DEIM Forum 2012 C1-3

Affinity Propagation のための高速化手法

藤原　靖宏† 　入江　豪†† 　北原　友恵†††

† ＮＴＴサイバースペース研究所　〒 230–0847 神奈川県横須賀市光の丘 1-1
†† ＮＴＴサイバーソリューション研究所　〒 230–0847 神奈川県横須賀市光の丘 1-1
††† 日本大学大学院理工学研究科　〒 101–8308 東京都千代田区神田駿河台 1-8-14

あらまし　Affinity Propagation は Frey らによって提案されたクラスタリング手法である．Affinity Propagation は k-means 法などに代表される既存のクラスタリング手法よりクラスタリング精度が良いため，近年様々な分野において用いられている．オリジナルの Affinity Propagation では精度の良いクラスタリング結果を得るため全てのデータポイント間でメッセージと呼ばれる値を繰返し収束するまで計算する．しかしこのオリジナルの手法はデータポイントの数の２乗の計算コストを要するため，データポイントの数が多い場合は非常に計算時間がかかるという問題点がある．本論文では収束後においてオリジナルの Affinity Propagation とクラスタリング結果が同じになることを保証する高速化手法を提案する．提案手法は（１）収束値を計算するのに不必要なデータペアを枝刈りするアイデアと，（２）枝刈りされたデータペアの収束値を枝刈りされなかったデータペアの収束値から計算するアイデアから構成される．実データを用いて比較実験を行い，提案手法はオリジナルの手法より高速にクラスタリングを行えることを確認した．

キーワード　クラスタリング，Affinity Propagation，高速化

Efficient Approach for Affinity Propagation

Yasuhiro FUJIWARA†, Go IRIE††, and Tomoe KITAHARA†††

† NTT Cyber Space Laboratories, 1-1 Hikarinooka, Yokosuka, Kanagawa, 230–0847 Japan
†† NTT Cyber Solution Laboratories, 1-1 Hikarinooka, Yokosuka, Kanagawa, 230–0847 Japan
††† Graduate School of Science and Technology, Nihon University, Kandasurugadai 1-8-14, Chiyoda, Tokyo 101–8308 Japan

1. はじめに

クラスタリングはデータの解析における汎用的なツールの１つである [1]．そのため今まで多くのクラスタリング手法が提案されている．クラスタリングはデータの集合が与えられたとき，それらをクラスタに分割することが目的である．クラスタリングを行うことにより，データ集合における代表的なデータを見つけることなどができる．

近年 Frey らによって Affinity Propagation というクラスタリング手法が提案された [2]．Affinity Propagation はデータポイントの間でメッセージと呼ばれる値を再帰的に収束するまで計算する．そして収束したメッセージの値から exemplar と呼ばれるクラスタを代表するデータを求める．Affinity Propagation は今までの k-means 法 [1] などのクラスタリング手法と比較して，（１）クラスタリングの誤差が少ない，（２）非対称で三角不等式の成り立たない類似度に対応する，（３）クラスタリング結果が初期状態に依存しないなどのメリットがある．そのため Affinity Propagation はアクティブ・ラーニング [3]，キーフレーズの抽出 [4]，画像のクラスタリング [5] [6]，代表画像の抽出 [7] などの多くのアプリケーションにおいて用いられている．

しかし Affinity Propagation の１つの弱点として計算時間が挙げられる [5]．すなわちデータポイントの数が多くなった場合，Affinity Propagation によりクラスタリングを行うには非常に計算時間を要する．Affinity Propagation において再帰的にメッセージの値を計算するには，データポイントの２乗の計算コストを要するためである．この問題を解くために Jia らは近年 FSAP を提案した [5]．しかし FSAP は高速なクラスタリングと引き替えに，クラスタリングの結果がオリジナルの Affinity Propagation の結果と異なるという弱点がある．高速なクラスタリングと引き換えにクラスタリング結果を犠牲にするのはアプリケーションの観点から好ましくない．

本論文では Affinity Propagation のための高速化手法を提案

する．提案手法はメッセージが収束した後のクラスタリング結果が，オリジナルの手法のクラスタリング結果と同じになるという性質を持つ．クラスタリング結果の精度を保ちつつ計算コストを減らすために提案手法では，（1）収束値を計算するのに不必要なデータペアを枝刈りして高速に収束値を計算し，（2）枝刈りされたデータペアの収束値を枝刈りされなかったデータペアの収束値から計算する．従来手法の FSAP では高速にクラスタリングするために内部パラメータの設定が必要であるが，内部パラメータの設定によっては適切なクラスタリング結果が得られないことがある．しかし提案手法は FSAP とは異なり，内部パラメータを設定する必要がない．すなわち Affinity Propagation は複数の内部パラメータの設定が必要であるが，提案手法は Affinity Propagation の内部パラメータを外部パラメータとし，提案手法によって新たに必要になる内部パラメータはない．本論文では複数の実データを用いて検証を行い提案手法の有効性を確認した．この結果は提案手法が実際のアプリケーションにおいて有用であることを示している．

本論文の構成は以下の通りである．2. 章で関連研究について述べる．3. 章で Affinity Propagation と従来手法である FSAP の詳細な説明を行う．4. 章で提案手法の詳細を説明する．5. 章で実験結果を示す．6. 章で結論を述べる．

2. 関 連 研 究

Affinity Propagation は k-means 法などよりクラスタリング精度の良い手法であり，様々な分野のアプリケーションで応用されている．本論文で提案する手法を利用する事により，これらのアプリケーションを高速に処理することができる．

アクティブ・ラーニングは文書のラベリングを効率化する目的でテキスト分類において用いられている [8]．アクティブ・ラーニングにおいて重要な問題の 1 つとして，機械学習の初期の訓練データとしてどのサンプル文書を用いるのかというものがある．これは初期の訓練データによりアクティブ・ラーニングの性能が大きく影響されるからである．初期の訓練データを選択する方法として k-means 法や k-medoids 法が有名である [9], [10]．Hu らはこれらの手法の代わりに Affinity Propagation を初期の訓練データを選択するアプローチを提案した [3]．彼らは実験により k-means 法より Affinity Propagation を用いた手法の方が良い結果になることを示した．

キーフレーズは文書の要約をしたものとして広く使われており，一般的に記事の著者によって付けられるものである．しかしウェブにおける記事の多くは人手によってキーフレーズが付けられていなく，さらにこのような要約情報のないウェブ文章は爆発的に増加している．そのため自動的にウェブ文章に要約を付与することが 1 つの重要な研究テーマになっている [11]．Liu らは要約情報の付与に Affinity Propagation を用いることを提案した [4]．彼らの手法ではまず候補の単語をクラスタに分け，exemplar になる単語を抽出する．そして exemplar を用いて文書の要約を付与する．彼らは実験により彼らの手法が有名な TextRank による手法より優れていることを示した [12]．

近年，インターネット技術の進展に伴い，サーチエンジンを用いて非常の多くの画像データを検索することができる．多くのサーチエンジンではユーザからの問い合わせに対して画像データのランキングをして検索結果を返しているが，ユーザは所望の画像を探すときに検索結果をひとつひとつ確認する必要がある．ユーザが容易に検索結果を閲覧するため，検索された画像をクラスタリングして提示すること手法が様々提案されている [13], [14]．Jia らは Affinity Propagation を用いて画像のクラスタリングを行い，検索結果として提示する方法を提案した [5]．また彼らは同時に 3.2 章で示す Affinity Propagation の高速化手法も併せて提案し，Affinity Propagation の手法を用いるよりも高速に検索結果を提示できることを示した．

表 1　主な記号とその定義

記号	定義
N	データポイントの数
T	繰返しの計算回数
$s[i,j]$	データポイント i と j の間の類似度
$r[i,j]$	データポイント i と j の間の responsibility
$a[i,j]$	データポイント i と j の間の availability
$\rho[i,j]$	データポイント i と j の間の responsibility の伝搬値
$\alpha[i,j]$	データポイント i と j の間の availability の伝搬値
λ	ダンピングファクタ $0 \leqq \lambda < 1$

3. 前　準　備

この章では Frey らによって提案された Affinity Propagation [2] とその最新の高速化手法である FSAP [5] について説明する．表 1 に主な記号とその定義を示す．

3.1 Affinity Propagation

Affinity Propagation はデータをクラスタリングし，クラスタを代表する exemplar を求めるクラスタリング手法である．Affinity Propagation の入力はデータペアの類似度 $s[i,j](i,j = 1,2,\ldots,N)$ である(注1)．ここで類似度に対する制約はない．たとえば負のユークリッド距離でもよいし，Jaccard 係数などでもよい．

与えられた類似度に対して Affinity Propagation は exemplar とその他のデータポイントの間の類似度を最大化するようにクラスタリングを行う．Affinity Propagation では responsibility と availability と呼ばれるメッセージを全てのデータペアに対して再帰的に計算する．responsibility $r[i,j]$ はデータポイント i から j のメッセージで，データポイント j がデータポイント i の exemplar としてどれほど適切であるかを表す値である．availability $a[i,j]$ はデータポイント j から i のメッセージで，データポイント i がデータポイント j の exemplar として選ぶことがどれほど適切であるかを表す値である．responsibility と availability ははじめ 0 に初期化され，これらの値は以下の式により再帰的に収束するまで計算される．

$$\begin{aligned} r[i,j] &= (1-\lambda)\rho[i,j] + \lambda r[i,j] \\ a[i,j] &= (1-\lambda)\alpha[i,j] + \lambda a[i,j] \end{aligned} \tag{1}$$

(注1)：ここで $s[i,i](i,j = 1,2,\ldots,N)$ は preference と呼ばれる値で，preference はクラスタリング結果のクラスタ数に影響する．

ここで λ は繰り返し計算の中で responsibility と availability が振動するのを防ぐためのダンピングファクタと呼ばれるものである．また $\rho[i,j]$ と $\alpha[i,j]$ はそれぞれ各繰返し計算における responsibility と availability の伝搬値である．$\rho[i,j]$ と $\alpha[i,j]$ は以下の式から計算する．

$$\rho[i,j]=\begin{cases}s[i,j]-\max_{k\neq j}\{a[i,k]+s[i,k]\} & (i\neq j)\\ s[i,j]-\max_{k\neq j}\{s[i,k]\} & (i=j)\end{cases} \quad (2)$$

$$\alpha[i,j]=\begin{cases}\min\{0,r[j,j]+\sum_{k\neq i,j}\max\{0,r[k,j]\}\} & (i\neq j)\\ \sum_{k\neq i}\max\{0,r[k,j]\} & (i=j)\end{cases} \quad (3)$$

データポイント i の exemplar は収束値の計算の後，以下の式から計算する．

$$\arg\max\{r[i,j]+a[i,j]:j=1,2,\ldots,N\} \quad (4)$$

ここで N と T をそれぞれデータポイントの数と再帰的計算の繰り返し回数としたときに，オリジナルの Affinity Propagation における計算コストは $O(N^2T)$ となる．この計算コストはデータポイントの数の２乗に比例するため，データポイントの数が多くなると非常に計算時間を要するという問題点がある．

3.2 FSAP

Affinity Propagation における計算コストを削減することを目的に，近年 Jia らによって FSAP が提案された [5]．Affinity Propagation の繰返し計算において値を計算するメッセージの数を減らすというのが彼らのアイデアである．彼らの手法はこのアイデアに基づき，はじめに類似度の K 近傍を用いて疎なグラフ構造を構築する．すなわちデータポイント i がデータポイント j に対して K 個以内の高い類似度を持つのであれば，データポイント i と j の間にエッジを張る．FSAP ではまずこのように構築されたグラフのエッジにのみメッセージの収束値を計算するため，その結果として多く（少なくとも N/K 個）のデータポイントが exemplar となる．そのため FSAP では複数の exemplar を１つのクラスタにまとめるために，以下の３つの条件に基づき構築したグラフに対して新たにエッジを追加し，再度メッセージの収束値を計算する．

（1） データポイント i がデータポイント j の exemplar であれば，データポイント i と j の間にエッジを張る．

（2） データポイント i と j に対して，もしデータポイント i と j をそれぞれ exemplar とするデータポイントト m と n があり，さらにデータポイントト m と n がそれぞれ K 近傍であれば，データポイント i と j の間にエッジを張る．

（3） ２番目の条件でデータポイント i と j の間にエッジを張った場合，さらにデータポイント i または j を exemplar とするデータポイントとデータポイント i または j の間にエッジを張る．

FSAP は最後にメッセージの収束値を計算した後に，式 (4) に基づき exemplar を計算する．

彼らは FSAP がオリジナルの手法と比較して高速にクラスタリングを行えることを示した．しかし彼らの手法は類似度の K 近傍を用いてグラフ構造を構築するという経験的なものであり，クラスタリング結果がオリジナルの手法と同じになることを保証していない．後の章で示す通り，このことはクラスタリング結果に大きく影響を与える．本論文では FSAP よりさらに高速でしかも収束後においてオリジナルの手法とクラスタリング結果が同じになる事を保証する手法を提案する．

4. 提 案 手 法

この章では収束後においてオリジナルの手法とクラスタリング結果が同じになることを保証する Affinity Propagation の高速化手法を述べる．まずはじめに提案手法の概要について述べた後で手法の詳細を示す．またこの章では提案手法についての理論的な解析も示す．

4.1 基本的アイデア

クラスタリングの速度を上げるために提案手法では疎なグラフ構造 $G=[V,E]$ を作る．ここで V はデータポイントに対応し，E はデータペアの一部とする．この構造を用いることにより収束値を高速に計算できる．これは responsibility と availability の計算がグラフのエッジのみに限定されるからである．しかし類似度の K 近傍を用いて単純に疎なグラフ構造を構築するような手法であればクラスタリング結果がオリジナルの手法と同じになることを保証できないという問題がある．

この問題に対して提案手法では，収束時における responsibility と availability の上限値と下限値を推定し不要な計算を枝刈りする．さらに後に述べるように枝刈りされたメッセージの収束値は，枝刈りされなかったメッセージの収束値から計算できるという性質がある．そのため提案手法では枝刈りされたグラフ構造を用いて収束値を求めたあとに，枝刈りされたメッセージの収束値を求めるというアプローチをとる．結果として提案手法はオリジナルの手法と同じクラスタリング結果を得ることができる．

提案手法は FSAP と比較してさらに内部パラメータを設定する必要がないという特徴がある．すなわち FSAP は疎なグラフ構造を構築するために K を設定する必要があるが，提案手法にはこのようなパラメータは存在しない．これは上限値と下限値に内部パラメータは必要ないからである．

4.2 アプローチの詳細

ここではまず疎なグラフ構造の構築方法とそのグラフにおける理論的な特徴について述べる．そして具体的なクラスタリングのアルゴリズムを示す．

4.2.1 疎なグラフ構造

提案手法ではメッセージの収束値を高速に計算するために繰返し計算を行うメッセージの数を減らす．高速に計算するために提案手法では疎なグラフ構造 $G=[V,E]$ （V はデータポイントで，E はデータペアの一部）を構築する．提案手法では収束時におけるメッセージの値の上限値と下限値を推定し，不要な繰返し計算を枝刈りする．まず収束時におけるメッセージの値における以下の補助定理を示す．

［**補助定理** 1］（メッセージの収束値）収束後 responsibility と availability の値はそれぞれ $r[i,j]=\rho[i,j]$ と $a[i,j]=\alpha[i,j]$ となる．

証明 式 1 より明らか． □

この補助定理は Affinity Propagation においてメッセージの値が収束していれば，それらの収束値は収束時におけるメッセージの伝搬値（式 (2) と式 (3)）から求められることを示している．

この補助定理を用いることにより responsibility と availability の上限値と下限値を類似度の値のみから計算できる．定式的には上限値と下限値を類似度を用いて以下のように計算する．

［**定義** 1］（上限値と下限値）収束時における上限値と下限値，$\underline{a}[i,j]$ と $\overline{r}[i,j]$ と $\overline{a}[i,j]$ を以下のように計算する．

$$\underline{a}[i,j]=\begin{cases}\min\{0,r[j,j]\} & (i\neq j)\\ 0 & (i=j)\end{cases} \tag{5}$$

$$\overline{r}[i,j]=\begin{cases}s[i,j]-\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\} & (i\neq j)\\ s[i,j]-\max_{k\neq j}\{s[i,k]\} & (i=j)\end{cases} \tag{6}$$

$$\overline{a}[i,j]=\begin{cases}\min\{0,\overline{r}[j,j]+\sum_{k\neq i,j}\max\{0,\overline{r}[k,j]\}\} & (i\neq j)\\ \sum_{k\neq i}\max\{0,\overline{r}[k,j]\} & (i=j)\end{cases} \tag{7}$$

なおここで $\underline{a}[i,j]$ はデータペアの類似度 s からのみ計算できる．これは $r[j,j]=s[j,j]-\max_{k\neq j}\{s[j,k]\}$ (式 (2)) だからである．さらに定義から $\overline{r}[i,j]$ と $\overline{a}[i,j]$ はそれぞれ $\underline{a}[i,j]$ と $\overline{r}[i,j]$ から計算できることがわかる．すなわち上限値と下限値はすべてデータペアの類似度 s からのみ計算でき，これらを計算するのに繰り返し計算は必要ないことがわかる．

ここで $\underline{a}[i,j]$ と $\overline{r}[i,j]$ と $\overline{a}[i,j]$ の性質を示すために以下の補助定理を示す．

［**補助定理** 2］（上限値と下限値）グラフ G における任意のデータポイント i と j において，$\underline{a}[i,j]\leqq a[i,j]$, $\overline{r}[i,j]\geqq r[i,j]$, と $\overline{a}[i,j]\geqq a[i,j]$ が収束後に成り立つ．

証明　まず $\underline{a}[i,j]\leqq a[i,j]$ となることを示す．補助定理 1 から収束後に $a[i,j]=\alpha[i,j]$ となる．もし $i\neq j$ であれば $a[i,j]\geqq\min\{0,r[j,j]\}=\underline{a}[i,j]$ となる．これは $\sum_{k\neq i,j}\max\{0,r[k,j]\}\geqq 0$ だからである．もし $i=j$ であれば $\underline{a}[i,j]\leqq a[i,j]$ となる．これは $\sum_{k\neq i}\max\{0,r[k,i]\}\geqq 0$ だからである．次に $\overline{r}[i,j]\geqq r[i,j]$ であることを示す．これは補助定理 1 と式 (2) と $\underline{a}[i,j]\leqq a[i,j]$ から明らかである．最後に $\overline{a}[i,j]\geqq a[i,j]$ であることを示す．これは補助定理 1 と式 (3) と $\overline{r}[i,j]\geqq r[i,j]$ から明らかである．□

提案手法では上限値と下限値を用いて繰返し計算において不要なメッセージを枝刈りし，疎なグラフ構造を作る．定式的にはグラフのエッジは以下の定義に従って作られる．

［**定義** 2］（グラフ構造）V がデータポイントに対応するグラフ G において，以下の条件を満たすときのみノード i と j の間にエッジを張る．

$$(1)\overline{r}[i,j]\geqq 0,\ \text{or}$$
$$(2)\overline{a}[i,j]+s[i,j]\geqq\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\} \tag{8}$$

このグラフ構造における性質を示すために以下の 2 つの補助定理を示す．

［**補助定理** 3］（availability の収束値）任意のノードペア i と j において availability の収束値は $\overline{r}[i,j]\geqq 0$ の条件を満たす responsibility のみを用いて計算できる．

Algorithm 1 提案手法のアルゴリズム

Input: データペアの類似度
Output: 各データポイントに対する exemplar

for それぞれのデータペア $[i,j]$ に対して **do**
　$\underline{a}[i,j]$ と $\overline{r}[i,j]$ と $\overline{a}[i,j]$ を式 (5)〜(7) から計算;
end for
for それぞれのデータペア $[i,j]$ に対して **do**
　if $\overline{r}[i,j]\geqq 0$ or $\overline{a}[i,j]+s[i,j]\geqq\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\}$**then**
　　データペア $[i,j]$ をエッジでつなぐ;
　end if
end for
for $t=1$ to T **do**
　for それぞれのエッジ $[i,j]$ に対して **do**
　　$r[i,j]$ と $a[i,j]$ を式 (1) を用いて更新する;
　end for
end for
for エッジのないデータペア $[i,j]$ それぞれに対して **do**
　$r[i,j]=\rho[i,j]$ と $a[i,j]=\alpha[i,j]$ を計算する;
end for
for それぞれのデータポイント i に対して **do**
　exemplar を式 (4) から計算する;
end for

証明　補助定理 1 から収束後 $a[i,j]=\alpha[i,j]$ となる．もし $\overline{r}[i,j]<0$ があるノードペアに成り立てば，収束後にそのノードペアの responsibility の値は式 (3) における $\sum_{k\neq i,j}\max\{0,r[k,j]\}$ と $\sum_{k\neq i}\max\{0,r[k,j]\}$ の値には影響しない．□

［**補助定理** 4］（responsibility の収束値）任意のノードペア i と j において responsibility の収束値は $\overline{a}[i,j]+s[i,j]\geqq\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\}$ の条件を満たす availability のみを用いて計算できる．

証明　補助定理 1 から収束後 $r[i,j]=\rho[i,j]$ となる．もし $i\neq j$ であれば，式 (2) より収束後に $r[i,j]=s[i,j]-\max_{k\neq j}\{a[i,k]+s[i,k]\}$ となる．$\max_{k\neq j}\{a[i,k]+s[i,k]\}\geqq\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\}$ であるため，もし $\max_{k\neq j}\{\underline{a}[i,k]+s[i,k]\}>\overline{a}[i,j]+s[i,j]$ が成り立てば，そのノードペアに対して $\max_{k\neq j}\{a[i,k]+s[i,k]\}>\overline{a}[i,j]+s[i,j]$ となる．そのためそのノードペアの availability の値は responsibility の値に影響しない．またもし $i=j$ であれば responsibility の値は式 (2) より類似度のみから計算できる．□

補助定理 3 と 4 から以下の補助定理が成り立つ．

［**補助定理** 5］（グラフ構造における収束値）任意のノードペア i と j においてグラフ G を用いてメッセージの収束値は正確に計算できる．

証明　補助定理補助定理 3 と 4 から明らか．□

補助定理 5 によって収束後において提案手法がオリジナルの手法とクラスタリング結果が同じになる事を示すことができる．

4.2.2　クラスタリングアルゴリズム

Algorithm 1 に提案手法のアルゴリズムを示す．提案手法ではまず定義 1 に基づきメッセージの上限値と下限値を計算す

る（1〜3 行目）．そして定義 2 を用いて疎なグラフ構造を作る（4〜8 行目）．グラフのエッジに対応するメッセージの値から全てのデータペアのメッセージの収束値を計算できるため（補助定理 5），グラフのエッジに対して収束値を繰返し計算で求める（9〜13 行目）．そして繰返し計算の後で枝刈りしたメッセージに対して収束値を計算する（14〜16 行目）．これらの値はメッセージの伝搬値から計算することができる（補助定理 1）．最後に収束値から exemplar を計算する（17〜19 行目）．

提案手法におけるアルゴリズムでは内部パラメータを必要としない．そのため実際のアプリケーションにおいて提案手法を用いて容易に Affinity Propagation の高速化を行うことができる．

4.2.3 理論的解析

提案手法のクラスタリング結果と計算コストについての理論的な解析を行う．なおここで M をグラフにおけるエッジの数とする．すなわち $M = |E|$ である．

[**定理** 1]（クラスタリング結果）収束後において，提案手法のクラスタリング結果はオリジナルの Affinity Propagation のクラスタリング結果と同じになる．

証明 補助定理 5 より明らか． □

[**定理** 2]（計算コスト）提案手法における計算コストは $O(N^2 + MT)$ である．

証明 提案手法においてはまず全てのデータペアに対して上限値と下限値を計算し，グラフ構造を構築する．上限値と下限値はデータペアの類似度からのみ求められるので，これらには $O(N^2)$ の計算コストを要する．そしてグラフの全てのエッジに対してその収束値を繰返し計算で求める．これには $O(MT)$ の計算コストを要する．そして最後にエッジを張っていないノードペアに対して収束値を計算し，最後に exemplar を計算する．これらにはそれぞれ $O(N^2 - M)$ と $O(N^2)$ の計算コストを要する．そのため提案手法における計算コストは $O(N^2 + MT)$ である． □

5. 評価実験

提案手法の有効性を確認するためにオリジナルの手法 [2] と FSPA [5] と比較実験を行った．実験により以下項目について確認した．

- 高速性: 提案手法はオリジナルの手法と FSAP によりクラスタリングを高速に行える（5.1 章）．
- 正確性: FSAP と異なり提案手法はオリジナルの手法と同じクラスタリング結果を得られる（5.2 章）．
- 有用性: 提案手法と FSAP のクラスタリング結果は異なり，提案手法のクラスタリング結果は適切である（5.3 章）．

この章において *Proposed* と *Original* はそれぞれ提案手法とオリジナルの手法の結果を示している．また FSPA においては K の値を複数設定して検証を行った．*FSAP(5%)* と *FSAP(10%)* と *FSAP(20%)* はそれぞれ K の値を $0.05N$，$0.1N$，$0.2N$ としたときの結果を示している．実験ではメッセージの収束値を 1,000 回繰返し計算して求めた．またダンピングファクタの値 λ は 0.5 に設定した．

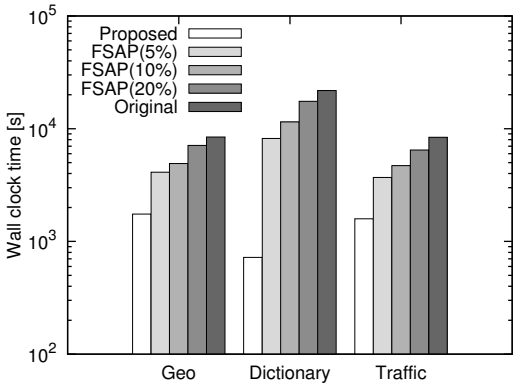


図 1 クラスタリングの処理時間

実験では以下に示すように 3 つのデータセットを用いた．

- *Geo*: このデータは地理情報を持つ写真を Flickr (注2) から API (注3) を用いて収集したものである．このデータは 10,000 個の写真から構成され，全ての写真はニューヨークの中心街で 2006 年の 1 月 1 日から 2009 年の 6 月 30 日までに撮られた．写真が撮られた場所は緯度と経度（x と y）で表現され，写真 i と j の類似度は負のユークリッド距離として定義される．
- *Dictionary* (注4): このデータは FOLDOC (注5) における単語のネットワークである．FOLDOC はオンラインで利用できるコンピュータ分野の辞書である．このネットワークにおいて，単語 i から j へのエッジの重みは単語 i の説明に単語 j が使われた頻度である．単語間の類似度は random walk with restart [15] を用いて計算した．単語の数は 13,356 である．
- *Traffic* (注6): これは高速道路における交通量を計測したセンサデータである．このデータは 5 分おきにロサンゼルスにおいて 2005 年 4 月 10 日から 2005 年 10 月 1 日までに得られたものである．実験では 10,000 個の時系列データを取り出し，類似度を負のユークリッド距離として計算した．

実験は CPU が Intel Xeon Quad-Core 3.33GHz，メモリが 32GB の Linux サーバで行った．またすべてのアルゴリズムは GCC で実装した．

5.1 高速性

それぞれの手法に対してクラスタリングを行うのに必要な処理時間を評価した．図 1 に結果を示す．また図 2 にデータセット *Geo* に対してデータポイントの数を変えた場合の各手法のスケーラビリティを示す．

これらの図からオリジナルの手法と比較して提案手法が最大 30 倍高速にクラスタリングを行えることがわかる．これは 3. 章で述べたとおりオリジナルの手法は $O(N^2T)$ の計算コストが必要であるが，提案手法は定理 2 の証明の中で述べたように $O(MT)$ のコストで繰返し計算を行え，表 2 に示すように提案

(注2)：http://www.flickr.com/
(注3)：http://maps.google.com/
(注4)：http://vlado.fmf.uni-lj.si/pub/networks/data/dic/foldoc/foldoc.zip
(注5)：http://foldoc.org/
(注6)：http://archive.ics.uci.edu/ml/datasets/Dodgers+Loop+Sensor

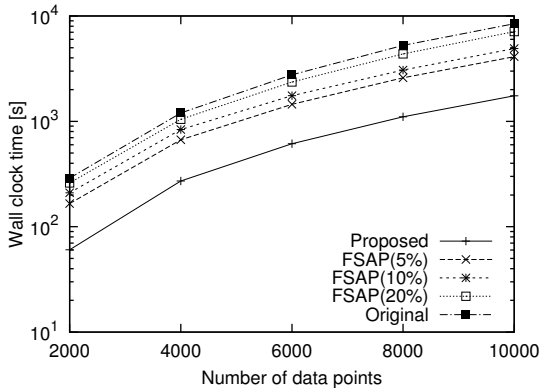


図 2　スケーラビリティ

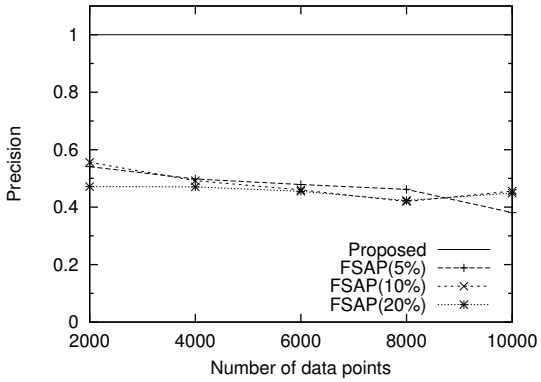


図 3　適　合　率

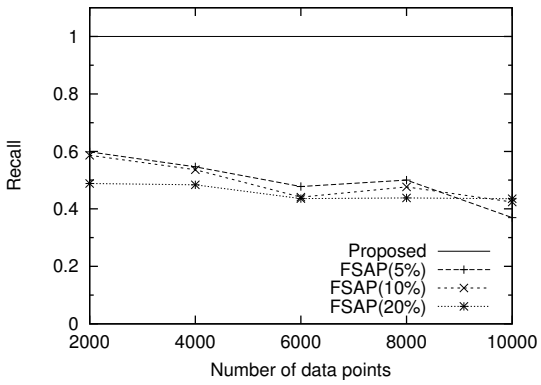


図 4　再　現　率

表 2　提案手法とオリジナルの手法におけるエッジの数

手法	データセット		
	Geo	*Dictionary*	*Traffic*
Proposed	10,009,002	2,799,319	10,034,228
Original	100,000,000	178,382,736	100,000,000

手法におけるグラフは非常に疎な構造であるからである．

また実験では複数の K の値を用いて検証を行ったが，提案手法は FSAP より大幅に高速にクラスタリングを行えた．FSAP は 3.2 章で述べたとおり，まず K 近傍を用いて構築したグラフにおける収束値を計算し，クラスタリング結果からさらにグラフのエッジを追加し再度収束値を計算し直すアプローチである．すなわち FSAP では 2 回収束値を計算する必要があるため，提案手法は FSAP より高速にクラスタリングを行うことができた．

5.2　正　確　性

従来手法の FSAP と比較して，提案手法の 1 つの利点として収束後においてオリジナルの手法と同じクラスタリング結果になることが挙げられる．この実験ではどれほど FSAP がオリジナルの手法と同じクラスタリング結果になるのかの検証を行った．

クラスタリングの精度を調べるために，実験では提案手法と FSAP から得られる examplar とオリジナルの手法から得られる examplar を比較した．クラスタリングの精度としては適合率と再現率を調べた．適合率とはオリジナルの手法で得られた examplar のなかから各手法から得られた examplar の割合である．再現率とは各手法から得られた examplar のなかからオリジナルの手法でも得られた examplar の割合である．実験では *Geo* をデータセットとして用いた．

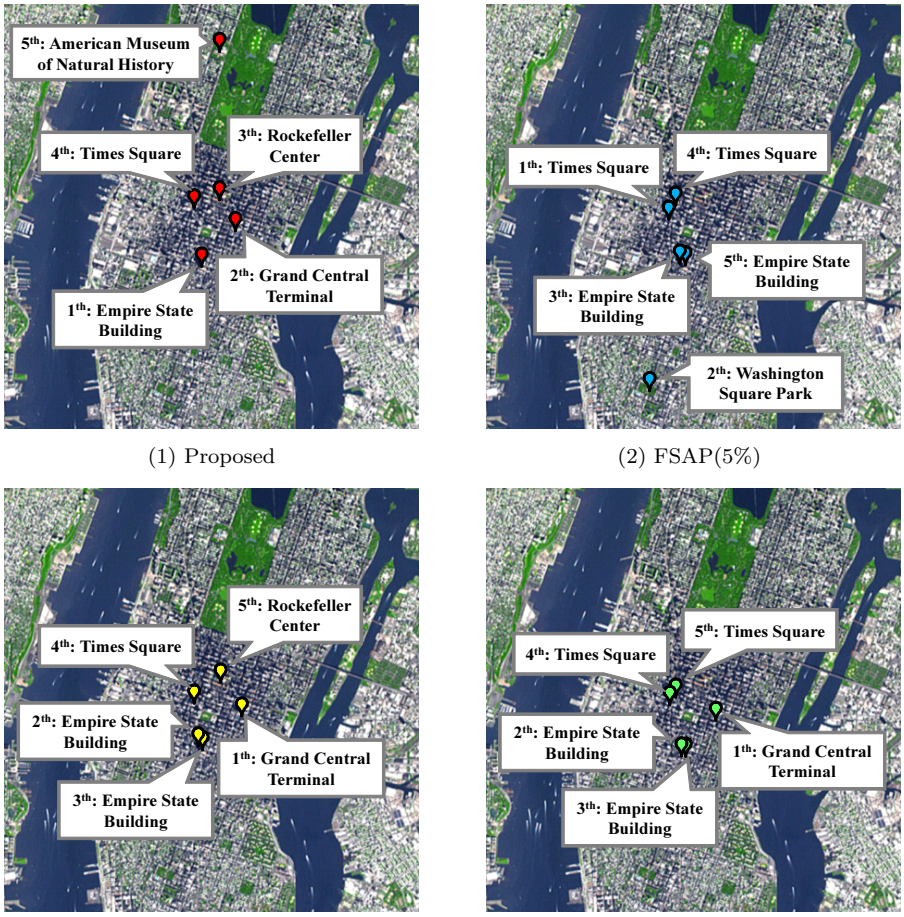
図 3 に適合率を，図 4 に再現率を示す．これらの図からわかるとおり，提案手法の適合率と再現率はつねに 1 である．これは提案手法が理論的にメッセージの収束後において収束値を正しく計算できるように設計されているため，高速化においてクラスタリングの精度を犠牲にしないからである．一方 FSAP においては適合率と再現率が提案手法より低い結果となった．これは 3.2 章で述べたとおり，FSAP は経験的な手法により疎なグラフ構造の構成するからである．図 1〜4 からわかるとおり，提案手法はオリジナル手法より高速であり，また FSAP より高速でクラスタリングの精度が良いことがわかる．

5.3　有　用　性

実際のアプリケーションにおいて提案手法が有用であることを示すために *Geo* をデータセットにしたときのクラスタリング結果を示す．緯度と経度の情報をもつ写真をクラスタリングすることにより，どの場所で多くの写真が撮られたかを検出することができ，これは位置情報による写真の閲覧 [16] やランドマーク写真の検出 [17] などのアプリケーションに用いられる．

図 5 は提案手法と FSAP により得られた 5 つの最も大きなクラスタを示す．各 examplar の位置の名前を得るために公開されている位置情報付きウィキペディアのツール (注7) を用いた．各手法によりエンパイア・ステート・ビルディングやタイムズスクウェアやグランド・セントラル駅などのニューヨークの中心街における有名なランドマークを検出できた．しかし FSAP は同じランドマークを一度に複数検出した．例えば FSAP(5%) においてはエンパイア・ステート・ビルディングとタイムズスクウェアがそれぞれ 3 番目と 5 番目，1 番目と 4 番目に 2 回検出されている．一方，提案手法ではそれぞれ異なるランドマークが検出できている．これらの結果は提案手法が実際のアプリケーションにおいても有効に利用できることを示している．

(注7)：http://www.geonames.org/export/wikipedia-webservice.html



(1) Proposed　(2) FSAP(5%)

(3) FSAP(10%)　(4) FSAP(20%)

図 5　各手法により得られたニューヨークの中心街における 5 つのランドマーク

6. ま と め

この論文では収束後においてオリジナルの手法と同じクラスタリング結果になることを保証する Affinity Propagataion の高速化手法を提案した．提案手法は（1）繰返し計算において不要なメッセージの計算を枝刈りし，（2）枝刈りしたメッセージの収束値を枝刈りしなかったメッセージの収束値から計算するアイデアから構成される．提案手法について検証を行ったところ，提案手法はクラスタリングの精度を犠牲にすることなく，高速にクラスタリングを行えることを確認した．Affinity Propagataion は様々な分野で用いられているクラスタリング手法であるが，本論文で提案した高速化手法を利用する事により様々なアプリケーションを高速に行うことができる．

文　献

[1] A. K. Jain, M. N. Murty and P. J. Flynn: “Data clustering: A review”, ACM Comput. Surv., **31**, 3, pp. 264–323 (1999).

[2] B. J. Frey and D. Dueck: “Clustering by passing messages between data points”, Science, **315**, p. 2007 (2007).

[3] R. Hu, B. M. Namee and S. J. Delany: “Off to a good start: Using clustering to select the initial training set in active learning”, Florida Artificial Intelligence Research Society Conf. (FLAIRS) (2010).

[4] Z. Liu, P. Li, Y. Zheng and M. Sun: “Clustering to find exemplar terms for keyphrase extraction”, Conf. Empirical Methods in Natural Language Processing (EMNLP), pp. 257–266 (2009).

[5] Y. Jia, J. Wang, C. Zhang and X.-S. Hua: “Finding image exemplars using fast sparse affinity propagation”, ACM Int. Conf. Multimedia (ACM MM), pp. 639–642 (2008).

[6] D. Dueck and B. J. Frey: “Non-metric affinity propagation for unsupervised image categorization”, IEEE Int. Conf. Computer Vision (ICCV), pp. 1–8 (2007).

[7] Z.-J. Zha, L. Yang, T. Mei, M. Wang and Z. Wang: “Visual query suggestion”, ACM Int. Conf. Multimedia (ACM MM), pp. 15–24 (2009).

[8] D. A. Cohn, L. E. Atlas and R. E. Ladner: “Improving generalization with active learning”, Machine Learning, **15**, 2, pp. 201–221 (1994).

[9] J. Zhu, H. Wang, T. Yao and B. K. Tsou: “Active learning with sampling by uncertainty and density for word sense disambiguation and text classification”, COLING, pp. 1137–1144 (2008).

[10] H. T. Nguyen and A. W. M. Smeulders: “Active learning using pre-clustering”, ICML (2004).

[11] M. P. Grineva, M. N. Grinev and D. Lizorkin: “Extracting key terms from noisy and multitheme documents”, WWW, pp. 661–670 (2009).

[12] R. Mihalcea and P. Tarau: “TextRank: Bringing order into

texts", EMNLP (2004).

[13] D. Cai, X. He, Z. Li, W.-Y. Ma and J.-R. Wen: "Hierarchical clustering of www image search results using visual, textual and link information", ACM Multimedia, pp. 952–959 (2004).

[14] Y. Hu, N. Yu, Z. Li and M. Li: "Image search result clustering and re-ranking via partial grouping", ICME, pp. 603–606 (2007).

[15] H. Tong, C. Faloutsos and J.-Y. Pan: "Random walk with restart: fast solutions and applications", Knowl. Inf. Syst., **14**, 3, pp. 327–346 (2008).

[16] D. J. Crandall, L. Backstrom, D. Huttenlocher and J. Kleinberg: "Mapping the world's photos", Int. World Wide Web Conf. (WWW), pp. 761–770 (2009).

[17] L. S. Kennedy and M. Naaman: "Generating diverse and representative image search results for landmarks", Int. World Wide Web Conf. (WWW), pp. 297–306 (2008).