満たす．

$$V(G_s) \subseteq V(G)\ ,$$
$$E(G_s) = E(G) \cap (V(G_s) \times V(G_s))\ ,$$
$$lb(v_i) = lb(f(v_i))\ , \forall v_i \in V(G_s)\ ,$$
$$lb((v_i, v_j)) = lb((f(v_i), f(v_j))), \forall e_h = (v_i, v_j) \in E(G_s)$$

ここで，$f$ は $V(G_s) \to V(G)$ の写像を表し，単射である．

**定義 7 (支持度)** グラフ構造データベース $GD$ が与えられたとき，グラフ $G_s$ の支持度 $sup(G_s)$ は以下のように定義される．

$$sup(G_s) = \frac{G_s\text{を誘導部分グラフとして含むグラフの数}}{GD\text{ に含まれるグラフの総数}}$$

グラフ $G_s$ の支持度 $sup(G_s)$ がユーザによって指定された最小支持度 *(minsup)* を上回る場合，グラフ $G_s$ を多頻度グラフと呼ぶ．

## 2.2 概略

AGM アルゴリズムは，Apriori アルゴリズム [Agrawal 94] をグラフ構造データに拡張したアルゴリズムであり，猪口によって提案された [Inokuchi 00]．AGM アルゴリズムはグラフ構造データベース $GD$ にある最小支持度以上の頻度で誘導部分グラフとして含まれるグラフ構造を効率よく抽出するアルゴリズムである．抽出する多頻度グラフは，大きさが 1 のものから逐次的に大きな多頻度グラフを抽出する．図 1 に AGM アルゴリズムの概略を示す．始めに，大きさが 1 の多頻度グラフを $F_1$ に代入する．次に，関数 apriori-gen-join では，大きさが $k$ の多頻度グラフから大きさ $k+1$ の多頻度グラフの候補を生成し，それを $\hat{C}_{k+1}$ に代入する．次に，関数 apriori-gen-prune では $\hat{C}_{k+1}$ に格納されている各多頻度グラフの候補について，多頻度グラフであるための必要条件を調べる．この条件を調べることで多頻度グラフの候補を絞りこみ，残ったグラフのみを $C_{k+1}$ に格納する．次に，関数 count ではグラフ構造データベース $GD$ にアクセスして，$C_{k+1}$ の各要素の支持度を求める．$C_{k+1}$ の各要素の支持度が最小支持度を上回る場合は，そのグラフを多頻度グラフとし，それを $F_{k+1}$ に格納する．以上の操作を $F_k$ が空集合になるまで繰り返し，グラフ構造データベース $GD$ に含まれる多頻度グラフをすべて抽出する．

```
// GD:グラフ構造データベース
// F_k:大きさ k の多頻度グラフの集合
// Ĉ_{k+1}:大きさ k の多頻度グラフを合成したものの集合
// C_k:大きさ k の多頻度グラフの候補の集合
// minsup:最小支持度
1) F_1 = { Frequent subgraph of size=1} ;
2) for(k = 1; F_k ≠ ∅; k++) do begin
3)    Ĉ_{k+1} =apriori-gen-join(F_k);
4)    C_{k+1} =apriori-gen-prune(Ĉ_{k+1});
5)    count(GD, C_{k+1});
6)    F_{k+1} ={ c_{k+1} ∈ C_{k+1} | sup(G(c_{k+1})) ≥ minsup};
7) end
8) Answer= ⋃_k F_k;
```

図 1: AGM アルゴリズム

## 2.3 多頻度グラフの候補生成 (join 部)

多頻度グラフの候補生成は，大きさが $k$ の多頻度グラフを合成して大きさ $k+1$ の多頻度グラフの候補を作成する join 部と，合成された多頻度グラフの候補が多頻度グラフになるための必要条件を満たすかどうかを調べる prune 部の 2 つの部分から成り立つ．join 部では以下の条件を満たすように多頻度グラフの候補 $G(Z_{k+1})$ を順に生成していく．

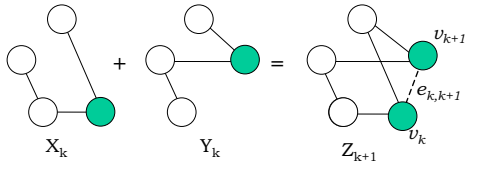


図 2: 多頻度グラフの候補生成例

**条件 1** 大きさが $k$ の多頻度グラフを *2* つ考え，その隣接行列を $X_k$，$Y_k$ とする． $X_k$，$Y_k$ の $k$ 行及び $k$ 列以外の要素が全て等しいとき，すなわち各グラフの第 $k$ 頂点を除いて構造が等しいとき，以下のように $X_k$，$Y_k$ を結合し，大きさ $k+1$ の 隣接行列 $Z_{k+1}$ を生成する．

$$X_k = \begin{pmatrix} X_{k-1} & \boldsymbol{x}_1 \\ \boldsymbol{x}_2^T & 0 \end{pmatrix}, Y_k = \begin{pmatrix} X_{k-1} & \boldsymbol{y}_1 \\ \boldsymbol{y}_2^T & 0 \end{pmatrix}$$

$$Z_{k+1} = \begin{pmatrix} X_{k-1} & \boldsymbol{x}_1 & \boldsymbol{y}_1 \\ \boldsymbol{x}_2^T & 0 & z_{k,k+1} \\ \boldsymbol{y}_2^T & z_{k+1,k} & 0 \end{pmatrix}$$

ここで，$X_{k-1}$ は大きさ $k-1$ のグラフの隣接行列，$\boldsymbol{x}_i, \boldsymbol{y}_i (i = 1, 2)$ は $(k-1)\times 1$ の縦ベクトルである．$G(X_k)$，$G(Y_k)$ をそれぞれ $G(Z_{k+1})$ の第 *1* 生成グラフ，第 *2* 生成グラフと呼ぶ．

**条件 2** $i = 1, \cdots, k-1$ として，生成される $G(Z_{k+1})$ の頂点ラベルには以下の条件がある．

$$\begin{aligned}
lb(v_i \in V(G(Z_{k+1}))) &= lb(v_i \in V(G(X_k))) \\
&= lb(v_i \in V(G(Y_k))), \\
num(lb(v_i \in V(G(X_k)))) &\leq num(lb(v_{i+1} \in V(G(X_k)))), \\
lb(v_k \in V(G(Z_{k+1}))) &= lb(v_k \in V(G(X_k))), \\
lb(v_{k+1} \in V(G(Z_{k+1}))) &= lb(v_k \in V(G(Y_k))), \\
num(lb(v_k \in V(G(X_k)))) &\leq num(lb(v_k \in V(G(Y_k))))
\end{aligned}$$

ただし，隣接行列 $Z_{k+1}$ の $(k, k+1)$ 要素 $z_{k,k+1}$ および $(k+1, k)$ 要素 $z_{k+1,k}$ は $X_k, Y_k$ から決定することはできない．

**条件 3** そこで，隣接行列 $Z_{k+1}$ は以下の条件を満たすものすべてが作られる．すなわち，

$$\begin{gathered}
\hat{C}_{k+1} \leftarrow G(Z_{k+1}) \\
where\ z_{k,k+1} = lb1\ and\ z_{k+1,k} = lb2, \\
\forall\ lb1, 0 \leq lb1 \leq |L_E(E(G))|\ and \\
\forall\ lb2, 0 \leq lb2 \leq |L_E(E(G))|
\end{gathered}$$

である．有向グラフの場合は $(|L_E(E(G))|+1)^2$ 個のグラフが生成される．無向グラフの場合は $z_{k,k+1} = z_{k+1,k}$ であるため，$(|L_E(E(G))|+1)$ 個のグラフが生成される．

**条件 4** ここでグラフ構造 $G(X_k)$ と $G(Y_k)$ の第 $k$ 頂点のラベルが等しい場合，$G(Y_k), G(X_k)$ をそれぞれ第 *1* 生成グラフ，第 *2* 生成グラフとして *2* つのグラフを結合した場合，このグラフは冗長である．そこで，このような冗長な生成を避けるため，以下の関係にある場合にのみグラフを結合する．

$$CODE(\text{第 } 1 \text{ 生成グラフ}) \leq CODE(\text{第 } 2 \text{ 生成グラフ})$$

以上の *4* つの条件のもとで生成されるグラフを**正規形**（*normal form*）と呼ぶ．

## 2.4 多頻度グラフの候補生成 (prune 部)

前節の join 部で合成された多頻度グラフの候補 $G(Z_{k+1}) \in \hat{C}_{k+1}$ が多頻度グラフであるための必要条件は，$G(Z_{k+1})$ の全ての誘導部分グラフが多頻度グラフであることである．そこで，この必要条件と等価である以下の必要条件を調べる．

**誘導部分グラフの必要条件**
グラフ $G(Z_{k+1})$ が多頻度グラフであるための必要条件は，$G(Z_{k+1})$ の第 $i (1 \leq i \leq k-1)$ を除去してできるグラフが全て多頻度グラフであることである．

先にも述べたように，このアルゴリズムでは正規形の隣接行列しか探索生成しないために，第 $i$ 頂点を開放除去したグラフの隣接行列が正規形でなければ，それが多頻度グラフであるかを過去の探索から容易にチェックする事ができない．よって，非正規形の隣接行列を正規化する手法が必要である．

正規化の具体例を図 3 の非正規形の隣接行列 $X_4$ の正規化で示す．(A) はじめに頂点が 1 つからなる $X_4$ の部分グラフの隣接行列を考える．(B) 多数ある正規形の中で，最終的に 1 つ正規形が見つかれば十分なので，結合の組み合わせを限定し，$v_1$ を元にして結合を行う．(C) 結合により得られない情報，例えば，$v_1, v_2$ からなる隣接行列の (1,2) 要素，(2,1) 要素は元の隣接行列 $X_4$ の $x_{12}$ 及び $x_{21}$ から補う．(D) 次に頂点数が 2 の隣接行列の結合を行う．(E) このとき隣接行列のコードが最小の行列を第 1 生成行列とする．ここではコードが 0 の隣接行列が 2 つあるが，どちらか一方を選択する．以下，順に繰り返し，非正規形の隣接行列 $X_4$ を再構築し正規化された行列を得る．

上記の方法によって頂点を除去した全てのグラフが過去の探索結果から多頻度グラフであることを確定できれば $G(Z_{k+1})$ は多頻度グラフの候補となり，$C_{k+1}$ に格納される．

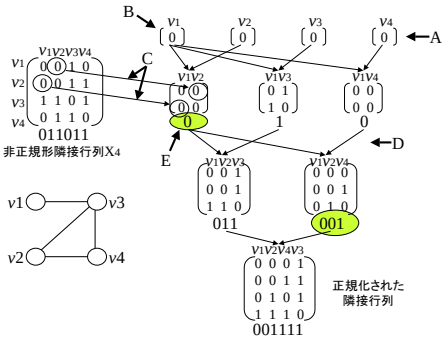


図 3: 正規化の例

全ての多頻度グラフの候補を取り出した後，実際にデータベースをスキャンして，それらの支持度を求める．しかし，正規形の中にも同じグラフ構造を表すグラフが存在する場合があるため，支持度を計算する前に正準形を求める処理が必要となる．正準形を求める処理と支持度の計算方法については文献 [Inokuchi 00] を参照されたい．$C_{k+1}$ の各要素について多頻度グラフの候補の支持度を計算して，その支持度が最小支持度を上回る場合には，その多頻度グラフの候補を多頻度グラフとして，$F_{k+1}$ に格納する．つまり，

$$\begin{gathered}
\text{if } \forall c_{k+1} \in C_{k+1} \text{ and } sup(G(c_{k+1})) \geq minsup \\
\text{then } F_{k+1} \leftarrow c_{k+1}
\end{gathered}$$

である．

表 1: 人工データのパラメータとそのデフォルト値

| パラメータ | 意味 | デフォルト値 |
|---|---|---|
| $D$ | $GD$ に含まれるグラフ数 | 1000 |
| $\|T\|$ | グラフデータの平均頂点数 | 15 |
| $L$ | 基本パターンの数 | 8 |
| $\|I\|$ | 基本パターンの平均頂点数 | 7 |
| $\|L_V\|$ | 頂点ラベルの種類数 | 5 |
| $\|L_E\|$ | 辺ラベル種類数 | 5 |
| $p$ | 頂点間に辺が存在する確率 | 4% |
| $minsup$ | 最小支持度 | 10% |

# 3 提案手法

## 3.1 多頻度グラフの候補生成のための新条件

2.3 章の図 2 のように，大きさが $k$ の多頻度グラフ $G(X_k)$ と $G(Y_k)$ を結合し，大きさが $k+1$ の多頻度グラフの候補 $G(Z_{k+1})$ を生成する場合を考える．グラフ $G(Z_{k+1})$ の要素 $z_{k,k+1}$ と $z_{k+1,k}$ は $X_k, Y_k$ から決定することができないため，辺のラベル数 $|L_E(E(G))|$ に応じて条件 3 で示された数のグラフ $G(Z_{k+1})$ が生成される．生成されたグラフ $G(Z_{k+1})$ に含まれるすべての誘導部分グラフが多頻度グラフであることを確認するために，それと等価な必要条件を確認している．この方法では合成されたすべてのグラフ $G(Z_{k+1})$ について，その誘導部分グラフを正規化する必要があるため多くの計算時間を要する．

そこで，$G(Z_{k+1})$ の頂点 $v_k, v_{k+1}$ とその頂点間の辺 $e_{k,k+1} = (v_k, v_{k+1})$ から構成される大きさ 2 のグラフ $G(Z_{S2})$ に着目する．$G(Z_{k+1})$ が多頻度グラフになるためには，その誘導部分グラフである $G(Z_{S2})$ も多頻度グラフであることが必要条件の 1 つである．つまり，$G(Z_{S2})$ が多頻度グラフでない場合は，$G(Z_{k+1})$ も多頻度グラフになり得ない．そのため，このような合成を行わない．そこで，$k \geq 2$ では条件 3 の代わりに以下の条件 3' を使用する．

**条件 3'**

$$\text{If } k \geq 2 \text{ and } \forall\ G(Z_{S2}) \in F_2 \text{ then } \hat{C}_{k+1} \leftarrow G(Z_{k+1}),$$
$$\text{where } V(G(Z_{S2})) = \{v_k, v_{k+1}\}$$

条件 1, 条件 2, 条件 4 と条件 3' を満たすグラフ $G(Z_{k+1})$ を改めて正規形と定義する．提案手法で合成された $G(Z_{k+1}) \in \hat{C}_{k+1}$ は 2.4 章で述べた必要条件を確認するために $G(Z_{k+1})$ の誘導部分グラフを正規化する必要がある．しかし，従来の AGM アルゴリズムと比較して，提案手法の $\hat{C}_{k+1}$ は要素が絞り込まれているため，正規化の計算時間を短縮できる．

また，条件 3' は必要条件の一つであるため，2.4 章ですべての必要条件が確認された後に残るグラフの数 $|C_{k+1}|$ は，AGM アルゴリズム，提案手法ともに同じである．

# 4 評価実験

本研究の提案手法を C 言語で実装し，CPU が PentiumIII 1GHz, メモリが 1.5GB 搭載された計算機を使用して，評価実験を行った．表 1 に実験で用いた人工データのパラメータとそのデフォルト値を示す．まず，平均 $|T|$, 分散 1 のガウス分布を用いて各グラフ構造データのサイズを決める．各グラフ構造データ中の頂点のラベルは等確率で決定する．次に存在確率 $p$ をもとに頂点間に辺を結ぶ．辺のラベルも等確率で決定する．同様にして，平均サイズ $|I|$ の基本パターンを $L$ 個作る．各グラフ構造データに対し，基本パターンからランダムに 1 つを選択し，上記グラフ構造データが基本パターンを誘導部分グラフとして含むように上書きする．

図 4 は辺のラベル数 $|L_E|$, 図 5 は頂点のラベル数 $|L_V|$, 図 6 はグラフデータの平均頂点数 $|T|$ を変化させた場合の，多頻度グラフの候補生成に要する計算時間を評価した結果である．AGM' は AGM の条件 3 の代わりに条件 3' を用いた提案手法を表す．また，$|\hat{C}_{sum}|$ は合成されたグラフの数，$|C_{sum}|$ は正規形の多頻度グラフ候補の数，$|F_{sum}|$ は正規形の多頻度グラフの数を大きさが 1 から最大のものまですべてを合計したものとする．表 2 は辺のラベル数 $|L_E|$, 表 3 は頂点のラベル数 $|L_V|$ 表 4 はグラフデータの平均頂点数 $|T|$ を変化させた場合の，$|\hat{C}_{sum}|$,$|C_{sum}|$,$|F_{sum}|$ の生成数を示す．なお，$|C_{sum}|$,$|F_{sum}|$ の生成数は AGM アルゴリズムと提案手法の両手法とも同じであるため，一つにまとめている．

AGM と提案手法を比較すると，辺のラベル数 $|L_E|$ および頂点のラベル数 $|L_V|$ が少ない場合に，提案手法は多頻度グラフの候補生成における候補の生成数 $|\hat{C}_{sum}|$ と計算時間をほとんど削減できていない．これはラベル数が少なく，グラフ構造データベースに多くの多頻度グラフが多く含まれているためである．しかし，辺のラベル数 $|L_E|$ および頂点のラベル数 $|L_V|$ が多い場合には，提案手法は多頻度グラフの候補生成において，生成数 $|\hat{C}_{sum}|$ と計算時間を削減できている．また，$|L_E| \geq 3$ および $|L_V| \geq 7$ のようにある閾値を超えると，抽出される多頻度グラフは基本パターンに含まれるものがほとんどであるため，提案手法の $|\hat{C}_{sum}|$ と計算時間はほぼ一定になると考えられる．

また，図 6 のようにグラフデータの平均頂点数 $|T|$ を増やした場合にも，提案手法の条件 3' は多頻度グラフの候補生成にかかる計算時間を削減できている．平均頂点数がより大きなグラフデータに対しても，提案手法が計算時間を削減することが期待できる．

図 7 は各頂点間に存在する辺の割合を $p = 1\%$ に, 図 8 は $p = 30\%$ に固定し，最小支持度 $minsup$ を変化させた場合の多頻度グラフの候補生成に要する計算時間を評価した結果である．この 2 つの場合を比較すると，提案手法が有効な場合は頂点間の辺の存在する確率 $p$ が低いグラフ (疎グラフ) でかつ，最小支持度 $minsup$ が低い時である．

## 4.1 応用例

実際のデータに対する応用例として，PAKDD2000 Workshop(KDD Challenge 2000)[PAKDD] で提供された変異原性のデータを使用した．このデータは 230 個の化合物からなり，化合物を構成する原子の種類は 10 種類，結合の種類は 4 種類である．原子，結合，原子の種類，結合の種類をそれぞれグラフの頂点，辺，頂点ラベル，辺ラベルとして扱う．1 つのグラフデータに含まれる頂点数は，平均で約 17.6 ，最大のもので 28 であり，頂点間に辺が存在する確率は，約 12.4%であった．

また，化学構造データは疎グラフであるため，頂点間に辺が存在しない場合が多い．そこで頂点間の距離が 2

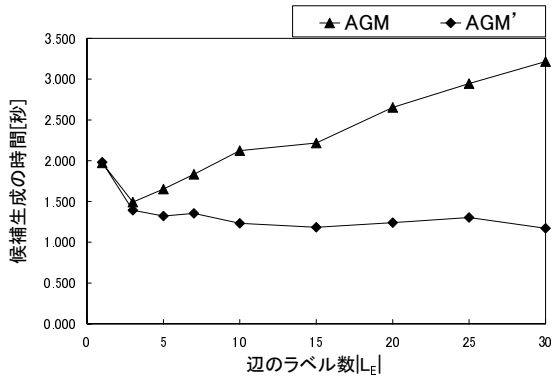


図 4: $|L_E|$ v.s. 多頻度グラフの候補生成の計算時間

表 2: $|L_E|$ v.s. $|\hat{C}_{sum}|$,$|C_{sum}|$,$|F_{sum}|$

| | $|\hat{C}_{sum}|$ | | $|C_{sum}|$ | $|F_{sum}|$ |
|---|---|---|---|---|
| $|L_E|$ | AGM | AGM' | | |
| 1 | 68,950 | 68,950 | 14,532 | 8,595 |
| 3 | 103,932 | 92,280 | 11,854 | 5,357 |
| 5 | 140,838 | 103,217 | 12,389 | 4,591 |
| 7 | 171,824 | 114,644 | 14,039 | 4,261 |
| 10 | 217,965 | 104,657 | 14,661 | 3,882 |
| 15 | 237,200 | 101,940 | 16,875 | 3,522 |
| 20 | 297,696 | 108,151 | 18,092 | 3,448 |
| 25 | 330,096 | 102,529 | 20,608 | 3,202 |
| 30 | 363,816 | 87,164 | 20,588 | 3,165 |

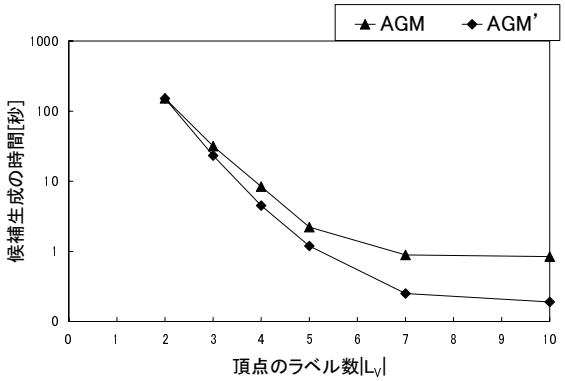


図 5: $|L_V|$ v.s. 多頻度グラフの候補生成の計算時間

表 3: $|L_V|$ v.s. $|\hat{C}_{sum}|$,$|C_{sum}|$,$|F_{sum}|$

| | $|\hat{C}_{sum}|$ | | $|C_{sum}|$ | $|F_{sum}|$ |
|---|---|---|---|---|
| $|L_V|$ | AGM | AGM' | | |
| 2 | 4,331,264 | 4,310,981 | 36,506 | 20,629 |
| 3 | 1,522,880 | 1,079,426 | 28,572 | 9,012 |
| 4 | 658,476 | 262,046 | 33,582 | 4,546 |
| 5 | 237,200 | 101,940 | 16,875 | 3,522 |
| 6 | 160,030 | 20,863 | 6,686 | 2,074 |
| 7 | 104,192 | 17,282 | 5,308 | 2,389 |
| 8 | 99,034 | 7,526 | 4,170 | 1,866 |
| 9 | 114,920 | 6,294 | 4,714 | 2,037 |
| 10 | 95,552 | 7,771 | 5,762 | 2,532 |

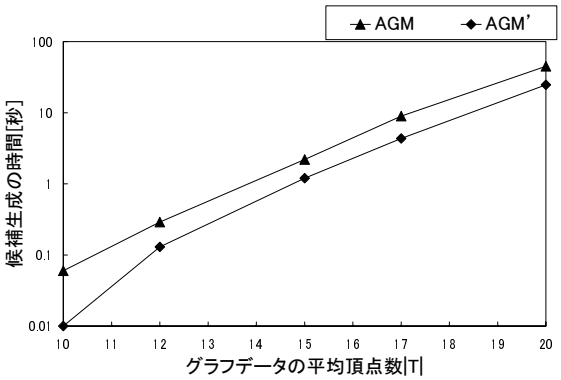


図 6: $|T|$ v.s. 多頻度グラフの候補生成の計算時間

表 4: $|T|$ v.s. $|\hat{C}_{sum}|$,$|C_{sum}|$,$|F_{sum}|$

| | $|\hat{C}_{sum}|$ | | $|C_{sum}|$ | $|F_{sum}|$ |
|---|---|---|---|---|
| $|T|$ | AGM | AGM' | | |
| 10 | 8,640 | 1,146 | 917 | 247 |
| 12 | 43,520 | 13,193 | 5,089 | 827 |
| 15 | 237,200 | 101,940 | 16,875 | 3,522 |
| 17 | 666,320 | 287,525 | 29,814 | 8,353 |
| 20 | 2,323,824 | 1,141,953 | 53,902 | 21,017 |

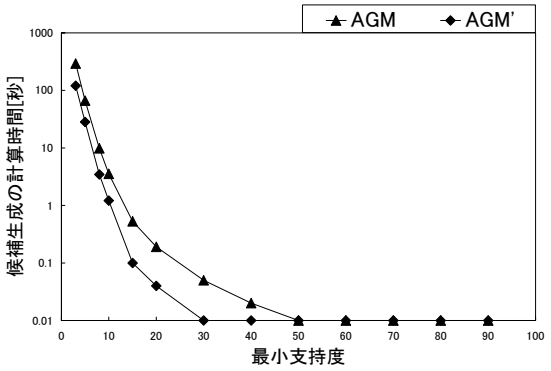


図 7: $p = 1\%$ の minsup v.s. 多頻度グラフの候補生成の計算時間

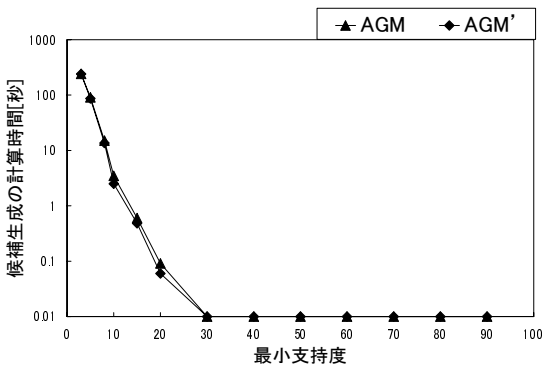


図 8: $p = 30\%$ の minsup v.s. 多頻度グラフの候補生成の計算時間

から 15 の頂点間には仮想的な辺ラベルを付け加えた．仮想的な辺ラベルのつけ方の例を図 9 にしめす．辺が存在しない頂点間に距離が 2，3 であるという仮想的なラベルを張る．すなわち，単結合，芳香族結合，距離 2，距離 3 などのラベルを持つ辺が存在することになる．仮想の辺ラベルをつけることで，頂点間の辺ラベルを特定する条件が厳しくなるため，計算時間を短縮できる．

最小支持度 $minsup$ を変化させた時のシステム全体の計算時間の変化を図 10 に，$|\hat{C}_{sum}|,|C_{sum}|,|F_{sum}|$ の生成数を表 5 に示す．提案手法における $|\hat{C}_{sum}|$ の生成数は AGM の約 50%であり，システム全体の計算時間も AGM の約 70%〜 80%となった．提案手法が化学構造データに対して有効であることが確認できた．

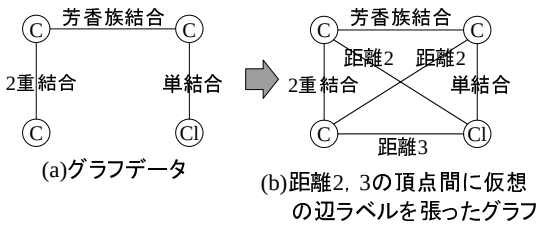


図 9: 仮想の辺ラベル

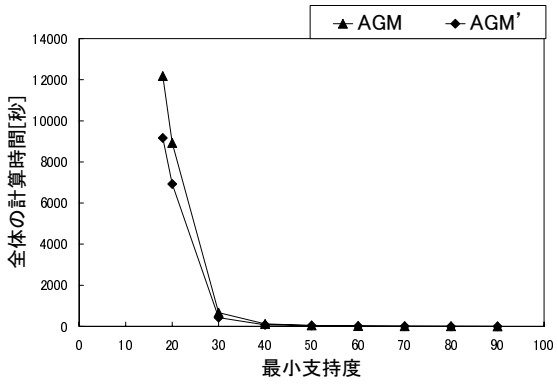


図 10: 化学構造データ (PAKDD) の結果

表 5: 化学構造データ (PAKDD) の $minsup$ v.s. $|\hat{C}_{sum}|,|C_{sum}|,|F_{sum}|$

| | $\|\hat{C}_{sum}\|$ | | $\|C_{sum}\|$ | $\|F_{sum}\|$ |
|---|---|---|---|---|
| $minsup$ | AGM | AGM' | | |
| 20 | 9,860,354 | 4,151,854 | 160,448 | 151,265 |
| 30 | 1,389,109 | 584,952 | 24,253 | 20,582 |
| 40 | 489,763 | 206,259 | 8,882 | 6,761 |
| 50 | 206,473 | 76,155 | 4,728 | 3,157 |
| 60 | 84,303 | 26,723 | 2,335 | 1,329 |
| 70 | 33,478 | 8,941 | 1,292 | 620 |
| 80 | 13,034 | 3,542 | 830 | 355 |
| 90 | 11,381 | 1,959 | 631 | 330 |

# 5 むすび

本研究では AGM アルゴリズムの多頻度グラフの候補生成の条件として，異なる条件を使用することで多頻度グラフの候補生成の計算時間の高速化を行った．さらに提案手法を実装したシステムを構築し，人工データによる性能評価によって提案手法の特徴を分析した．また，提案手法を実データへ適用し，実際に AGM アルゴリズム全体の計算時間も高速化できることを示した．

## 参考文献


[Agrawal 94] Agrawal, R. and Srikant, R.: Fast Algorithms for Mining Association Rules, in Bocca, J. B., Jarke, M., and Zaniolo, C. eds., *Proc. of the 20th Very Large Data Bases Conference*, pp. 487–499, Morgan Kaufmann (1994).

[Dehaspe 98] Dehaspe, L., Toivonen, H., and King, R. D.: Finding frequent substructures in chemical compounds, in Agrawal, R., Stolorz, P., and Piatetsky-Shapiro, G. eds., *Proc. of the 4th International Conference on Knowledge Discovery and Data Mining*, pp. 30–36, AAAI Press. (1998).

[Inokuchi 00] Inokuchi, A., Washio, T., and Motoda, H.: An Apriori-Based Algorithm for Mining Frequent Substructures from Graph Data, in *Proc. of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 13–23 (2000).

[Kuramochi 01] Kuramochi, M. and Karypis, G.: Frequent Subgraph Discovery, in *Proc. of the 1st IEEE International Conference on Data Mining* (2001).

[Motoda 97] Motoda, H. and Yoshida, K.: Machine Learning Techniques to Make Computers Easier to Use, in *Proc. of the 15th International Joint Conference on Artficical Intelligence*, Vol. 2, pp. 1622–1631 (1997).

[PAKDD] http://www. slab. dnj. ynu. ac. jp/ challenge2000/.

[Raedt 01] Raedt, L. D. and Kramer, S.: The Levelwise Version Space Algorithm and its Application to Molecular Fragment Finding, in *Proc. of the 17th IJCAI*, pp. 853–859 (2001).

[Yoshida 95] Yoshida, K. and Motoda, H.: CLIP: Concept Learning from Inference Pattern, *Artificial Intelligence*, Vol. 75, No. 1, pp. 63–92 (1995).

[猪口 01] 猪口 明博, 鷲尾 隆, 元田 浩：Apriori-Based Graph Mining アルゴリズムの効率化, 第 15 回人工知能学会全国大会 (2001).