

## Locality-Sensitive Hashing を用いた階層的クラスタ解析手法の高速化

石橋 徹夫, 古賀 久志, 渡辺 俊典, 菅原 研

電気通信大学 大学院 情報システム学研究科

〒182-8585 東京都調布市調布ヶ丘 1-5-1

E-mail {ishib-te, koga, watanabe, sugawara}@sd.is.uec.ac.jp

### 概要

階層的クラスタ解析手法は類似度でデータを階層的に分類し、その結果は樹形図で表現することができる。この手法を用いると細かい分類から大まかな分類までクラスタ間の包含関係が理解しやすいが、計算量は大きなものとなるので、高次元・大規模データに対して適用することは難しい。そこで本研究では最近接点の候補を高速に見つけるアルゴリズムである Locality-Sensitive Hashing によって作られるハッシュテーブルを用いて、計算量を減らし高速な近似階層的クラスタリングを実現する。

**キーワード** 階層的クラスタリング Locality-Sensitive Hashing

## Approximate Hierarchical Clustering Algorithm Using Locality-Sensitive Hashing

Tetsuo ISHIBASHI, Hisashi KOGA, Toshinori WATANABE, Ken SUGAWARA

Graduate School of Information Systems, University of Electro-Communications

Chofugaoka 1-5-1, Chofu city, Tokyo, 182-8585 JAPAN

E-mail {ishib-te, koga,watanabe, sugawara}@sd.is.uec.ac.jp

### Abstract

The hierarchical clustering techniques classify data by similarity and their results are represented by dendrogram. Although the hierarchical clustering makes it easy to understand both the fine and coarse inclusive relations between clusters, it cannot be used for the high-dimension case or for large-scale data because of its large time complexity. This paper realizes a fast approximated hierarchical clustering method with small time complexity which utilizes the hash table made by Locality-Sensitive Hashing that is an algorithm for finding the candidates of the nearest neighbor points.

**Keyword** hierarchical clustering Locality-Sensitive Hashing

## 1 はじめに

クラスタ解析は、データを類似度でいくつかのクラスタに分類しデータの分布の特徴を抽出することで、対象となるデータ集合を解析する。生物学、疾病分類学、地質学、マーケティング、心理学、情報検索などいろいろなところで古くから使われており、近年では、画像や音声などのメディアデータの自動解析手段として利用機会が増えている[1]。

与えられた  $n$  個のデータを、評価基準(例えば二乗誤差等)が最小となるような  $K$  個のクラスタに分類したいとする。これは NP 完全問題で最適解を求める多項式時間アルゴリズムは存在しないと考えられている。そのため様々な多項式時間近似アルゴリズムが考えられてきた[2]。クラスタリングの近似アルゴリズムには二つのタイプがあり、それぞれ階層的的手法・非階層的的手法と呼ばれている。

本研究は Indyk らによって提案された近似の最近接点探索のアルゴリズムである Locality-Sensitive Hashing[3]によって作られるハッシュテーブルを用いた階層的クラスタ解析手法の近似アルゴリズムを提案し、実験によって大規模データに対して高速に実行できることやクラスタ解析に十分な性能を持っていることを示す。

## 2 階層的的手法と非階層的的手法

### 2.1 階層的的手法

階層的的手法とは、 $n$  個のデータが与えられたとき、これを要素数 1 の  $n$  個のクラスタとし、それを1つのクラスタになるまで順に結合していく手法である。まず全クラスタ間の類似度を求め、類似度が最大となるクラスタの組を結合し、1つのクラスタとする。そして結合して新しくできたクラスタと他のクラスタ間の類似度を更新する。この類似度が最大となる

組の探索、結合、類似度の更新という行程をクラスタの数が 1 になるまで繰り返す。手法によってクラスタ間の類似度を更新する際に用いる更新式が違い、それによって結合の順序や結果のクラスタの形状が変化する。クラスタを結合していく過程を表す樹形図を作り、クラスタ間の包含関係が理解しやすく、樹形図から分析者がクラスタ数を適切な決めることができる。しかしデータ数を  $n$  としたとき、brute force で実行すると時間計算量は  $O(n^3)$  と大きなものとなる。これを改良したアルゴリズムも考えられているが、それでも  $O(n^2)$  はかかる[4]。

### 2.2 非階層的的手法

非階層的的手法とは、データを決まった数に分類する手法である。まず分割するクラスタ数  $K$  を決め、 $K$  個のクラスタを適当に作る。そしてクラスタが妥当か評価し、良いクラスタが作れるのであれば作成したクラスタを元に再度計算し作り直す。これを作成したクラスタが評価基準を満たし妥当さが収束するか、決められた回数になるまで繰り返す。非階層的的手法は階層的的手法より高速に処理することができ、クラスタ間の類似度があまり重要ではなく、ただ単純に決められた数のクラスタに分割すればいいと言う場合有効だが、求まるクラスタが初期解に依存してしまうという不利な点もある。

### 2.3 本研究の位置づけ

評価基準を変えて非階層的的手法を複数回繰り返すことで階層的的手法を近似する。まずクラスタ数を決めずに厳しい評価基準で非階層的的手法でクラスタを作る。その結果、小さなクラスタが多数できる。次に評価基準を緩くした非階層的的手法で先ほど作ったクラスタをクラスタリングする。これをクラスタ数が 1 になるまで評価基準を緩くしていきながら繰り返す。クラスタが結合されていく細かい様子は分からないが、大まかなクラスタ間の包含関係は十分理解できる。

### 3 Locality-Sensitive Hashing

Locality-Sensitive Hashing(以下 LSH)とは、Indyk らによって提案された近似の最近接点探索のアルゴリズムで、これは類似しているデータ同士のハッシュ値は一致し、類似していないデータは異なるハッシュ値をとるようなハッシュ関数を用いてあらかじめハッシュテーブルを作っておくことで最近接点探索を高速に実行する。高次元のデータセットに対しても「次元の呪い」の影響を受けず、高速に処理することができる。

#### 3.1 ハッシュ関数

$C$  をデータ集合  $P$  のデータの中で最大の値とし、 $d$  を次元の数とする。Unary $C(x)$  は  $x$  の Unary 表現で、 $x$  個の 1 の列に  $C-x$  個の 0 の列が続く。データ  $p=(x_1, \dots, x_d)$  を以下のような二進数のベクトル  $v(p)$  に変換する。  $v(p) = \text{Unary}C(x_1) \dots \text{Unary}C(x_d)$ 。例えば  $p=(3,2)$ 、 $C=5$  とすると、 $v(p)=1110011000$  となる。

ハッシュ関数を定義する。  $\{1 \dots d'\}$  ( $d'=Cd$ ) からランダムに選ばれた  $k$  個の要素を持つ部分集合を  $l$  とする。ハッシュ関数は  $l$  によって選ばれた bit の値を繋げてハッシュ値とする。  $l=\{1,3,5,6,9\}$ 、 $v(p)=1110011000$  とすると、ハッシュ値  $g_l(p)=11010$  となる。

見る bit の位置を変えたハッシュ関数を  $L$  個用意する。見る bit の数  $k$ 、ハッシュ関数の数  $L$  は LSH の重要なパラメータである。

#### 3.2 LSH のアルゴリズム

LSH のアルゴリズムは二つの step からなる。  $P_j$  は集合  $P$  の要素を表す。

Step1: 全てのデータにハッシュ関数を適用し、

各ハッシュ関数に対してハッシュ値と同じ index のついたバケツにデータを格納したハッシュテーブルを作る

For each  $i=1 \dots L$

hash 関数  $g_i()$  を生成する

For each  $i=1 \dots L$

For each  $j=1 \dots n$

ハッシュテーブル  $T_i$  のバケツ  $g_i(P_j)$  に  $P_j$  を格納する

Step2: query  $q$  に対しても各ハッシュ関数でのハッシュ値を求め、各ハッシュテーブルの query のハッシュ値と同じ index のついたバケツの要素の和集合の中から最近接点を探索する

$S =$

For each  $i=1 \dots L$

$S \cup \{T_i \text{ のバケツ } g_i(q) \text{ の中に見つかった点}\}$

集合  $S$  の中から  $q$  に類似したデータを探す

### 4 提案手法

非階層的な手法の評価基準を変えたものを複数組み合わせることで階層的な手法を近似する。評価基準を満たしているかどうかの判定に LSH のハッシュテーブルを用い、同じバケツに属していれば評価基準を満たしているとみなす。ハッシュ関数で見る bit の数  $k$  を減らし、データが同じバケツに入りやすくなるように作り直したハッシュテーブルを用いることで評価基準を緩くする。

本提案手法は階層的クラスタリングの代表的な手法である Single Link の近似アルゴリズムである。

#### 4.1 アルゴリズム

本提案手法のアルゴリズムを説明する。各データ

を  $P_1, \dots, P_n$ 、ハッシュ関数の数を  $L$ 、ハッシュ関数の見る bit の数を  $k$  とする。アルゴリズムは以下のようなになる。

Step1: LSH のハッシュテーブルを作る

Step2: クラスタの結合

For each  $i=1 \dots n$

For each  $j=1 \dots L$

ハッシュテーブル  $T_j$  のバケツ  $g_j(P_i)$  に  $P_i$  とは異なるクラスタに属するデータがあるとき、そのクラスタと  $P_i$  の属するクラスタを結合する(図 1)

Step3: クラスタ数が 1 になるまで、 $k$  の値を減らす。クラスタ数が 1 でなければ Step1 に戻る

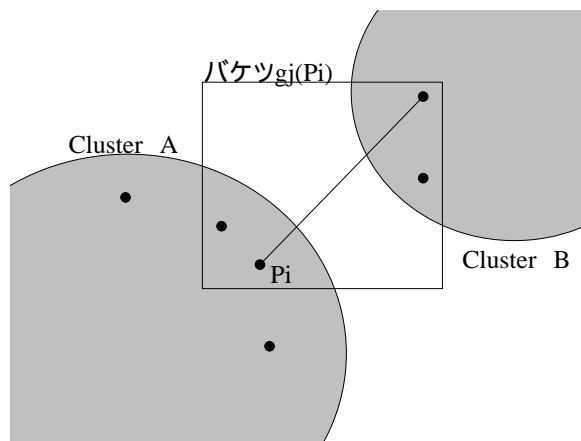


図1 step2 の様子

ハッシュテーブルを作り直す回数を  $X$  とすると、このアルゴリズムの時間計算量は  $O(nLX)$  となる。 $L$  も  $X$  も  $n$  と比べて極めて小さいので、計算時間は  $n$  の一乗に比例する。

#### 4.2 パラメータについて

本提案手法のパラメータはハッシュ関数の数  $L$ 、ハッシュ関数の見る bit の数  $k$ 、段階的に  $k$  を減らすやり方、特に  $K$  と  $K$  の更新方法は結果に大きな影響を与える。 $K$  の値は各階層での評価基準に影響し、 $K$  が大きいほど評価基準は厳密なものとなる。 $K$  の更新方法は各階層間の間隔とハッシュ

テーブルを作り直す回数に影響し、 $K$  の更新式が  $K$  の値をあまり大きく減らさなければ解析結果の精度は良くなるが計算時間は増える。

## 5 実験結果

本提案手法の性能を測るために brute force の Single Link との比較実験を行った。

### 5.1 計算時間

5次元で各次元は0-100の範囲の値をランダムにとるデータを作り、このデータに対して実験を行い計算時間を測定した。各パラメータは  $K$  の初期値を 100、 $K$  の更新式は  $K - K/2$ 、ハッシュ関数の数  $L=5$  とし、100,200,300,500,700,1000,1500,2000,3000,5000,7000 のデータ数に対して実験を行い、表 1 のような結果が得られた。

表 1 データ数に対する計算時間

データ数	Single Link	本手法(ms)
100	1 未満	445
200	31.5	461
300	78	484.5
500	328	523.5
700	898.5	562.5
1000	2375	617
1500	7289	711
2000	16453	797
3000	52586	976.5
5000	232891	1429.5
7000	625977	1726.5

この結果から、データ数が大きい場合、本手法は Single Link より高速に実行できることが分かる。データ数の少ない場合、Single Link より計算時間がかかっているが、これは  $O(n)$  の係数が大きいためである。

## 5.2 Irisの樹形図

クラスタ解析によく使用される Iris のデータに対して実験を行い、結果として作成した樹形図を比較した。Iris のデータは 4 次元の 150 個のデータである。提案手法のパラメータは  $K$  の初期値を 100、 $K$  の更新式  $K = K/2$ 、 $L=7$  とした。Single Link と本提案手法で解析した結果から作成した樹形図を図2に示す。

Single Link で求められた樹形図からデータを二つのクラスタに分割することが適切だと考える。データを二つのクラスタに分割したとき、両手法のクラスタの構成要素は全く同じで、同等のクラスタ析能力を持つと言える。

## 6 まとめ

Locality-Sensitive Hashing のハッシュテーブルを利用した  $O(n)$  で実行できる階層的クラスタ解析手法の Single Link の近似アルゴリズムを提案した。そして実験によって同等の解析能力を持つことと大規模データに対してはより高速に実行できることを示した。

本手法は大規模画像データベースからの各種の知識発見の基礎として利用できる可能性を持っている。

今後は適切なパラメータの決め方や LSH を利用した他階層的クラスタ解析手法の近似アルゴリズムを検討する予定である。

## 参考文献

[1] Anil K. Jain, "HANDBOOK OF PATTERN RECOGNITION AND IMAGE PROCESING", Academic Press Inc,1984

[2] Michael R. Anderberg, "Cluster Analysis for Applications", Academic Press Inc,1973

[3] Aristides Gionis, P. Indyk, R. Motwani, "Similarity Search in High Dimensions via Hashing", Proceedings of the 25th VLDB Conference, pp.518-528,1999

[4] Sung Young Jung, Taek-Soo Kim, "An Agglomerative Hierarchical Clustering using Partial Maximum Array and Incremental Similarity Computation Method", IEEE International Conference on Data Mining 2001

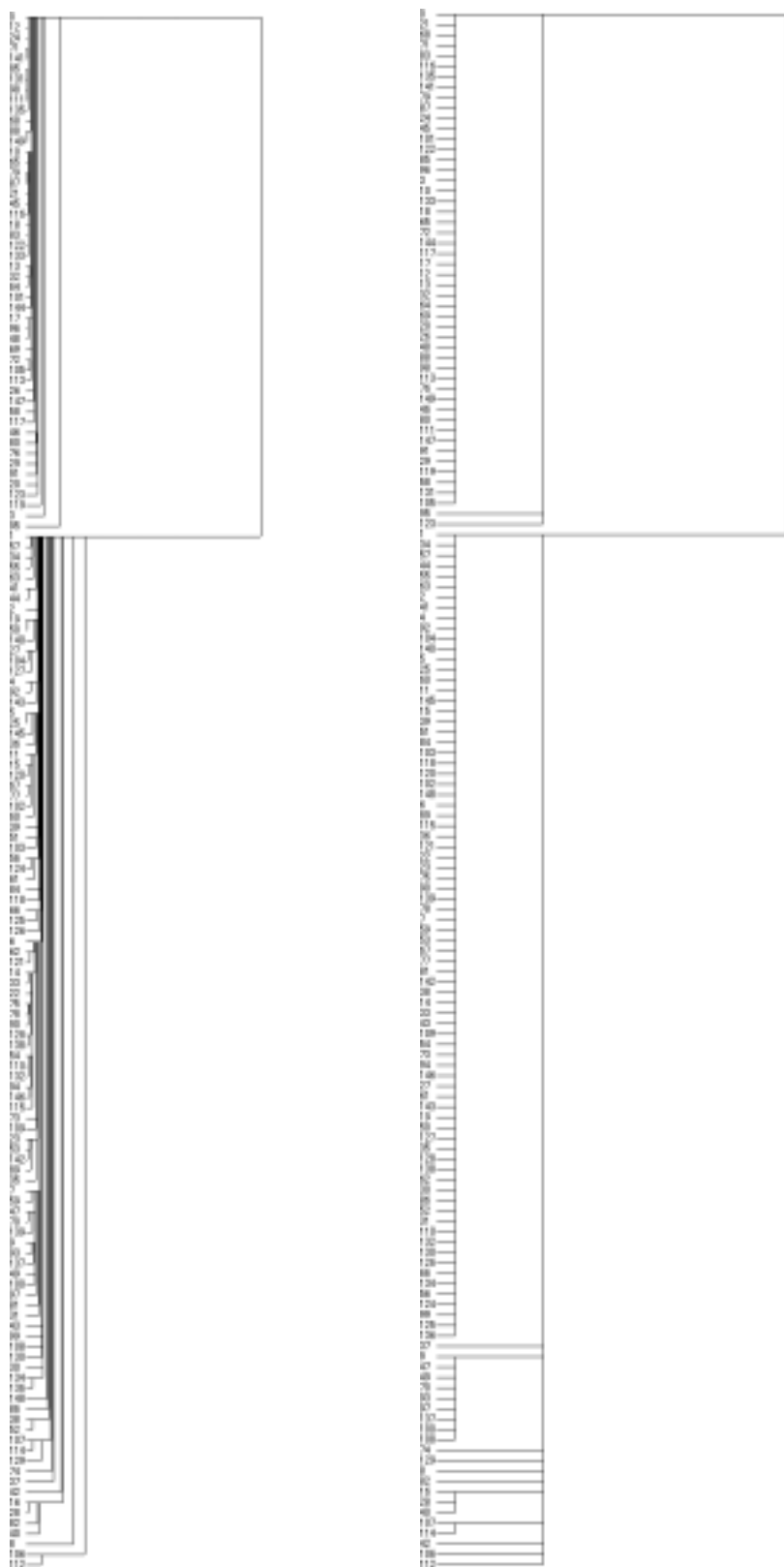


図2 Iris の解析結果から作成した樹形図 左:Single Link 右:提案手法  
樹形図の高さは結合したクラスタ間の距離