# 社会ネットワークにおける影響最大化問題を解く並列分散アルゴリズムの提案

佐々木亮輔† 豊田 哲也†† 大原 剛三††

† 青山学院大学大学院理工学研究科理工学専攻 〒252–5258 神奈川県相模原市中央区淵野辺 5–10–1
†† 青山学院大学理工学部 〒252–5258 神奈川県相模原市中央区淵野辺 5–10–1
E-mail: †c5615134@aoyama.jp, ††{toyota,ohara}@it.aoyama.ac.jp

**あらまし** 近年，個人や組織等の社会的な主体とその関係性を，それぞれ頂点と辺で表現した社会ネットワークに関する多くの研究成果が報告されている．それらの研究の対象問題の 1 つに影響最大化問題がある．これは，頂点が情報源となったときに対象ネットワーク上で最も広範囲に情報を伝達可能な一定数の頂点集合を求める離散最適化問題であり，その解法は口コミに基づいた顧客の獲得や商品の宣伝を行うバイラルマーケティングの効率化・低コスト化等に応用される．影響最大化問題を解く既存手法の多くは，大規模なネットワークに対応するために何らかの近似手法やヒューリスティックを導入しているが，得られる解の精度が必ずしも理論的に保証されるわけではない．そこで本研究では，情報の伝播に確率モデルを仮定したとき，頂点間の情報伝播を並列に実行するシミュレーション手法を提案する．評価実験により，本手法の効率化手法の一つであるシミュレーションの多重化が並列分散処理の効率化に大きく貢献していることが明らかとなった．

**キーワード** 社会ネットワーク，影響最大化問題，Pregel，並列分散処理

## 1. はじめに

社会ネットワークとは，個人や組織等の社会的な主体とその関係性を，頂点と辺で構成されるネットワークで表現したものである．インターネットの普及とともに大規模な社会ネットワークのデータを収集できるようになったため，社会ネットワークの構造的な特徴や情報拡散の性質などを明らかにしようと，研究が盛んに取り組まれている．そのような研究の対象の一つに，社会ネットワーク上での口コミに基づいた顧客の獲得や商品の宣伝を行うバイラルマーケティング [1,2] がある．この戦略を効率的に行うためには，限られた広告経費でより多くの顧客に情報を伝達することが求められる．これは，情報を事前に与える対象の数を限定した上で影響度を最大化させる離散最適化問題に相当し，影響最大化問題と呼ばれる [3]．ネットワーク上の情報拡散現象を確率過程と捉え，情報伝播モデルと呼ばれる確率モデルでその過程をモデル化した場合，頂点の影響度は，その頂点が情報源として発信した情報を受け取る頂点数の期待値として定量化される．このとき，有向ネットワーク上の影響最大化問題では，与えられた頂点数の下で影響度が最大となるような情報源頂点集合を求める．このような影響最大化問題に対して，モンテカルロ法によるシミュレーションベースの手法 [4,5] を用いることで，その近似解を精度よく求めることが出来るが，大規模なグラフを対象とした高速な処理は困難である．そのため，影響最大化問題を解く既存手法の多くは，大規模ネットワークに対応するためのヒューリスティック [6,7] を導入しているが，得られる解の精度が必ずしも理論的に保証されるわけではない．

以上のような背景から，本稿では，有向ネットワーク上の影響最大化問題に対するシミュレーションベースの並列分散アルゴリズムを提案し，計算の効率化を図る．具体的には，1）辺に各シミュレーションにおける情報伝播の有無を属性として付与してシミュレーションを多重化し，2）情報伝播過程を頂点間の情報伝播に分解し，頂点単位の独立した処理と捉えることで，ネットワーク全体の情報伝播シミュレーションを並列化する．また，提案手法を複数の計算ノードからなる並列分散処理環境に実装し，評価実験を通してその有用性を示す．

## 2. 影響最大化問題の既存研究

本節では，情報の伝播を表現する確率モデルと，頂点集合の影響度，影響最大化問題の定義を述べた後，影響最大化問題をモンテカルロ法に基づいて解く既存手法を概説する．

### 2.1 用語の定義

$V$ を頂点集合，$E \subseteq V^2$ を頂点間の有向辺の集合としたとき，社会ネットワークを有向グラフ $G = (V, E)$ で表現する．$(u, v) \in E$ は頂点 $u$ から頂点 $v$ への有向辺を表し，頂点 $u$ を出頂点，頂点 $v$ を入頂点と呼ぶ．本稿では，$N_o(v) = \{u \mid (u, v) \in E\}$ を $v$ に対する出頂点集合，$N_i(u) = \{v \mid (u, v) \in E\}$ を $u$ に対する入頂点集合とする．ここでは，辺の両端点が同一となる自己閉路や，頂点 $u$，$v$ 間に辺が複数存在するような多重辺が存在しない単純グラフのみを対象とする．

各行と各列がそれぞれ $G$ 中の頂点の 1 つに対応する $|V|$ 行 $|V|$ 列の正方行列 $\boldsymbol{A}^G$ を考える．このとき，頂点 $u$，$v$ に対応する要素 $\boldsymbol{A}^G_{uv}$ が，$(u, v) \in E$ のときに 1，$(u, v) \notin E$ のときに 0 を取るとする．このような行列 $\boldsymbol{A}^G$ をグラフ $G$ の隣接行列と呼ぶ．有向グラフ $G$ においては $\boldsymbol{A}^G_{uv} = \boldsymbol{A}^G_{vu}$ が必ずしも成り立たないことに注意されたい．このことから，隣接行列 $\boldsymbol{A}^G$ の各行は有向

辺の出頂点に対応し，各列は入頂点に対応する．隣接行列 $\boldsymbol{A}^G$ が各頂点から有向辺 1 本を辿って到達可能な頂点を表すのに対し，$\left(\boldsymbol{A}^G\right)^2$ は各頂点から有向辺 2 本を辿って到達可能な頂点を表す．このことから，単位行列を I とするとき，ブール代数演算に基づく行列計算による可到達行列を $\boldsymbol{R}^G = \lim_{n\to\infty}\left(\boldsymbol{A}^G + \mathrm{I}\right)^n$ と定義する．このとき，頂点集合 $U \subseteq V$ の可到達頂点集合は $\hat{R}_U^G = \left\{v \in V \,\middle|\, \boldsymbol{R}_{uv}^G = 1, u \in U\right\}$ と定義できる．

### 2.2 情報伝播モデル

情報伝播モデルとは，ネットワーク上の情報の拡散現象を確率モデルで表現したものである．実際の情報拡散は様々な要因の影響を受けるが，それらすべてをモデル化できないため，情報拡散過程を確率過程として近似し，情報がどのように伝播するかを確率的に予測する．本稿では，独立カスケード（IC: Independent Cascade）モデル [8] と，一定の条件の下で IC モデルと等価とみなされるボンドパーコレーション（BP: Bond Percolation）モデル [4] について述べる．

#### 2.2.1 独立カスケードモデル

IC モデルは，頂点間の情報伝播がその頂点間に割り当てられた確率のみに依存し独立に試行される基本的な情報伝播モデルの一つである．IC モデルでは次のような仮定をおく．

- 頂点は活性もしくは，非活性のいずれかの状態を取る．
- 頂点 $u$ から頂点 $v$ へ情報が伝播する確率 $p_{(u,v)} \in [0,1]$ を伝播確率として辺 $(u,v) \in E$ に事前に与える．
- 情報を受け取った頂点を活性頂点，そうでない頂点を非活性頂点とする．
- 情報伝播過程は，離散時間 $t \geqq 0$ で展開される．
- $S$ を事前に情報が与えられる情報源頂点集合としたとき，時刻 $t = 0$ で $u \in S$ は活性頂点となり，$u \notin S$ は非活性頂点となる．
- 頂点の状態は非活性から活性へ変化し得るが，その逆は起こらない．
- ネットワーク上での情報の伝播は，頂点の活性状態の伝播として表現する．

以上のような仮定の下，IC モデルにおける情報伝播過程を定義する．このとき，時刻 $t > 0$ において新たに活性頂点となる頂点集合を $L_t$ とする．時刻 $t$ の時点で，$u \in L_t$ である頂点 $u$ に対して，$u \in N_o(v)$ であるような非活性頂点 $v$ を考える．ここで，頂点 $u$ から頂点 $v$ への情報伝播は確率 $p_{(u,v)}$ で成功し，成功した場合，時刻 $t+1$ において頂点 $v$ の状態は活性となる．頂点 $v$ に対して，$N_o(v)$ 中の頂点が $L_t$ 中に複数存在する場合，各頂点 $u$ が頂点 $v$ を活性させる試行は時刻 $t$ で任意の順序で独立に行われ，そのうちの少なくとも 1 つが成功した場合，頂点 $v$ は時刻 $t+1$ で活性となる．新たな活性頂点 $L_{t+1}$ は時刻 $t+1$ で同様に隣接する頂点に情報を伝播する．新たな活性頂点集合が空集合となるとき，情報伝播過程は終了する．

#### 2.2.2 ボンドパーコレーションモデル

本節では，文献 [4] に従い BP モデルについて説明する．いま，$|E|$ 次元ベクトルの集合 $Q_G$ を式 (1) のように定義する．

$$Q_G = \left\{\boldsymbol{q} = \left(q_{(u,v)}\right)_{(u,v)\in E} \in \{0,1\}^{|E|}\right\} \tag{1}$$

すなわち，$Q_G$ の元 $\boldsymbol{q}$ の各要素は $G$ 中のいずれかの辺 $(u,v)$ に対応し，その値を $q_{(u,v)}$ としたとき，$q_{(u,v)}$ は 1 か 0 のいずれかの値を取る．このとき，$Q_G$ 上の確率分布 $\gamma$ はグラフ $G$ 上のボンドパーコレーション過程を決定する．具体的には，$\gamma$ に基づき選択されるバイナリベクトル $\boldsymbol{q} \in Q_G$ に対して，辺 $(u,v) \in E$ に対応する値が $q_{(u,v)} = 1$ なら 辺 $(u,v)$ を占領，$q_{(u,v)} = 0$ なら辺 $(u,v)$ を非占領とする．ここで，あるバイナリベクトル $\boldsymbol{q}_o \in Q_G$ に対する占領辺集合を $E_o$ とし，グラフ $G_o = (V, E_o)$ を占領グラフと呼ぶ．このとき，情報源頂点集合 $S$ を起点とした情報伝播過程により得られる活性頂点集合を $G_o$ 上の $S$ からの可到達頂点集合 $\hat{R}_S^{G_o}$ とする決定論的情報伝播モデルを考える．このモデルを $Q_G$ 上の確率分布 $\gamma$ に随伴させた $G$ 上の確率的情報伝播モデルを $G$ 上の BP モデルと呼ぶ．

いま，式 (2) で定義される確率分布 $\gamma(\boldsymbol{q})$ に従う BP モデルは，辺 $(u,v) \in E$ に伝播確率 $p_{(u,v)}$ を与えた IC モデルと等価であると考えることができる．

$$\gamma(\boldsymbol{q}) = \prod_{(u,v)\in E}\left\{\left(p_{(u,v)}{}^{q_{(u,v)}}\right)\left(1 - p_{(u,v)}\right)^{1-q_{(u,v)}}\right\} \tag{2}$$

上記のように，式 (2) を満たすことで IC モデルと同一視される BP モデルを独立カスケードボンドパーコレーション（ICBP: Independent Cascade Bond Percolation）モデルと呼ぶ [4]．すなわち，$G$ に ICBP モデルを仮定し，情報源頂点集合 $S$ が与えられたときの活性頂点集合は，対応する IC モデルにおける活性頂点集合と等価とみなせる．

### 2.3 影 響 度

影響度はネットワーク上の頂点の重要性を示す中心性指標の一つであり，ネットワークの分析に利用されている．影響度の値が大きいほど，ネットワーク上において情報を拡散する性能が高いことを表す．本節では影響度の定義とその計算法を述べる．

#### 2.3.1 頂点集合の影響度の定義

有向グラフ $G = (V, E)$ における情報源頂点集合 $S \in V$ の影響度 $\sigma(S)$ を，$S$ 中の頂点を起点とした情報伝播過程を通してその情報を受け取る頂点数の期待値として定義する．このとき，$\sigma(S)$ の値域は $0 \leqq \sigma(S) \leqq |V|$ となる．$\omega$ を $S$ から情報が伝播した経路，$\mathbb{P}(\omega)$ を $\omega$ の発生確率，$\rho(\omega)$ を $\omega$ 中の頂点数としたとき，IC モデルにおける頂点集合 $S$ の影響度 $\sigma(S)$ を式 (3) と定義する [3,4]．ただし，$\omega$ の総和は，IC モデルの情報伝播過程における情報源頂点集合 $S$ を起点とした全経路で計算される．

$$\sigma(S) = \sum_{\omega}\rho(\omega)\mathbb{P}(\omega) \tag{3}$$

#### 2.3.2 頂点集合の影響度の計算法

影響度を厳密に求めることは，複雑性クラス#P-hard の問題であるため，解くことは困難である [6]．そのため，その値を推定する方法が幾つか提案されている．本稿では，その中の一つであるモンテカルロ法によるシミュレーションベースの計算法 [3] を利用する．この手法は，情報源頂点集合 $S \subseteq V$ 中の頂点を起点とした $M$ 回の情報伝播シミュレーションを試行し，各結果において情報を受け取った頂点数の平均により $S$ の影響度

$\sigma(S)$ を近似する．

IC モデルを仮定した場合の具体的な影響度の計算法について述べる．ここで，与えられた情報源頂点集合 $S$ に対して，IC モデルの下で情報伝播シミュレーションにより活性頂点集合を求めることは，対応する ICBP モデルにおいて，$Q_G$ から確率分布 $\gamma$ に従って選択したベクトル $\boldsymbol{q}_o$ に基づき生成した占領グラフ $G_o$ 上で $S$ の可到達頂点集合 $\hat{R}_S^{G_o}$ を求めることに等しい．

いま，グラフ $G$ 上のある IC モデルに対応する ICBP モデルにおいて，その確率分布 $\gamma$ に従い $M$ 回のシミュレーションそれぞれに対応するベクトル $\boldsymbol{q}_1, \cdots, \boldsymbol{q}_M \in Q_G$ を生成する．各 $\boldsymbol{q}_m$ により生成される占領辺集合 $E_m$ に対して得られる占領グラフを $G_m = (V, E_m)$ とする．このとき，$G_m$ の隣接行列 $\boldsymbol{A}^{G_m}$ の各要素 $\boldsymbol{A}_{uv}^{G_m}$ は式 (4) を満たす．

$$\boldsymbol{A}_{uv}^{G_m} = \begin{cases} 1 & q_{(u,v)}^m = 1 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

隣接行列 $\boldsymbol{A}^{G_m}$ に対する可到達行列 $\boldsymbol{R}^{G_m}$ から生成される頂点集合 $S$ の可到達頂点集合を $\hat{R}_S^{G_m}$ とするとき，$G$ 上の頂点集合 $S$ を起点とした $M$ 回の ICBP モデルに基づく情報伝播シミュレーションにより得られる $S$ の影響度 $\sigma(S)$ は式 (5) により求めることができる．

$$\sigma(S) \approx \sigma_M(S) = \frac{1}{M} \sum_{m=1}^{M} \hat{R}_S^{G_m} \tag{5}$$

このシミュレーションは経験的に $10^2$ から $10^4$ 回程度行われ，その時間計算量は $O(M|V||E|)$ となる．

### 2.4 影響最大化問題

影響最大化問題とは，ネットワーク上の頂点から拡散力の高い頂点集合を選択する最適化問題である [3–5]．この問題を解くことは，バイラルマーケティングや P2P のネットワークなどにおいて強い影響力を持つ少数の頂点を特定し，広告費用や通信コストを最小化することに応用できる．本節では，影響最大化問題の定義とその解法について述べる．

#### 2.4.1 影響最大化問題の定義

ある情報伝播モデルに従う有向グラフ $G = (V, E)$ 上の情報伝播に対して，影響度が最大となる大きさ $K$ の強影響力頂点集合 $I_K \subseteq V$ を求める離散最適化問題を影響最大化問題と呼ぶ．このとき，$K \leq |V|$ は強影響力頂点数を表す正整数であり，事前に与えられる．形式的には，強影響力頂点集合 $I_K$ は式 (6) のように定義できる．

$$I_K = \mathrm{argmax}_{I \in \{S \subseteq V \mid |S| = K\}} \sigma(I) \tag{6}$$

#### 2.4.2 影響最大化問題の解法

IC モデルを仮定した影響最大化問題を厳密に解くことは，複雑性クラス NP-hard に属する集合被覆問題を解くことと同義であるため，その解を求めることは困難である [3]．そのため，近似手法として貪欲法に基づく解法が提案されている [3]．具体的には，頂点数 1 の解 $I_1$ として影響度 $\sigma(\{v\})$ が最大となるような頂点 $v$ を選択し，次に $\sigma(I_1 \cup \{v'\})$ が最大となるような $v'$ を $v$ に加えることで頂点数 2 の解 $I_2$ を得る．これを頂点数が $K$ となるまで繰り返す．このとき，選択した解を再考することはないため，影響最大化問題を多項式時間で解くことができる．この手順を Algorithm 1 にまとめる．このアルゴリズムの時間計算量は，$O(KM|V||E|)$ となる．また，貪欲アルゴリズムにより得られる任意の解 $I_K$ は，最適解と比較して少なくとも約 63%の解の精度が保証される [9].

**Algorithm 1** InfluenceMaximization $(V, E, K, M)$

**Input:** $V, E, K, M$

**Output:** $I$

```
I ← ∅
while |I| < K do
    i ← argmax_{v∈V\I} σ_M(I ∪ {v})
    I ← I ∪ {i}
end while
return I
```

## 3. グラフ構造と並列分散処理

本節では，提案手法が前提とするグラフアルゴリズムの並列化・分散化の概念として，データ並列計算の実行モデルであるバルク同期並列 [10] と頂点中心アプローチ [11] に基づくグラフ並列計算モデルを述べる．

### 3.1 データ並列計算モデル

バルク同期並列（BSP: Bulk Synchronous Parallel）とは，各データを分割し独立した処理を並列に実行するデータ並列計算モデルの一つである．このモデルの適用は，プロセッサ数に対する実行時間の線形的減少を目的とする．BSP の概念を図 1 に示す．処理過程は，プロセッサが持つローカルメモリにおける独立した処理（ローカル処理），プロセッサ間のデータの送受信（通信），ローカル処理及びデータの通信の終了までの待機（バルク同期）の 3 つの処理で構成される．この一連の処理はスーパーステップと呼ばれ，それを反復的に実行することで出力を得る．

### 3.2 グラフ並列計算モデル

グラフにおける反復的な並列計算に特化したグラフ並列計算モデルの一つに，頂点を中心にした処理を実行する頂点中心アプローチに基づく収集・適用・分散（GAS: Gather Apply Scatter）モデル [12] がある．GAS モデルは，1）隣接頂点から届いたメッセージを一つの出力にまとめ（収集），2）頂点が持つ情報をその出力を用いて更新し（適用），3）新たに作られたメッセージを隣接頂点へ送信する（分散），という 3 つのデータ並列計算の段階を並列ループとして実行する計算モデルである．ここで，メッセージとはプロセッサ間のデータの送受信の内容を指す．このモデルは，グラフを分割することで処理負荷の分散を図るグラフパーティショニングのアルゴリズムをユーザから隠蔽でき，かつ直感的で理解しやすいため，グラフ処理の並列・分散化では広く用いられている．ここで，モデルの制約は，頂点に対して隣接頂点以外の頂点情報を不可視にすることに注意されたい．

プロセッサ総数が $n$ 個の並列分散クラスタ上で実行するとき

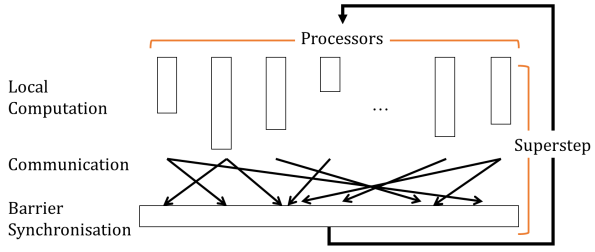


図 1　BSP の計算モデル

の GAS モデルの実装は，グラフ $G$ 上の頂点集合 $V$ を，$i \neq j$ に対して $V_i \cap V_j = \emptyset, \bigcup_{k=1}^{n} V_k = V$ となるような $n$ 個の部分頂点集合 $V_1, \cdots, V_n \subseteq V$ に分割し，それらを各プロセッサに割り当てることで分散処理を実行する．本稿では，GAS モデルの実装の一つとして BSP に基づく Pregel [11] を用いる．

ここで，辺 $(u, v)$ に着目した GAS モデルの計算構造（収集，適用，分散）の具体的に説明する．以下では頂点や辺に持たせる何らかの情報をそれぞれ $\boldsymbol{d}_V$，$\boldsymbol{d}_E$ とし，それらを頂点集合 $V$ の属性ベクトル，辺集合 $E$ の属性ベクトルと呼ぶ．そのとき，$\boldsymbol{d}_V$ の元 $d_u$，$d_v$ をそれぞれ頂点 $u$，$v$ の頂点属性，$\boldsymbol{d}_E$ の元 $d_{(u,v)}$ を辺 $(u, v)$ の辺属性とする．

収集段階（Gather）

処理対象となる頂点 $v$ は，送信頂点 $u$ から受け取る各メッセージ $\mathcal{M}_u$ を入力とする収集処理 $g$ を実行し，併合結果 $\Sigma_v$ を得る．形式的には，関数 $g$ を用いて式 (7) のように定義できる．

$$\Sigma_v \leftarrow g(\{\mathcal{M}_u \mid u \in N_i(v)\}) \tag{7}$$

適用段階（Apply）

処理対象となる頂点 $u$ は，併合結果 $\Sigma_u$ と頂点属性 $d_u$ を入力とする適用処理 $a$ を実行することで，頂点属性 $d_u$ を更新する．形式的には，関数 $a$ を用いて式 (8) のように定義できる．

$$d_u \leftarrow a(d_u, \Sigma_u) \tag{8}$$

分散段階（Scatter）

処理対象となる頂点 $u$ は，頂点属性 $d_u$ と辺属性 $d_{(u,v)}$ を入力とする分散処理 $s$ を実行し，その結果得られるメッセージ $\mathcal{M}_u$ を頂点 $v$ へ送信する．形式的には，関数 $s$ を用いて式 (9) のように定義できる．

$$\mathcal{M}_u \leftarrow s(d_u, d_{(u,v)}) \tag{9}$$

通常は，初期化された情報を併合結果として頂点にあらかじめ与え，適用，分散，収集の順序で処理を反復する．また，収集段階で少なくとも一つのメッセージを処理する場合は頂点を活動状態とし，そうでない場合は頂点を休眠状態とする．全頂点が休眠状態となるとき，この一連の処理を終了する．この GAS モデルの計算過程を図 2 に示す．

## 4. 提 案 手 法

本研究では，有向ネットワーク上で IC モデルを仮定した場合の影響最大化問題を解くシミュレーションベースの並列分散

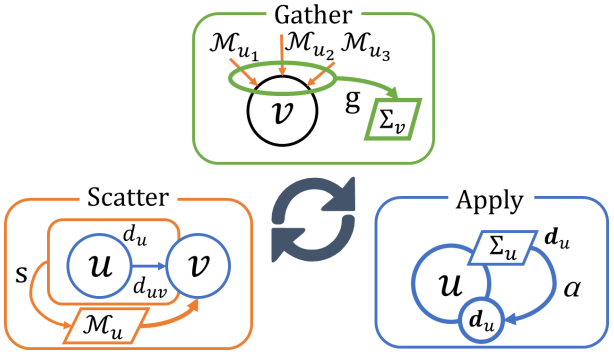


図 2　GAS モデル

アルゴリズムを提案し，計算の効率化を図る．グラフの並列処理には前述の Pregel を利用することを前提とし，頂点間の情報交換を基本的な処理単位とした並列分散アルゴリズムを提案する．具体的には，1）辺にシミュレーションに関する情報を与えることで各シミュレーションに対応する ICBP モデルにおける複数の占領グラフを一つにまとめ，複数回のシミュレーションにおける情報伝播を一度の情報伝播で伝える（**シミュレーションの多重化**）．また，2）シミュレーションに関する情報に基づいて独立に試行される頂点間の情報伝播をプロセッサ単位で同時に処理する（**並列分散化**）．

### 4.1 並列分散化への課題と解決法

既存研究 [3,4] では複数の占領グラフを生成することで影響最大化問題を解くが，複数の占領グラフに対して単純に頂点中心アプローチの並列分散アルゴリズムを実装する場合，各頂点における処理量が少ないために並列化の粒度の低下を引き起こす．グラフ処理の並列計算では，並列化の粒度の低下は細かい通信の発生によるレイテンシの増大とスループットの低下を招くことが問題となる．

そこで本手法では，Algorithm 1 における強影響力頂点集合 $I_K$ を求める $K$ 回の処理のそれぞれで $M$ 個の占領グラフ $G_1, \cdots, G_M$ を再利用して $\sigma(I_k)$ を計算し，生成する占領グラフ数を削減する手法を導入する [5]．また，その手法に加えて，提案手法では $G_1, \cdots, G_M$ 上の情報伝播シミュレーションを多重化する．具体的には，対象グラフ $G = (V, E)$ の辺 $(u, v) \in E$ に対して，各占領グラフ $G_1, \cdots, G_M$ における占領・非占領状態を表す $M$ 次元バイナリベクトルを属性として与える．このような属性を付与した辺集合を $E'$ としたとき，$G' = (V, E')$ は占領グラフ $G_1, \cdots, G_M$ を重ね合わせたグラフとみなすことができることから，$G'$ 上の一度の情報伝播シミュレーションを通して，$G_1, \cdots, G_M$ 上のシミュレーション結果をまとめて計算する方法を提案する．そのシミュレーションの多重化の概念図を図 3 に示す．具体的には，この $G'$ 上で，GAS モデルに従って各頂点は自身の可到達頂点集合を有向辺とは逆の向きに隣接頂点へ伝播する．その際に，辺に付与されたバイナリベクトルを利用することで，複数回のシミュレーションにおける頂点の可到達頂点集合を求め，その出力をもとに強影響力頂点集合を選択する．また，可到達頂点集合の逆伝播では各頂点間で発生する情報伝播が独立の試行であること，強影響力頂点の選択では各頂点が持つ可到達頂点集

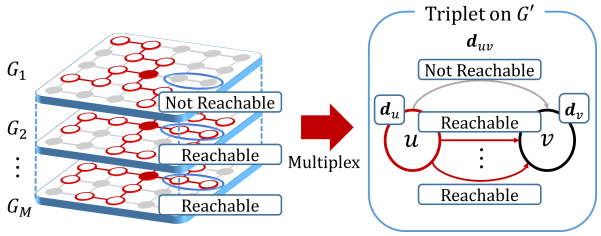


図 3　シミュレーションの多重化

合を頂点単位に独立に処理可能であることに着目し，それぞれの独立した処理を並列に実行する．以降の節では，多重化された単一のグラフ $G'$ を対象とした情報伝播過程を定義するために，辺と頂点のデータ構造を述べた後に，提案する並列分散アルゴリズムの詳細を説明する．

### 4.2　辺のデータ構造

複数のシミュレーションにおける辺の有無を事前に確定することを考える．ここで，$M$ 次元ベクトル集合 $Q_{G'}$ を式 (10) に定義する．

$$Q_{G'} = \left\{ \boldsymbol{q'}_{(u,v)} = \left(q^m_{(u,v)}\right)_{m\in\{1,\cdots,M\}} \,\middle|\, (u,v) \in E, \boldsymbol{q}^m \in Q_G \right\} \tag{10}$$

辺 $(u,v)$ に対応する属性 $\boldsymbol{q'}_{(u,v)}$ の元 $q^1_{(u,v)}, \cdots, q^M_{(u,v)}$ は占領グラフ $G_1, \cdots, G_M$ の辺 $(u,v)$ に対応することに注意されたい．具体的には，占領グラフ $G_m = (V, E_m)$ における辺 $(u,v) \in E_m$ の占領状態 $q^m_{(u,v)}$ を式 (11) で定義する．

$$q^m_{(u,v)} = \begin{cases} 1 & \text{if } (u,v) \in E_m \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

このとき，辺 $(u,v) \in E'$ の属性に $M$ 次元ベクトル $\boldsymbol{d}_{(u,v)} \in Q_{G'}$ を持たせた辺集合 $E'$ で表現されるグラフ $G' = (V, E')$ は，グラフを多重化している．

### 4.3　頂点のデータ構造

各頂点がもつ基本的な情報は，有向辺で接続した隣接頂点のみである．これに対して，頂点 $u$ を出頂点とする有向辺 $(u,v)$ が存在した場合，頂点 $v$ の隣接頂点，すなわち $v$ から 1 本の有向辺で到達できる頂点集合 $N_i(v)$ を $u$ に送信することで，頂点 $u$ は頂点 $v$ を介して有向辺を 2 本辿って到達可能な頂点を知ることができる．同様に，頂点 $v$ が有向辺を 2 本辿って到達可能な頂点集合を知っていた場合，その情報を頂点 $u$ に送信すれば，頂点 $u$ は頂点 $v$ を介して有向辺を 3 本辿って到達可能な頂点を知ることができる．これを一般化すると，単位時間ごとに，各頂点から到達可能な頂点集合を有向辺とは逆向きに隣接頂点に送信することで，$t$ ステップ目には各頂点は有向辺を $t$ 本辿って到達可能な頂点集合を知ることができる（$t = 1$ のときは各頂点は自身に隣接する頂点集合を得るものとする）．この逆伝播は，全頂点の可到達頂点集合が変化しなくなった時点で終了できる．ここで，実際の頂点 $u$ の可到達頂点情報は，各行を到達可能な頂点，各列をシミュレーションとした $|V| \times M$ の可到達頂点行列 $\boldsymbol{D}^u$ で定義される．ここで，$\boldsymbol{D}^u$ の頂点 $v$ に対応する行ベクトル $\boldsymbol{D}^u_v$ は，頂点 $u$ から頂点 $v$ への到達可能性を示し，$m$ 回目のシミュレーションに対応する占領グラフ $G_m$ において $u$ から $v$ に到達可能な場合，$\boldsymbol{D}^u_v$ の元 $b_m$ は 1 となり，到達不可能な場合，$b_m$ は 0 となる．

ここで，$G'$ の有向辺 $E'$ に含まれる各辺を反転した辺集合を $E'_r$ とするとき，辺 $(u,v) \in E'_r$ における情報伝播を考える．ここで，バイナリベクトルに対する論理和と論理積はビット単位の論理和，論理積とする．出頂点 $u$ では，可到達頂点行列を入頂点 $v$ へ伝播する際に，その伝播による可到達頂点行列の更新は可到達頂点行列 $D^u$ の各行ベクトル $\boldsymbol{D}^u_v$ について辺属性 $d_{(u,v)}$ の論理積を取ることで計算される．また，出頂点 $v$ では，入頂点集合 $N_i(v)$ から受け取る可到達頂点行列群を一つの可到達頂点行列に併合するときや自身の確定可到達頂点行列を更新する際に，可到達頂点行列同士の対応する行ベクトルについて，論理和を取ることで新たな可到達頂点行列を生成する．

しかし，実際の可到達頂点行列は多くの行ベクトルが零ベクトル $\boldsymbol{0}$ で表されるため，計算機への格納効率が悪化する．そこで，集合としてデータを格納することで不必要な保存領域を取り除き，その格納効率を改善することを考える．ここで，零ベクトルを取り除いた頂点 $u$ の可到達頂点集合 $d_u$ を式 (12) で定義する．

$$d_u = \{\boldsymbol{D}^u_i \,|\, i \in V, \boldsymbol{D}^u_i \neq \boldsymbol{0}\} \tag{12}$$

本手法では，隣接頂点と可到達頂点集合を交換することで，頂点 $u$ は最終的な確定可到達頂点集合 $d_u$ を求める．

### 4.4　可到達頂点集合の逆伝播の並列化

GAS モデルに従うグラフ $G'$ 上の可到達頂点集合の逆伝播により各頂点 $u$ の確定可到達頂点集合 $d_u$ を求める．情報伝播過程は離散時間 $t \geqq 0$ で展開され，$t$ が 1 進むことは次のスーパーステップに移行することを意味する．ここで，情報伝播過程において頂点が逆伝播する情報は，時刻 $t$ でその頂点が有向辺を $t$ 本辿って到達可能な可到達頂点集合である．辺 $(u,v) \in E'_r$ 上において頂点 $u$ から頂点 $v$ へ可到達頂点集合を送信する．辺 $(u,v)$ で発生する情報伝播が独立の試行であることから，これらの試行を並列に実行する．

辺 $(u,v) \in E'_r$ における情報伝播を考える．$\Sigma_u$ を頂点 $u$ の可到達頂点集合とするとき，$u$ から $u$ への併合結果の要素 $\boldsymbol{\Sigma}_{uu} \in \Sigma_u$ に全ての元が 1 となるバイナリベクトル $\boldsymbol{1}^M$ を事前に与える．これは，いずれのシミュレーションの場合でも頂点が発するメッセージはそれ自身に必ず到達可能であることを示す．以降の処理は，GAS モデルに従って適用段階・分散段階・収集段階の順番で反復処理を実行し，可到達頂点集合の要素であるバイナリベクトルと辺に与えられたバイナリベクトルの論理積を隣接頂点に伝播することで確定可到達頂点集合 $d_u$ を求める．ここで，メッセージ $\mathcal{M}_u$，併合結果 $\Sigma_u$，$\Sigma_v$ の各データ構造は $d_u$ と同型である．

適用段階

隣接頂点から受け取る可到達頂点集合を併合した可到達頂点集合 $\Sigma_u$ と更新前の確定可到達頂点集合 $d_u$ のそれぞれが持つ対応するバイナリベクトルの論理和から得られる結果を新たな確定可到達頂点集合 $d_u$ とする．

分散段階

頂点 $v$ へ伝播する可到達頂点集合 $\Sigma_u$ の各要素 $\boldsymbol{b}_u$ と対応する辺の属性 $\boldsymbol{d}_{(u,v)}$ の論理積を計算し，$u$ から $v$ へ伝播する可到達頂点集合 $\mathcal{M}_u$ を求める．ここで，あるバイナリベクトル $\boldsymbol{b}_u$ について，$\boldsymbol{b}_u \wedge \boldsymbol{d}_{(u,v)} = \boldsymbol{0}$ を満たす場合，到達する情報がないことを表すため，そのバイナリベクトル $\boldsymbol{b}_u$ の伝播を終える．

収集段階

頂点 $v$ に対して，$u \in N_o(v)$ となる頂点 $u$ が時刻 $t$ で複数存在する場合，各頂点 $u$ から頂点 $v$ へ時刻 $t$ で任意の順序でメッセージ $\mathcal{M}_u$ を受け取る．これらの各メッセージの論理和を計算することでメッセージの併合結果 $\Sigma_v$ が求められる．

全頂点において情報伝播により受け取る可到達頂点集合と自身が持つ確定可到達頂点集合の差分に変化が生じない場合，つまり，活動頂点集合が休眠状態になるとき，これらの一連の反復処理は停止する．この反復処理を終えた後に，更新済みの全頂点の可到達頂点集合をまとめた可到達頂点集合ベクトル $\boldsymbol{d}_V$ に対して強影響力頂点集合の選択アルゴリズムを実行することで，強影響力頂点集合を得る．

提案する並列分散アルゴリズムを Algorithm 2 にまとめる．ここで，Algorithm 2 中の "**in parallel**"および "**in serial**"は，記述個所のブロック以降の処理をそれぞれ並列，逐次的に実行することを示す．ここで，Algorithm 2 の 6〜12 行目は分散処理，13〜15 行目は収集処理，17〜20 行目は適用処理に対応する．

### 4.5 強影響力頂点を選択するアルゴリズムの並列分散化

可到達頂点集合の逆伝播により求めた確定可到達頂点集合ベクトル $\boldsymbol{d}_V$ から強影響力頂点集合 $I$ を求める．まず，各頂点 $v$ が持つ $d_v$ は頂点毎に独立した情報であることから，$d_v$ の各シミュレーションにおける可到達頂点数を並列に計算し，影響度 $\sigma(\{v\})$ を求める．このとき，影響度を最大にする頂点 $i$ を選択し，強影響力頂点集合 $I$ に加える．その後に各シミュレーションにおいて頂点 $i$ から各頂点 $v \in V$ に到達した情報を取り除くために，強影響力頂点 $i$ から頂点 $v$ への到達情報を示すバイナリベクトル $\boldsymbol{d}_{iv}$ の否定を取ってビットを反転させたものと頂点 $u$ から頂点 $v$ への到達情報を示すバイナリベクトル $\boldsymbol{d}_{uv}$ の論理積を計算することで確定可到達頂点集合ベクトルを更新する．なお，$d_{(u,v)}$ は辺の属性を示すバイナリベクトルであるのに対して，$d_{uv}$ は到達情報を示すバイナリベクトルであることに注意されたい．この更新は，$I_k \cup \{v\}$ に対する可到達頂点集合と $I_k$ の可到達頂点集合の差分を最大化する $v$ を求めることが，$\sigma(I_k \cup \{v\})$ を最大化することと等価であることを利用している．この一連の処理を貪欲法により反復し，局所最適解を選択し続けることで，大きさ $K$ の強影響力頂点集合 $I$ が求められる．貪欲法による強影響力頂点集合を選択する並列分散アルゴリズムを Algorithm 3 にまとめる．

### 4.6 解の精度

ICBP モデルを仮定した有向ネットワーク $G'$ 上の任意の情報源頂点集合 $S$ の影響度を導出する．その影響度 $\sigma(S)$ を式 (13) に示す．影響度は情報源頂点集合に対する可到達頂点集合ベクトルの各要素の総和で求められる．式 (13) は，情報源頂点集合 $S$ の各シミュレーションにおける可到達頂点集合の数と等しいため，式 (5) と同値となり，提案手法の解の精度は保証される．

$$\sigma(S) = \frac{1}{M} \sum_{u \in S} \sum_{\boldsymbol{b} \in d_u} |\boldsymbol{b}| \tag{13}$$

**Algorithm 2** Backward $\left(V, E'_r, \mathbf{d}_{E'_r}, K\right)$

**Input:** $V, E'_r, \mathbf{d}_{E'_r}, K$
**Output:** $I$

1: $t \leftarrow 1, V_a \leftarrow V, \mathbf{d}_V \leftarrow \left(\{\emptyset\}^{|V|}\right)$
2: **for all** $u \in V$ **do** $\boldsymbol{\Sigma}_{uu} \leftarrow \mathbf{1}^M$
3: **while** $V_a \neq \emptyset$ **do**
4: **if** $t \neq 1$ **then**
5: $V'_a \leftarrow V_a, V_a \leftarrow \emptyset$ **in serial**
6: **for all** $(u, v) \in E'_r$ **in parallel** $u \in V_a$ **is satisfied do**
7: $\mathcal{M}_u \leftarrow \emptyset$
8: **for all** $\boldsymbol{b}_u \in B_u$ **do**
9: **if** $\boldsymbol{b}_u \wedge \boldsymbol{d}_{(u,v)} = \text{true}$ **then** $\mathcal{M}_u \leftarrow \mathcal{M}_u \cup \left\{\boldsymbol{b}_u \wedge \boldsymbol{d}_{(u,v)}\right\}$
10: $V_a \leftarrow V_a \cup \{v\}$
11: **end for**
12: **end for**
13: **for all** $v \in V_a$ **in parallel do**
14: **for all** $u \in N_o(v)$ **do** $\Sigma_v \leftarrow \Sigma_v \vee \mathcal{M}_u$
15: **end for**
16: **end if**
17: **for all** $u \in V'_a$ **in parallel do**
18: $d_u \leftarrow d_u \vee \Sigma_u$
19: $B_u \leftarrow \Sigma_u$
20: **end for**
21: $t \leftarrow t + 1$ **in serial**
22: **end while**
23: $I \leftarrow \text{Greedy}(\boldsymbol{d}_V, V, K)$
24: **return** $I$

**Algorithm 3** Greedy$(\boldsymbol{d}_V, V, K)$

**Input:** $\boldsymbol{d}_V, V, K$
**Output:** $I$

1: $I \leftarrow \emptyset$
2: **while** $|I| < K$ **do**
3: **for all** $v \in (V \setminus I)$ **in parallel do** $\sigma(\{v\}) \leftarrow |d_v|$
4: $i \leftarrow \text{argmax}_{v \in V \setminus I} \sigma(\{v\})$
5: $I \leftarrow I \cup \{i\}$
6: **for all** $v \in V$ **in parallel do**
7: **for all** $u \in V$ **in parallel do** $\boldsymbol{d}_{uv} \leftarrow \boldsymbol{d}_{uv} \wedge \neg \boldsymbol{d}_{iv}$
8: **end for**
9: **end while**
10: **return** $I$

## 5. 評価実験

### 5.1 実験環境

本研究では，並列分散処理実行環境 Spark と，グラフ並列計算用ライブラリ GraphX を用いて提案手法を実装した．プログ

ラミング言語 Scala で実装された各手法を Linux サーバー (注1)（プロセッサ・ナンバー：Xeon E5-2640 v3，周波数：2.60GHz，メモリ：256GB，コア数：8，スレッド数：16，インテル® ハイパースレッディング・テクノロジー搭載のため，実質上のスレッド数は 32）の環境下に仮想マシン（仮想ソフトウェア名：VMware Player 7.0，メモリ：62GB，コア数：6）を複数台起動することで擬似的な分散環境を構築した．なお，各ソフトウェアのバージョンは，Spark：2.0.0-preview，Hadoop（YARN, HDFS）：2.7.1，Java：1.7.0_45，Scala：2.11.8 である．

本手法の入力とするデータセットには，2005 年度に goo ブログで収集された JR 福知山線脱線事故というテーマのブログからトラックバックを 10 段階たどったネットワーク [4] を用いた．そのネットワークの頂点数は 12,047，辺数は 53,315 である．その入次数分布と出次数分布は冪乗則に従い，グラフ平均直径が短いため，スケールフリー性とスモールワールド性を満たす．そのことから，提案手法を評価するデータセットとして適当であると考える．

本評価実験で用いた影響最大化問題と並列分散処理に関連する計五つのパラメータのデフォルト値を表 1 に示す．ここでグラフパーティショニングの実装は，グラフ $G$ の頂点集合 $V$ をプロセッサ数に分割し，それぞれのプロセッサに部分頂点集合 $V_i \in V$ は割り当てる．本実験では，計算機数，プロセッサ数，強影響力頂点数，シミュレーション数の四種類のパラメータそれぞれを変化させた場合の実行時間を評価し，考察する．各パラメータに対する実験結果は，いずれも 10 回試行の平均である．

### 5.2 実験結果

本実験の評価と考察を述べる．

#### 5.2.1 計算機数の変化に対する実行時間の評価

仮想計算機数に対する提案手法の実行時間を比較し，並列分散化の効果を検証した．計算機数に対する提案手法の実行結果を表 2 に示す．ここで，速度向上比は 1 台の計算機と複数台の計算機における平均実行時間の比で算出した．この結果から，計算機数の増加に対して実行時間は線形的に減少し，速度向上比が線形的に増加していることが確認できる．しかしながら，計算機数に等しい速度向上比は得られなかった．その理由は，並列化の粒度が提案手法でもまだ低いためであると考えられる．

#### 5.2.2 プロセッサ数の変化に対する実行時間の評価

本実験では，仮想計算機は使用せずに物理計算機上でプロセッサ数に対する実行時間を比較し，並列化の効果を検証した．プロセッサ数に対する実行時間を表 3 に示す．物理計算機の JVM のヒープメモリ量に 200GB を与えて実行時間を計測した．プロセッサ数が 8 以下の場合は実行時間の単調減少傾向がみられるが，16 以上の場合は逆に単調増加傾向がみられた．この原因を調べるために，詳細な処理時間を確認したところ，プロセッサ数に比例して，ガベージコレクション（GC: Garbage Collection）にかかる時間が増加することを確認した．プロセッサ数が増加するにつれて GC が占める割合が大きくなる原因は，ヒープメモリに生成されたオブジェクトを処理する計算がプロセッサ数に対して線形的に増加するためと考えられる．そのため，GC 時間を改善することでプロセッサ数に応じた速度向上比が得られると考えられる．

表 1　パラメータのデフォルト値

| パラメータ | デフォルト値 |
|---|---|
| 計算機数 | 4 |
| 強影響力頂点数 $K$ | 5 |
| 計算機 1 台あたりのプロセッサ数 | 6 |
| シミュレーション数 $M$ | 1,000 |
| 伝播確率 $p$ | 0.1 |

表 2　計算機数に対する提案手法の実行時間（秒，試行数：10）

| 計算機数 | 平均 | 標準偏差 | 最小 | 最大 | 平均実行時間の速度向上比 |
|---|---|---|---|---|---|
| 1 | 1143.0 | 57.2 | 1059 | 1252 | 1.00 |
| 2 | 850.7 | 70.9 | 701 | 929 | 1.34 |
| 3 | 805.6 | 63.9 | 733 | 917 | 1.42 |
| 4 | 767.2 | 30.7 | 732 | 825 | 1.49 |

#### 5.2.3 強影響力頂点数の変化に対する実行時間の評価

強影響力頂点数を変化させた場合の実行時間を比較した．その結果を図 4 に示す．図 4 中のエラーバーは，95%信頼区間を表す．この結果から，強影響力頂点数が増加するに従い，実行時間は線形的に増加することがわかる．ただし，その増加率はそれほど大きくない．これは，強影響力頂点集合を選択する並列アルゴリズムは，データ並列計算で処理するため，通信量が少なく各計算機に独立した計算をより多く与えることができる傾向にあるためと考えられる．この処理は並列分散環境にとって好条件であり，高速に計算が可能である．

#### 5.2.4 シミュレーション数の変化に対する実行時間の評価

シミュレーション数に対する実行時間を図 5 に示す．ここで，図 5 中のエラーバーは，95%信頼区間を示す．図 5 から，シミュレーション数に対して提案手法の平均実行時間は線形に増加することがわかる．シミュレーション数 125 の平均実行時間に対するシミュレーション数 125〜1,000 の平均実行時間の速度向上率について単回帰分析を実行したところ，実行時間はシミュレーション数に正比例し，その回帰係数は 0.83 であった．基本的には，シミュレーションの $x$ 倍に対して実行時間は同様に増倍するため，回帰係数が 1 未満になることは考えられない．これは，アルゴリズムを除いた処理時間が減少したことを意味し，具体的には通信効率の改善を意味すると考えられる．また，1 回のシミュレーション処理を逐次的に 1,000 回実行するときの平均実行時間と比較して，シミュレーション数 1,000 回の同時実行による平均実行時間は，約 40.4 倍高速化されることを確認した．このことから，シミュレーションの多重化が提案手法の並列分散処理の効率化に大きく貢献していると言える．提案手法では，通信処理よりも計算機内処理に比重を置いたため，並列化の粒度が高くなり，シミュレーションの並列実行が逐次実行より高速に動作したと考えられる．

(注1)：Intel® Xeon® Processor E5-2640 v3 の性能, http://ark.intel.com/ja/products/83359/Intel-Xeon-Processor-E5-2640-v3-20M-Cache-2_60-GHz (2017)

表 3　プロセッサ数に対する物理環境上の提案手法の実行時間（秒，試行数：10）

| プロセッサ数 | 実行時間 | 速度向上比 | ある実行結果の実行時間において GC が占める割合 |
|---|---|---|---|
| 1 | 1317.6 | 1.00 | 4% |
| 2 | 975.5 | 1.35 | 8% |
| 4 | 750.4 | 1.76 | 14% |
| 8 | 649.7 | 2.03 | 20% |
| 16 | 771.5 | 1.71 | 25% |
| 32 | 913.5 | 1.44 | 27% |

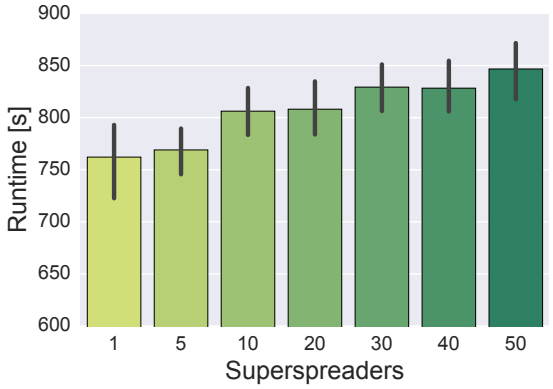


図 4　強影響力頂点数に対する提案手法の実行時間（秒，試行数 10）

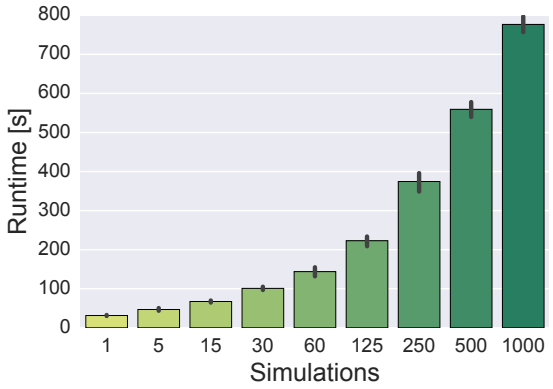


図 5　シミュレーション数に対する提案手法の実行時間（秒，試行数 10）

## 6. おわりに

本稿では，有向ネットワーク上の影響最大化問題に対するシミュレーションベースの並列分散アルゴリズムを提案した．評価実験では，計算機数，プロセッサ数，強影響力頂点数，シミュレーション数の 4 種類のパラメータの変化に対して実行時間を評価した．並列分散環境の検証では，プロセッサ数が 8 のときに最大 2.03 倍，計算機数が 4 のときに最大 1.49 倍の計算速度の向上率を確認し，プロセッサ数及び計算機数の増加に対して実行時間が減少することを示した．しかし，プロセッサ数が 16 以上の場合にガベージコレクションによる CPU 時間の消費により実行時間が単調増加傾向を示したことから独立した計算機内では最大 8 程度のプロセッサ数が有効であると考えられる．また，強影響力頂点集合の検証では，データ並列計算により各計算機で高速に処理が可能であることを示した．シミュレーション数の検証では，実行時間はシミュレーション数に正比例し，その回帰係数が 0.83 程度であること，および 1,000 回のシミュレーションを逐次的に実行した場合と比べて提案手法の実行時間が約 40.4 倍高速であることを確認した．このことから，シミュレーションの多重化が提案手法の並列分散処理の効率化に大きく貢献していることが明らかとなった．今後の課題に，並列化の粒度向上，冗長な計算処理の削除，およびグラフパーティショニングと計算機負荷の関係調査等が挙げられる．

## 7. 謝　　辞

本実験で使用したグラフデータは，龍谷大学理工学部電子情報学科 木村昌弘 教授，静岡県立大学経営情報学部 斉藤和巳 教授，大阪大学産業科学研究所 元田浩 教授にご提供いただいた．また，本研究の一部は，日本学術振興会科学研究費補助金（基盤研究 (C)，課題番号：26330261）による．

## 文　　献

[1] Domingos, P. and Richardson, M.: Mining the Network Value of Customers, Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 57–66 (2001).

[2] Richardson, M. and Domingos, P.: Mining Knowledge-Sharing Sites for Viral Marketing, Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 61–70 (2002).

[3] Kempe, D., Kleinberg, J., and Tardos, È.: Maximizing the Spread of Influence Through a Social Network, Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146 (2003).

[4] Kimura, M., Saito, K., Nakano, R., and Motoda, H.: Extracting Influential Nodes on a Social Network for Information Diffusion, Data Mining and Knowledge Discovery, Vol. 20, No. 1, pp. 70–97 (2010).

[5] Ohsaka, N., Akiba, T., Yoshida, Y., and Kawarabayashi, K.: Fast and Accurate Influence Maximization on Large Networks with Pruned Monte-Carlo Simulations, Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 138–144 (2014).

[6] Chen, W., Wang, C., and Wang, Y.: Scalable Influence Maximization for Prevalent Viral Marketing in Large-scale Social Networks, Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1029–1038 (2010).

[7] Cohen, E., Delling, D., Pajor, T., and Werneck, R. F.: Sketch-based Influence Maximization and Computation: Scaling Up with Guarantees, Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, pp. 629–638 (2014).

[8] Goldenberg, J., Libai, B., and Muller, E.: Talk of the Network: A Complex Systems Look at the Underlying Process of Word-of-mouth, Marketing Letters, Vol. 12, No. 3, pp. 211-223 (2001).

[9] Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L.: An analysis of the approximations for maximizing submodular set functions, Mathematical Programming, Vol. 14, No. 1, pp. 265–294 (1978).

[10] Valiant, L. G.: A Bridging Model For Parallel Computation, Communications of the ACM, Vol. 33, No. 8, pp. 103–111 (1990).

[11] Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G.: Pregel: A System for Large-scale Graph Processing, Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data,pp. 135–146 (2010).

[12] Gonzalez, J. E., Low, Y., Gu, H., Bickson, D., and Guestrin, C.: PowerGraph: Distributed Graph-parallel Computation on Natural Graphs, Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation, pp. 17–30 (2014).