

大規模疎行列データの不均等な2分割アルゴリズム

荒木 太志^{1,a)} スッパキットパイサーン ウォラポン^{1,b)} 須田 礼仁^{1,c)}

概要: 大規模な疎行列-ベクトル積 (SpMV) のための効率的な行列データの分割について扱う。グラフのアルゴリズムであるマルチレベル法を用い、頂点セパレータを求めることによって行列を二つのパーティションに2次元分割する。演算性能の異なるデバイスや計算機を用いるヘテロな計算環境に行列データを分散配置することを考慮し、任意の比率でデータを分割できるようにする。従って、分割には分割比率とセパレータサイズという二つの評価関数が存在する。従来手法は分割比率とセパレータサイズが同じ評価関数であったが、本研究では異なる評価関数の利用を提案する。実験では行ごとの非ゼロ要素数分布にばらつきがある行列で評価方法によって結果に違いがあらわれた。特に、非ゼロ要素数分布がべき乗則に従うような行列において、分割比率とセパレータサイズの評価にそれぞれ異なる指標を用いた方法では同じ指標を用いた方法に比べてセパレータサイズが最大約90%減少した。

1. はじめに

疎行列-ベクトル積 (SpMV) は線形方程式系の反復解法やグラフアルゴリズム等に現れ、時間的なボトルネックとなる重要な演算である。SpMV に用いられる疎行列はしばしば大規模になり、並列化し演算を高速化することが求められる。その計算を効率的に並列化するためには疎行列のデータをどのように分割して各計算ユニットに分配するかということが重要である。そこでは、各計算ユニットの負荷バランスが適切であること、ユニット同士の通信コストを小さくすることが求められる。

疎行列の分割には、行ごと、あるいは列ごとに分割される1次元分割アルゴリズムや、同じ行内、列内の要素でも異なる領域に分割される2次元分割アルゴリズムが存在する。1次元分割の代表的なライブラリには Metis[1] や Scotch[2] がある。2次元分割ではハイパーグラフを用いた方法が提案されている [3], [4]。ハイパーグラフを用いた2次元分割では1次元分割よりもしばしば通信データ量が少なかったが分割に時間がかかっていた [3]。しかし、近年 Nested Dissection 法 (ND) を用いた2次元分割アルゴリズムが提案された [5]。その手法では1次元分割と同程度の時間で、ハイパーグラフを用いた2次元分割と同程度のクオリティの2次元分割が可能になった。この方法では、

1次元ハイパーグラフ分割で辺セパレータを求めた後に境界の最小頂点被覆を求めることで間接的に頂点セパレータを構成し ND を実装している。それに対して、本研究では辺セパレータからではなく直接的な方法で頂点セパレータを構成する。

ND[6], [7] は従来、疎行列直接法のフィルインを抑える目的で行や列の並べ替えのために用いられてきた。そのため、SpMV のための疎行列分割を目的とはしておらず、従来の直接頂点セパレータを求める方法は最適な分割方法になっていない可能性がある。本研究では特に二つの問題について扱う。

問題の一つ目は分割比率である。フィルインの減少を目的とする ND の場合、データの分割比率は単純な 1:1 で問題なかった。このとき分割された領域の数や大きさの柔軟性は低くなる。しかし、SpMV の並列計算を想定すると並列数が3のような 1:1 分割の繰り返しでは構成しにくいケースが存在する。その場合は1回目の分割で 1:2 に分割した後、分割比率が2だった領域をさらに 1:1 に分割して 1:1:1 に3分割するという方法がシンプルな方法として考えられる。そのため、1:1 以外にも分割できることが望ましい。また、近年は CPU と GPU を両方用いた SpMV の並列化のようなヘテロな計算も行われている [8], [9]。このような場合にも 1:1 ではない不均等な分割が必要になる。不均等な分割はネットワークの論文に類似したものが存在する [10]。

二つ目の問題は重みである。ND を SpMV のための行列分割に転用するため、SpMV に適した重みを設定すべきである。SpMV のための分割では、負荷バランスを適切にす

¹ 東京大学大学院情報理工学系研究科
Graduate School of Information Science and Technology,
The University of Tokyo
a) araki-takayuki@g.ecc.u-tokyo.ac.jp
b) vorapong@is.s.u-tokyo.ac.jp
c) reiji@is.s.u-tokyo.ac.jp

る、通信コストを小さくする、という二つの目的関数がある。そのため、それぞれの目的に対して異なる重みを用いるほうが良い結果が出る可能性がある。

そこで本研究では、これらの分割比率と重みの問題を考慮した SpMV のための 2 次元分割アルゴリズムを提案する。アルゴリズムはマルチレベル法 [11] をベースとした。ヘテロ環境を考慮して任意の比率で分割を行えるよう実装した。そして、2 種類の重みと分割比率・セパレータサイズの 2 項目に対して、3 種類の評価方法の組み合わせを実装した。今回は単純化のために 2 分割のアルゴリズムを実装した。ND ではこの 2 分割を繰り返すことで 4 分割、8 分割のような分割もできるが、今回は対応していない。

複数の行列に対して分割を行い分割比率とセパレータサイズを評価した結果、行あたりの非ゼロ数にばらつきがある行列では評価方法によって性能に差が出た。特に、行あたりの非ゼロ数がべき乗則に従う行列で条件によってはセパレータサイズが最大 90% 減少した。

2 章では疎行列の分割について説明し、3 章では提案手法を示す。4 章では実験の条件と評価結果を示し、5 章にまとめを示す。

2. 疎行列の分割

本章では疎行列-ベクトル積 (SpMV) と疎行列の分割、グラフとの関係性について説明する。そして、既存の分割手法であるマルチレベル法を紹介する。

2.1 疎行列-ベクトル積

SpMV は次のような行列とベクトルの演算である。

$$\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \quad (1)$$

一般には、 \mathbf{A} は $m \times n$ 次元の疎行列、 \mathbf{x} は n 次元、 \mathbf{y} は m 次元である。ただし、今回は行列に対称行列を用いる。

SpMV 全体の計算量は \mathbf{A} のゼロではない各要素が 1 回ずつ積和されるため $O(nnz)$ となる。 nnz は行列の非ゼロ要素数を表す。

2.2 分割の種類

疎行列の分割で代表的なものは 1 次元分割である。1 次元分割では、行ごと、あるいは列ごとに行列を分割する。図 1 は 1 次元分割の例を示す。行列が薄青色と薄黄色に分割され、並列計算の際にはそれぞれ一つの計算ユニットに割り当てられる。 \mathbf{y} の青の部分の計算に薄青色の部分が用いられ、黄色の部分に薄黄色の部分が用いられる。この図の場合、ベクトル \mathbf{x} の緑色の領域はベクトル \mathbf{y} の青色の領域の計算にも黄色の領域の計算にも用いられるため、分散メモリ環境で並列計算する際には事前のコピー、転送が必要になる。よってこのベクトル \mathbf{x} の緑色の領域のサイズは通信コストに対応すると考えることができる。1 次元分割

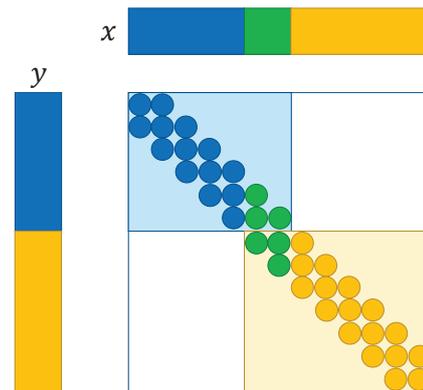


図 1: 行列の 1 次元分割

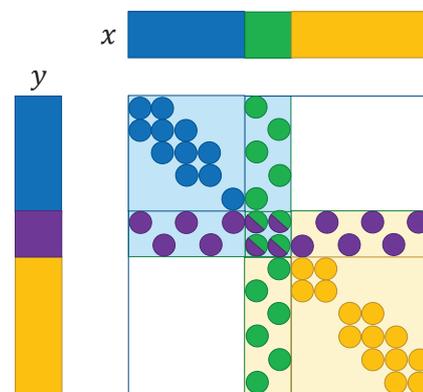


図 2: 行列の 2 次元分割

にはマルチレベル法 [12] やスペクトラル法 [13] のような方法が存在している。

それに対して、本研究では 2 次元分割を考える。2 次元分割では、一つの行や列に違う領域に所属する要素が存在するような分割をする。様々な分割方法があるが、図 2 では本研究で仮定する 2 次元分割の例を示す。ここでは、行列が薄青色の部分と薄黄色の部分に分割されている。SpMV では、ベクトル \mathbf{x} の緑色の部分は行列の緑色の要素と掛け算されるが、緑色の要素は薄青色と薄黄色にまたがって存在する。そのため、薄青色のデータを持つ計算ユニットと薄黄色のユニット間でベクトル \mathbf{x} の緑色の部分のデータ転送が必要になる。また、ベクトル \mathbf{y} の紫色の部分は行列の紫色の要素の畳み込みだが、紫色の要素は薄青色と薄黄色にまたがって存在するためベクトル \mathbf{y} の紫色の部分に対応するサイズ of データ転送が必要になる。そのため、ベクトル \mathbf{x} の緑色の部分とベクトル \mathbf{y} の紫色の部分のサイズが通信コストに対応すると考えることができる。2 次元分割にはハイパーグラフを用いる方法 [3], [4] や ND を用いる方法 [5] が存在している。ND のためにマルチレベル法 [11] を用いることがある。

2.3 疎行列分割とグラフの関係

疎行列の分割方法に用いられるマルチレベル法はグラフのアルゴリズムである。疎行列とグラフには密接な関係がある。対称な疎行列 A の対角成分を取り除き、非ゼロ要素を 1 に置き換えた行列を考え、無向グラフ $G(V, E)$ の隣接行列とみなす。すると、各行または列が頂点对応し、各非ゼロ要素が辺に対応する。

このグラフ G に対するグラフ分割を行い、頂点集合 V を二つの素集合 X と Y に分ける。例えば、 X に所属する頂点对応する行の集合と Y に所属する頂点对応する行の集合に行列を分割することで 1 次元分割をすることができる。 X に所属する頂点と Y に所属する頂点を結ぶ辺の集合を辺セパレータと呼ぶ。この辺セパレータは $SpMV$ において通信の量に対応しており、図 1 の緑色の領域の大きさに関係する。

2 次元分割の場合は 1 次元よりやや複雑になる。一つの方法は、グラフ G の頂点を 3 つの素集合 X, Y, S に分割する。ここで、 X に所属する頂点と Y に所属する頂点は辺で結ばれず、 S によって区切られる。この S は頂点セパレータと呼ばれ、図 2 の緑色と紫色の部分に対応している。すなわち、 S の大きさが転送されるデータの量に対応している。よって、この頂点セパレータ S を小さくすることで転送されるデータ量が少なくなることが期待される。ここで、行列 A を、 X に所属する頂点同士を結ぶ辺に対応する非ゼロ要素を含む部分行列 A_X (図 2 の青い丸が含まれる部分) と Y に所属する頂点同士を結ぶ辺に対応する要素を含む部分行列 A_Y (図 2 の黄色の丸が含まれる部分) と残りの要素を含む部分行列 A_S に行列を分ける。そして、 A_S に含まれる非ゼロ要素それぞれを A_X, A_Y のどちらかへ加えることで行列を 2 次元分割をすることができる。

2.4 マルチレベル法

提案手法のベースとなっているマルチレベル法について説明する。グラフのマルチレベル法には辺セパレータを求めるもの [12] や頂点セパレータ [11] を求めるものがあるが、ここでは 2 次元分割に用いられる頂点セパレータを求めるマルチレベル法を示す。

マルチレベル法は 3 つのフェーズから構成される。Coarsening, Initial partitioning, Uncoarsening である。頂点セパレータを求める問題は組み合わせ最適化問題であり NP 困難である [14]。そこでマルチレベル法では縮約によりグラフサイズを小さくし、小さいグラフで初期分割を行い、それを元のサイズに戻しながら解を改良するというステップを踏む。

図 3 はマルチレベル法の 3 フェーズの図である。

2.4.1 Coarsening

このフェーズでは元のグラフを階層的に小さくしていく。そのために頂点の縮約を行う。

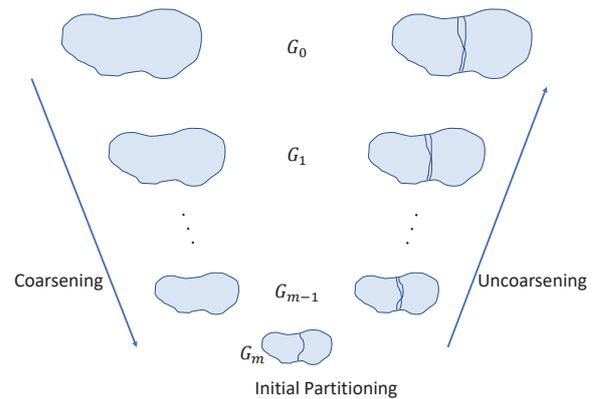


図 3: マルチレベル法 [12] をもとに作成

まずグラフに対して極大マッチングを取る。極大マッチングにはいくつかの方法が存在するが、Random Matching (RM) や Heavy-Edge Matching (HEM) 等がある [12]。今回、通常のグラフには RM を用いてマッチングを行った。

そして、マッチングに含まれる辺を縮約し、その縮約情報を記録し、生成されたグラフの辺や頂点重みを適切に設定することで復元可能な粗くなったグラフが得られる [12]。

これらのマッチングと縮約を繰り返すことである程度までグラフを小さくする。

RM ではマッチングが上手いかない種類のグラフがある。頂点次数がべき乗則に従うグラフでは従来のマッチング方法で縮約が途中から上手いかなくなることが実験的に示され、クラスタリングに基づく方法が提案された [15]。今回そのようなグラフにはクラスタリングに基づく方法のうち論文 [15] で FC と呼ばれているスキームを用いて縮約を行った。

2.4.2 Initial partitioning

粗く小さくなったグラフ上で分割を行う。セパレータを求めて分割する方法には Kernighan-Lin アルゴリズム (KL) [16] や Fiduccia と Mattheyses による高速版 (FM) [17] がある。KL や FM はもともとは辺セパレータ用のアルゴリズムだったが、頂点セパレータ用に拡張された [18], [19]。

アルゴリズム 1 は頂点セパレータ用の KL/FM アルゴリズムの概要である。KL/FM は反復改良アルゴリズムであり、局所最適化を繰り返すことで分割比率の制約を満たしながらセパレータサイズを減少させようとする。ただし、局所解に陥らないように一時的に解が悪化することを許容している。

$w_s(S)$ は頂点集合 S 、セパレータに含まれる頂点重みの和を表している。セパレータサイズが小さいほど通信コストが減るため、この和が小さいほうが良い解であると考えられる。

内側の while ループでは頂点の移動を行っているが、そ

アルゴリズム 1 KL アルゴリズム ([19] をもとに作成)

Input: グラフ G , パーティション X_0, Y_0 , セパラータ S_0

Output: パーティション X, Y , セパラータ S

```

1: while より良い解が見つからない do
2:    $X, Y, S \leftarrow X_0, Y_0, S_0$ 
3:   while 終了条件に到達していない do
4:     移動させる頂点  $v$  を  $S_0$  から選び  $X_0$  か  $Y_0$  に移動
5:     if  $X_0$  と  $Y_0$  を結ぶ辺がある then
6:        $T_0 \leftarrow X_0, Y_0$  のうち  $v$  を移動させなかった領域
7:        $T_0$  から適当な頂点を  $S_0$  に移動して調整
8:     end if
9:     if  $w_s(S) > w_s(S_0)$  then
10:       $X, Y, S \leftarrow X_0, Y_0, S_0$ 
11:     end if
12:   end while
13:    $X_0, Y_0, S_0 \leftarrow X, Y, S$ 
14: end while

```

の際には分割比率が許容範囲内に収まっていること、そのループ内で一度 S_0 から動かした頂点は再び動かさない等の制約がかかっている。

S_0 が頂点セパラータであるためには、 X_0 と Y_0 を結ぶ辺があってはならない。しかし、頂点 v を S_0 から移動させることで一時的に X_0 と Y_0 を結ぶ辺が生じることがある。これを解消するために、アルゴリズム 1 の 5 から 8 行目では v を移動させなかった領域から頂点を S_0 に戻すことで調整を行っている。 v を X_0 に動かした場合は Y_0 から頂点を戻し、 Y_0 に動かした場合は X_0 から頂点を戻す。

初期分割では最初に全頂点をセパラータ S に入れた状態からスタートし改良を繰り返して分割を求める。

2.4.3 Uncoarsening

前のフェーズでは粗く小さなグラフにおける解を求めたが、元問題の解を求めるには小さくしたグラフを復元して大きな元のグラフ上の解を構成する必要がある。Coarsening フェーズで記録した縮約の情報を用いることで小さなグラフの解から大きなグラフの解を構成することができる。大きなグラフ上の頂点 u, v が小さなグラフの頂点 w に縮約されたとする。 w が小さいグラフ G_{i+1} 上で集合 X_{i+1} に含まれるとき、 u, v を大きなグラフ G_i 上で集合 X_i に射影する。 Y_i, S_i でも同様のことを行う。すると S_i は G_i 上で頂点セパラータとなり、解が構成される。

この単純に射影した解よりもより良い解を構成するために、射影後に KL/FM によって反復改良を行う。

この射影と反復改良を繰り返すことで元のグラフにおける解が求められる。

3. 提案手法

本章では疎行列の不均等な 2 次元 2 分割についての提案手法を示す。大まかな部分はセクション 2.4 の頂点セパラータを求めるマルチレベル法と同様である。既存のマルチレベル法の実装と異なる点が二つある。一つは分割比率

を任意に変えられる点であり、もう一つは分割比率とセパラータサイズの評価指標が別である点である。

3.1 不均等な分割

3 プロセスで SpMV を並列化するケースやヘテロな環境を考えると 1:1 以外の比率にも分割できると良い。そのため、提案手法では任意の比率に疎行列を分割できるようにする。

今回、次のような分割比率の評価式を考える。

$$r = \frac{w_b(X) + w_b(S)}{w_b(Y) + w_b(S)} \quad (2)$$

w_b は分割比率を評価するために用いる重みである。 r は X が含まれる領域側の重みと Y が含まれる側の重みの比を表す。

ユーザーは、2 分割後の X を含む側の重み比率の目標値であるパラメータ P_X を設定する。 P_X は、次のような比率の式に近くなってほしい値を設定する。

$$\frac{w_b(X) + w_b(S)}{w_b(X) + w_b(Y) + 2w_b(S)} \quad (3)$$

1:1 分割ならば $P_X = 0.5$ 、3:7 分割なら $P_X = 0.3$ のような値を設定する。理想的な分割の場合、

$$r = \frac{P_X}{1 - P_X} \quad (4)$$

となる。式 (2) と式 (3) は S が十分小さく無視できることを仮定している。

マルチレベル法では 1:1 分割する際に厳密に 1:1 に分割を行うと解が非常に悪くなる可能性がある。そのため、実際には厳密な比率に分割せず若干比率にずれを許容する。ユーザーは tol という分割比率の許容範囲のパラメータを設定する。そして、 r が次の範囲に入るように初期分割、解の改良をすることで解の分割比率が一定範囲に収まるよう保証する。

$$\frac{P_X - tol}{1 - P_X} \leq r \leq \frac{P_X + tol}{1 - P_X} \quad (5)$$

例えば、 P_X に 0.3、 tol に 0.01 を設定すると、分割比率を 0.29 : 0.71 から 0.31 : 0.69 までの間に収める。この式はアルゴリズム 1 の 3 行目の終了条件判定の中で用いられる。

3.2 評価指標

頂点セパラータを求めるマルチレベル法では分割比率とセパラータサイズの二つの重みがある。

分割比率の重みは式 (2) の w_b である。セクション 3.1 で述べたように、分割比率が設定した範囲から外れないように解を求めするために用いられる。一方、セパラータサイズの重みはアルゴリズム 1 の 9 行目の重み w_s であり、他にもアルゴリズム 1 の 4 行目で移動させる頂点を選ぶ際の基準として用いられる。

マルチレベル法の実装には Metis[1] の ND オーダリング等があるが、それらは分割比率とセパレータサイズの評価に同一の頂点重みを用いる。しかし、分割比率とセパレータサイズに異なった指標を用いることもできる。SpMV の計算量は非ゼロ要素数に比例するため、分割比率の評価は行あたりの非ゼロ要素数 (nnz) を頂点重みとして行うことが自然に思われる。また、セパレータサイズについてはセクション 2.2 で述べたように、送受信されるベクトルの要素数に対応しているため、行数 (nrow) を頂点重みとして評価することが自然に思われる。nnz を用いる場合の重みを w_{nnz} 、nrow を用いる場合の重みを w_{nrow} と表すことにする。

そこで本研究では、(分割比率, セパレータサイズ) の重みの組に対して、三つのパターンによる評価を考える。

1. $w_b = w_s = w_{nrow}$
2. $w_b = w_s = w_{nnz}$
3. $w_b = w_{nnz}, w_s = w_{nrow}$

前二つは従来と同じ評価方法であり、Metis[1] の ND オーダリングでは頂点重みを適切に設定することで 1:1 分割は可能である。一方、最後の一つは分割比率とセパレータサイズで評価方法を変えた新しい評価方法となっている。

3.2.1 評価手法 1: nrow-only

この手法では分割比率もセパレータサイズも行数によって評価する。すなわち、

$$w_b = w_s = w_{nrow} \quad (6)$$

である。マルチレベル法の頂点重みでは、各頂点に対して 1 を重みとして設定する。セパレータサイズは行数に対応しているため、適切に評価していると考えられる。一方、分割比率は非ゼロ要素数で評価するほうが適していると考えられるため、この手法では分割のバランスが崩れてしまう可能性がある。

3.2.2 評価手法 2: nnz-only

この手法では分割比率もセパレータサイズも非ゼロ要素数で評価する。すなわち、

$$w_b = w_s = w_{nnz} \quad (7)$$

である。マルチレベル法の頂点重みでは、各頂点に対応する行に含まれる非ゼロ要素数を設定する。分割比率の評価に非ゼロ要素数を用いるため分割比率は上手く評価されるが、セパレータサイズは適切に評価できない可能性がある。

3.2.3 評価手法 3: nnz-nrow

この手法では分割比率の評価に非ゼロ要素数、セパレータサイズの評価に行数を用いる。すなわち、

$$w_b = w_{nnz}, w_s = w_{nrow} \quad (8)$$

である。そのため、分割比率もセパレータサイズも適切に評価できると考えられる。

この新しい方法の実装は、従来のマルチレベル法の一部を変更することで実現できる。まず、従来 1 種類の頂点重みをマルチレベル法の各階層に伝播していたところを 2 種類の重み両方を伝播させていく。そして、分割比率を評価する部分では w_{nnz} を用い、セパレータサイズを評価する部分では w_{nrow} を用いるように従来の実装を変更すればよい。

4. 実験結果

提案手法による分割比率とセパレータサイズを評価するために実験を行った。

4.1 入力行列

分割する行列には SuiteSparse Matrix Collection[20] から得た行列と Barabási-Albert モデル (BA モデル) [21] から生成した行列を用いた。表 1 は入力行列の一覧である。nrow は行数、nnz は非ゼロ要素数、平均は行あたりの非ゼロ要素数平均、標準偏差は行ごとの非ゼロ要素数のばらつきを表す。べき乗則が yes のものは、行ごとの非ゼロ要素数のばらつきがべき乗則に従う。

4.1.1 Barabási-Albert モデル

BA モデルは次数分布がべき乗則に従うグラフを生成するモデルである。このグラフは、頂点数 x と次数分布 $f(x)$ の関係が適当な定数 a, k に対して

$$f(x) = ax^{-k} \quad (9)$$

に近くなるようなグラフである。図 4 はこのモデルで生成したグラフの次数分布に対数をとって表したものである。BA モデルはグラフを生成する際に初期の頂点数 m_0 、各ステップに追加する辺の数 m を設定する。また、グラフが連結であることを保証するために初期の m_0 個の頂点をランダムに辺で結び、木にしている。

実験ではこのモデルに従って生成されたグラフの隣接行列 BA20000_10_3, BA20000_10_6, BA20000_10_10 を入力とした。BA xx_yy_zz は、 xx が頂点数、 yy が m_0 、 zz が m の値を表す。

4.2 実験条件

4.2.1 変動させた条件

今回は分割する行列の種類、評価方法、分割比率を変えて実験を行った。各行列に対して、分割比率/セパレータサイズを nrow-only, nnz-only, nnz-nrow の 3 種類で評価し、分割比率は 1:9, 2:8, 3:7, 4:6, 5:5 の 5 種類で実験を行った。すなわち各行列に対して 15 種類の異なる条件下で実験を行った。

4.2.2 固定した条件

Coarsening は頂点数が 1000 以下になるまで行った。Coarsening におけるマッチングの方法は、行あたりの非

表 1: 入力行列

名前	nrow	nnz	平均	標準偏差	べき乗則	種類
bundle1	10,581	770,811	72.8	189.8	no	Computer Graphics/Vision Problem
crankseg_1	52,804	10,614,210	201.0	88.6	no	structural Problem
ecology2	999,999	4,995,991	5.0	0.06	no	2D/3D Problem
G3_circuit	1,585,478	7,660,826	4.8	0.64	no	Circuit Simulation Problem
ldoor	952,203	42,493,817	44.6	11.9	no	structural Problem
nd12k	36,000	14,220,946	395.0	83.2	no	2D/3D Problem
parabolic_fem	525,825	3,674,625	7.0	0.15	no	Computational Fluid Dynamics Problem
com-Amazon	334,863	1,851,744	5.5	5.6	yes	Undirected Graph with communities
com-DBLP	317,080	2,099,732	6.6	9.6	yes	Undirected Graph with communities
coAuthorsCiteseer	227,320	1,628,268	7.2	10.5	yes	Undirected Graph(Citation Networks)
BA20000_10_3	20,000	119,958	6.0	19.0	yes	BA モデル
BA20000_10_6	20,000	239,898	12.0	22.8	yes	BA モデル
BA20000_10_10	20,000	399,818	20.0	30.6	yes	BA モデル

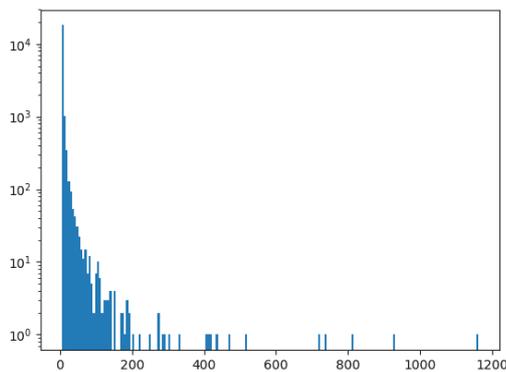


図 4: BA20000_10_3 の次数分布

ゼロ数がべき乗則に従う行列に対してクラスタリングに基づく方法 [15] を用い、それ以外の行列に対しては Random Matching [12] を行った。分割比率は 0.5% までずれを許容した。マッチングでは乱数が影響するため、各条件に対して 10 回分割を行った。

4.3 分割比率の評価

SpMV の計算量は $O(nnz)$ であり、分割比率は行数ではなく非ゼロ要素数で評価を行うのが自然に思われる。nrow-only ではバランスを $w_b = w_{nrow}$ と行数で評価しているため、 w_{nrow} でのバランスは 0.5% 以内に収まっている。しかし、非ゼロ要素数に基づく w_{nnz} のバランスが崩れている可能性がある。そこで、このセクションでは nrow-only における非ゼロ要素数のバランスのずれを評価した。

表 2 は 10 回の試行のうち最も分割比率のずれが大きいときのずれを百分率で表している。縦には行列の種類が並び、横には分割比率が並んでいる。"ok" のセルは分割比率のずれが 0.5% の許容範囲内に収まっている条件である。数字が書いてあるセルは分割比率が許容範囲を超えていて、

表 2: 分割比率のずれ (%)

名前	1:9	2:8	3:7	4:6	5:5
bundle1	19.18	15.00	7.40	4.55	2.91
crankseg_1	ok	-0.92	-1.62	-1.04	ok
ecology2	ok	ok	ok	ok	ok
G3_circuit	ok	ok	ok	ok	ok
ldoor	ok	ok	ok	ok	ok
nd12k	-0.89	-0.84	-0.54	ok	ok
parabolic_fem	ok	ok	ok	ok	ok
com-Amazon	ok	ok	-1.10	-1.53	-1.51
com-DBLP	0.72	-0.57	-1.42	-2.86	-2.52
coAuthorsCiteseer	-1.58	-2.61	-3.72	-4.79	-5.16
BA20000_10_3	12.70	10.68	7.69	4.24	ok
BA20000_10_6	9.45	9.44	6.86	3.65	ok
BA20000_10_10	8.96	8.63	6.46	3.56	ok

※ ok はバランスのずれが許容範囲に収まっていることを表す。

次のような式に基づいて計算される非ゼロ要素数による分割比率のずれを示す。

$$\frac{w_{nnz}(X) + w_{nnz}(S)}{w_{nnz}(X) + w_{nnz}(Y) + 2w_{nnz}(S)} - P_X \quad (10)$$

1:9 分割では $P_X = 0.1$ 、2:8 分割では $P_X = 0.2$ のように P_X を設定した。nrow-only において P_X は w_{nrow} について設定するためずれが生じることがある。

ecology2, G3_circuit, ldoor, parabolic_fem は常に許容範囲内に収まっていた。一方で、その他の行列では許容範囲を超える条件が存在した。行あたりの非ゼロ要素数がべき乗則に従わない行列で許容範囲を超えている行列は、行ごとの非ゼロ要素数の標準偏差が大きく、ばらつきが大きかった。このことから、行ごとの非ゼロ要素数のばらつきが大きい行列の分割に nrow-only は向かないと考えられる。

べき乗則に従う行列では BA モデルの行列とそうでない行列で異なる傾向がみられた。BA モデルの行列では分割比率が 1:9 のように偏っているほど分割比率のずれが大きかった。一方、BA モデルではない行列では分割比率が 5:5

に近いほど分割比率のずれが大きかった。このことから、BA モデルのグラフはコミュニティのグラフと異なる特徴を持っていると考えられる。いずれにしても、べき乗則に従う行列に nrow-only は向かないと考えられる。

4.4 セパレータサイズの評価

各手法によるセパレータサイズの比較を行った。セパレータサイズは頂点数、非ゼロ要素数どちらでも評価できるが、頂点数が通信コストに対応しているため頂点数で評価した。図 5 は各条件下におけるセパレータサイズの平均を図示している。(a) から (e) までは行あたりの非ゼロ要素数がべき乗則に従わない行列で、(f) から (j) まではべき乗則に従う行列である。(k) と (l) は coAuthorsCiteseer と BA20000_10.3 のセパレータサイズについて nrow-only を基準に比を取ったものを表す。各グラフのエラーバーは各条件で 10 回計測したセパレータサイズの標準偏差を表す。

行あたりの非ゼロ要素数がべき乗則に従わない行列では、bundle1 で評価方法により大きな差が出た。分割比率とセパレータサイズの評価をどちらも頂点数で行う nrow-only は他の方法に比べて頂点数を大幅に減少させた。ただし、表 2 で示したように bundle1 の nrow-only は分割のバランスが取れていないため、nrow-only が適しているとは言い難い。また、nd12k では 2:8, 3:7, 4:6 分割で nnz-only のときに他の手法に比べてセパレータサイズが小さくなった。crankseg_1 では 3:7 分割のときに nrow-only が良い結果を出した。しかし、それ以外の条件ではセパレータサイズに統計的に有意な差は出なかった。bundle1, nd12k, crankseg_1 は行あたりの非ゼロ要素数のばらつきが大きい行列であるため、手法によって差が出たのだと考えられる。というのも、nrow-only, nnz-only, nnz-nrow は行あたりの非ゼロ要素数が全ての行で等しいとき同じ振る舞いをし、行あたりの非ゼロ要素数のばらつきが小さいとあまり差が出ないと考えられるからである。

行あたりの非ゼロ要素数がべき乗則に従う行列では、手法ごとに大きく差が出るケースが多かった。差が出ているケースでは nnz-nrow が優れた結果を出した。特に、図 5(l) のように人工的な行列の BA20000_10.3 の 1:9 分割では、nnz-nrow は nrow-only に比べてセパレータサイズが約 90%減少した。SuiteSparse Matrix Collection の行列では coAuthorsCiteseer のケースで手法によって最もセパレータサイズの大きさが変化した。図 5(k) のように 2:8 分割のとき nnz-nrow では nrow-only と比べて約 52%セパレータサイズが減少した。他の行列でも、1:9 や 2:8 のような分割比率の偏りが大きいときに nnz-nrow が他の手法に比べてセパレータサイズを大きく減らしていた。一方、com-Amazon の 3:7, 4:6, 5:5 分割や BA20000_10.6, BA20000_10.10 の 5:5 分割では nrow-only と nnz-nrow に大きな差は出なかった。

行あたりの非ゼロ要素数がべき乗則に従う行列において nnz-nrow が他の手法よりも優れていたのは、セパレータに含まれる頂点の平均非ゼロ要素数が大きいことと関係があるかもしれない。次のような式でセパレータ内の平均非ゼロ要素数を評価した。

$$\frac{\sum_i w_{nnz}(S_{1i})}{\sum_i w_{nrow}(S_{1i})} / \frac{\sum_i w_{nnz}(S_{2i})}{\sum_i w_{nrow}(S_{2i})} \quad (11)$$

ここで、 S_{1i} は nnz-nrow で i 番目に得られたセパレータ、 S_{2i} は nrow-only で i 番目に得られたセパレータを表す。 \sum_i で重みの和を取っているのは各条件で 10 回実験を行って、その平均を考えるためである。/ の左側は nnz-nrow で得られたセパレータのセパレータ内頂点の平均非ゼロ要素数を表し、右側は nrow-only の平均非ゼロ要素数を表す。よって、式 (11) の値が大きいほど nnz-nrow によるセパレータ内頂点の平均非ゼロ要素数が相対的に大きいことを表す。値が 1 に近いときは二つの手法による違いがあまりないことを表す。表 3 にその値をまとめた。行あたりの非ゼロ要素数がべき乗則に従う行列では値が大きくなり、nrow-only が nnz-nrow よりもセパレータサイズの小さくなっていた bundle1 では値が小さくなっていた。この結果からセパレータ内頂点の平均非ゼロ要素数とセパレータサイズには関係があるかもしれない。

表 3: 各条件における式 (11) の値

名前	1:9	2:8	3:7	4:6	5:5
bundle1	0.04	0.21	0.34	0.36	0.58
crankseg_1	1.00	1.01	1.07	1.00	1.00
ecology2	1.00	1.00	1.00	1.00	1.00
G3_circuit	1.03	1.00	1.00	1.02	1.00
ldoor	1.00	1.00	1.00	1.00	1.00
nd12k	1.00	1.00	1.00	1.00	1.00
parabolic_fem	1.00	1.00	1.00	1.00	1.00
com-Amazon	1.12	1.10	1.11	1.08	1.06
com-DBLP	1.31	1.29	1.21	1.17	1.07
coAuthorsCiteseer	1.61	1.27	1.09	1.03	1.09
BA20000_10.3	4.05	1.81	1.27	1.14	1.14
BA20000_10.6	2.72	1.55	1.25	1.07	1.02
BA20000_10.10	2.26	1.40	1.22	1.07	1.00

表 4 は次の式が示すような、セパレータに含まれる行数の全体の行数に占める割合を示す。

$$\frac{w_{nrow}(S)}{w_{nrow}(X \cup Y \cup S)} \times 100(\%) \quad (12)$$

nd12k やべき乗則に従う行列では割合が多いものが存在する。このような行列では、式 (2) では計算量をうまく評価できない可能性がある。式 (2) はセパレータサイズが小さいときには $w_b(S)$ が無視できるが、そうでないときは $w_b(S)$ の影響を受けてバランスが崩れるかもしれない。そのため、分割比率の評価方法を再検討する必要があるかもしれない。

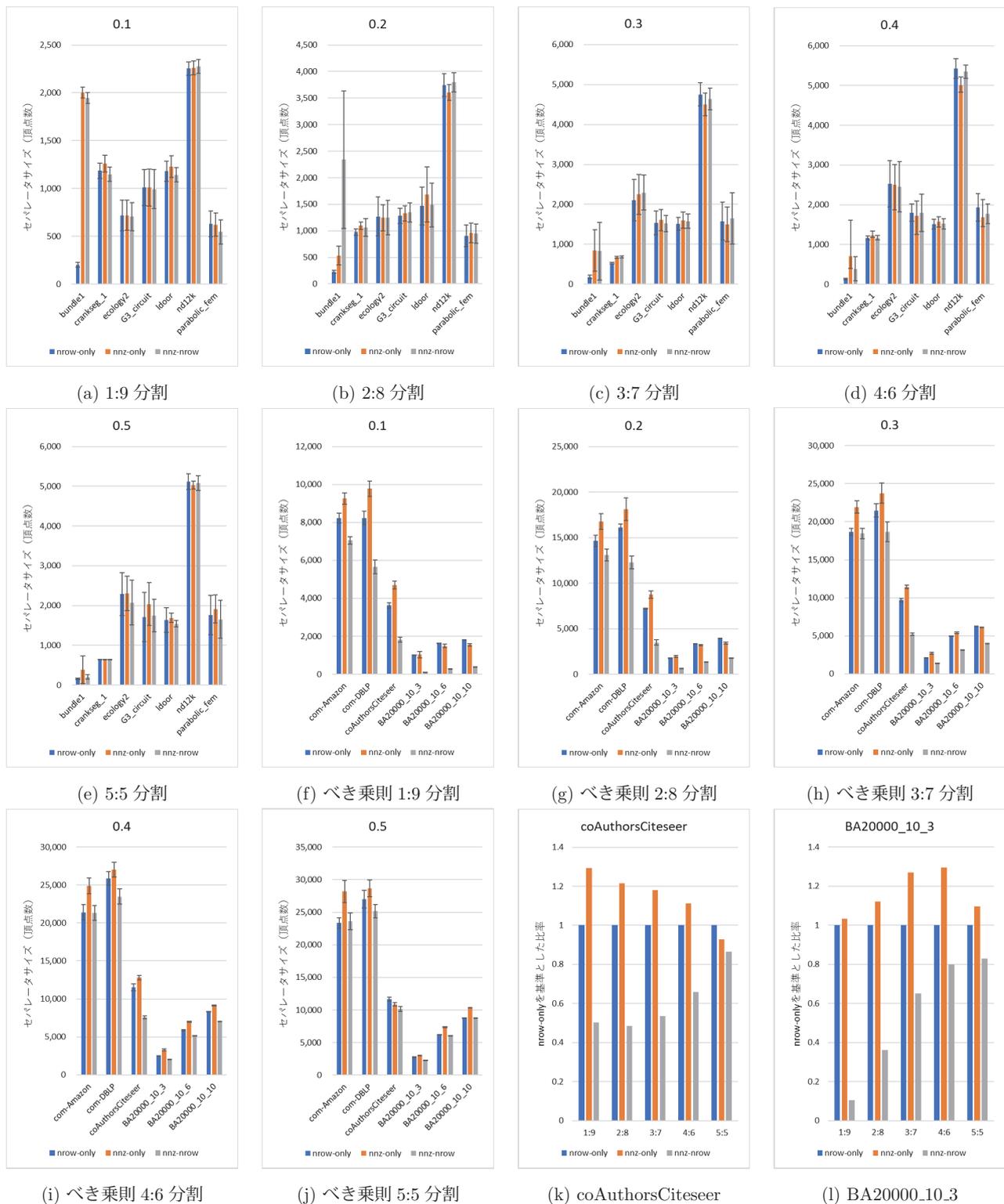


図 5: セパレータサイズの評価

5. まとめ

本研究では、大規模な SpMV のための行列データ 2 次元 2 分割を扱った。分割比率とセパレータサイズの評価を、nrow-only, nnz-only, nnz-nrow という 3 種類の方法を用

いて行い比較した。また、ヘテロな環境で SpMV を計算することを考慮し、分割比率を 1:1 だけでなく任意に変更できるようにし、5つの分割比率で実験を行った。

行ごとの非ゼロ要素数にばらつきがある行列では 3 種類の方法でセパレータサイズに違いがあらわれた。特に、行

表 4: セパレータに含まれる行数の割合

名前	割合 (%)
bundle1	1.92
crankseg_1	1.22
ecology2	0.21
G3_circuit	0.11
ldoor	0.16
nd12k	14.10
parabolic_fem	0.31
com-Amazon	7.05
com-DBLP	7.94
coAuthorsCiteseer	4.46
BA20000_10_3	11.36
BA20000_10_6	30.29
BA20000_10_10	43.75

ごとの非ゼロ要素数がべき乗則に従う行列では nnz-nrow によるセパレータサイズが他の方法に比べて減少した。分割比率が 1.9 のようなかなり不均等な場合にその減少幅は大きく、最大 90%他の方法に比べてセパレータサイズが小さくなった。

今回は行列を分割しただけであり、実際に分割した行列で並列に SpMV を計算し、性能を確かめて、提案手法の有効性を確認することは今後の課題である。

実験では、全体の行数に対してセパレータに含まれる行数の割合が大きい行列があった。そのケースでは今回用いた分割比率の評価式ではうまくいかない可能性があるため、評価式を見直す必要がある。

今回は 2 分割のみを扱っていたが、実際の並列計算では 3 分割以上されることもある。よって、任意の分割数に対応できる実装をし、評価する必要がある。そして、今回は実装した方法同士の比較だけを行ったため、ハイパーグラフを用いる ND[5] のような他の方法との比較も重要である。

参考文献

[1] Karypis, G.: Family of Graph and Hypergraph Partitioning Software, the University of Minnesota (online), available from <http://glaros.dtc.umn.edu/gkhome/views/metis> (accessed 2021-5-25).

[2] Pellegrini, F.: Software package and libraries for sequential and parallel graph partitioning, static mapping and clustering, sequential mesh and hypergraph partitioning, and sequential and parallel sparse matrix block ordering, LABORATOIRE BORDEAIS DE RECHERCHE EN INFORMATIQUE (online), available from <https://www.labri.fr/perso/pelegrin/scotch/> (accessed 2021-5-25).

[3] Çatalyürek, Ü. V. and Aykanat, C.: A Fine-Grain Hypergraph Model for 2D Decomposition of Sparse Matrices., *IPDPS*, Vol. 1, Citeseer, p. 118 (2001).

[4] Vastenhouw, B. and Bisseling, R. H.: A two-dimensional data distribution method for parallel sparse matrix-vector multiplication, *SIAM review*, Vol. 47, No. 1, pp.

67–95 (2005).

[5] Boman, E. G. and Wolf, M. M.: A nested dissection partitioning method for parallel sparse matrix-vector multiplication, *2013 IEEE High Performance Extreme Computing Conference (HPEC)*, IEEE, pp. 1–6 (2013).

[6] George, A.: Nested dissection of a regular finite element mesh, *SIAM Journal on Numerical Analysis*, Vol. 10, No. 2, pp. 345–363 (1973).

[7] Lipton, R. J., Rose, D. J. and Tarjan, R. E.: Generalized nested dissection, *SIAM journal on numerical analysis*, Vol. 16, No. 2, pp. 346–358 (1979).

[8] Nie, J., Zhang, C., Zou, D., Xia, F., Lu, L., Wang, X. and Zhao, F.: Adaptive sparse matrix-vector multiplication on CPU-GPU heterogeneous architecture, *Proceedings of the 2019 3rd High Performance Computing and Cluster Technologies Conference*, pp. 6–10 (2019).

[9] Alvarez, X., Gorobets, A. and Trias, F. X.: Strategies for the heterogeneous execution of large-scale simulations on hybrid supercomputers, *Proceedings of the European Conference on Computational Fluid Dynamics* (2018).

[10] Chen, Y., Paul, G., Havlin, S., Liljeros, F. and Stanley, H. E.: Finding a better immunization strategy, *Physical Review Letters*, Vol. 101, No. 5, p. 058701 (2008).

[11] Hendrickson, B. and Rothberg, E.: Improving the run time and quality of nested dissection ordering, *SIAM Journal on Scientific Computing*, Vol. 20, No. 2, pp. 468–489 (1998).

[12] Karypis, G. and Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs, *SIAM Journal on Scientific Computing*, Vol. 20, No. 1, pp. 359–392 (1998).

[13] Pothén, A., Simon, H. D. and Liou, K.-P.: Partitioning sparse matrices with eigenvectors of graphs, *SIAM journal on matrix analysis and applications*, Vol. 11, No. 3, pp. 430–452 (1990).

[14] Bui, T. N. and Jones, C.: Finding good approximate vertex and edge partitions is NP-hard, *Information Processing Letters*, Vol. 42, No. 3, pp. 153–159 (1992).

[15] Abou-Rjeili, A. and Karypis, G.: Multilevel algorithms for partitioning power-law graphs, *Proceedings 20th IEEE International Parallel & Distributed Processing Symposium*, IEEE, pp. 10–pp (2006).

[16] Kernighan, B. W. and Lin, S.: An efficient heuristic procedure for partitioning graphs, *The Bell system technical journal*, Vol. 49, No. 2, pp. 291–307 (1970).

[17] Fiduccia, C. M. and Mattheyses, R. M.: A linear-time heuristic for improving network partitions, *19th design automation conference*, IEEE, pp. 175–181 (1982).

[18] Ashcraft, C. and Liu, J. W.: A partition improvement algorithm for generalized nested dissection, *Boeing Computer Services, Seattle, WA, Tech. Rep. BCSTEC-94-020* (1994).

[19] Hendrickson, B. and Rothberg, E.: Improving the run time and quality of nested dissection ordering, *SIAM Journal on Scientific Computing*, Vol. 20, No. 2, pp. 468–489 (1998).

[20] Davis, T.: SuiteSparse Matrix Collection, Texas A&M University (online), available from <https://sparse.tamu.edu/> (accessed 2021-5-29).

[21] Barabási, A.-L. and Albert, R.: Emergence of scaling in random networks, *science*, Vol. 286, No. 5439, pp. 509–512 (1999).