正規圧縮距離の木距離近似における枝長の定め方

米　巧[1,a)]　山内　由紀子[2]　来嶋　秀治[2]　山下　雅史[2]

概要：音楽, 文章, 文学作品, プログラム, 遺伝子情報, 自然言語など文字列で表現できるものは少なくない. このような文字列データ群に対し, 分類が行えると様々な応用につながる. 本論文では, 正規圧縮距離を用いた文字列データの分類手法について議論する. 具体的には, 分類するデータ間の正規圧縮距離行列を, 4 点法を用いて木距離に近似し, 分類するデータの系統樹を作成する. 本論文では特に系統樹に対して, 枝長を定める方法を提案し, 文字列データ間の距離の視覚性の向上を図る.

キーワード：4 点法 (quartet method) 正規圧縮距離 クラスタリング

On edge lengths in a tree metric approximating a normalized compression distance

Abstract: Those that can be represented by a strings, for example, music, text, literature, programs, genomes, executables, and natural language, are not few. There are a lot of applications of clustering of string datum. This paper is concerned with the clustering of string datum based on normalized compression distance: NCD. In particular, using quartet method, NCD matrix on string datum is approximated to a tree metric, and we get a phylogenetic tree of string data. This paper is proposes an algorithm to assign edge lengths in a tree metric approximating a normalized compression distance, aiming of improving the visibility of the distance between string data on the phylogenetic tree.

Keywords: quartet method normalized compression distance:NCD clustering

1. はじめに

音楽, 文章, 文学作品, プログラム, 遺伝子情報, 自然言語など文字列で表現できるものは少なくない. このような文字列データ群に対し, 文字列データ間の類似性を決定し, 類似のものから構成されるグループ (クラスタ) に分類するすること, すなわち, **クラスタリング**を検討する. クラスタリングが可能になると, たとえば, 好みの文学作品と類似した作品を検索することや, 自然言語間の関係の解明に使用できる. このように, 文字列データの分類は様々な応用を持っている.

Cilibrasi と Vitanyi[5] は, 一般には計算不可能な**コルモゴロフ記述量の理論 (Kolmogorov complexity theory)** に基づく類似距離を, 現実に使われている圧縮プログラムを用いた圧縮度に基づく**正規圧縮距離 (normalized compression distance: NCD)** によって近似する方法を提案している. さらに, 類似距離を木距離に近似するために 4 点法 (quartet method) を提案し, 木距離から系統樹を作成することでクラスタリングを行った. 文献 [5] に, 哺乳類のゲノムデータ, 文学作品, 音楽 MIDI ファイル, プログラムをこの手法を用いてクラスタリングがした結果が示されている.

Cilirasi と Vitanyi[5] の提案した 4 点法で作成された系統樹は, 葉にデータを持つ (2 分) 木であり, 近い位置にあるデータ同士は類似度が高く, 離れているデータほど類似度が低い. 本研究では, 作成した系統樹の, 各枝に長さを定める方法を提案する. 系統樹の任意のデータ対に対して, データ間の距離が, データ間を結ぶ重み付き最短路の長さに一致する系統樹を作成することで, 視覚的にもデータ間の距離をとらえることができるようになる.

[1] 九州大学工学部電気情報工学科
Department of Electrical Engineering and Computer Science, Kyushu University
[2] 九州大学大学院システム情報科学研究院)
Department of Imformatics, Kyushu University
[a)] yone@tcslab.csce.kyushu-u.ac.jp

2. 準備

2.1 正規圧縮距離を用いた類似距離

本研究では, 対象とするデータ間の類似度を**正規圧縮距離** [5] を用いて測る. 正規圧縮距離は文字列データ間の距離を圧縮度に基づいて定義する.

2.1.1 コルモゴロフ記述量

有限集合 U に対し, 非負実数値を与える距離関数 $D : U \times U \to R$ が U の任意の元 x, y, z に対し, 以下の 3 つの条件

$$D(x,y) = 0 \Leftrightarrow x = y \quad (\text{非退化性})$$
$$D(x,y) = D(y,x) \quad (\text{対称性})$$
$$D(x,z) \leq D(x,y) + D(y,z) \quad (\text{三角不等式})$$

を満たすとき D を U 上の距離と呼ぶ.

コルモゴロフ記述量の理論 (Kolmogorov complexity theory) を用いることで対象とするデータに対し圧縮度に基づいて類似度の測定を行い, 汎用な類似距離を導くことができる. コルモゴロフ記述量とは有限長のデータ列の複雑度を表す指標の一つで u を万能チューリング計算機 l (x) を x の文字列の長さ $u(p)$ を計算機 u 上でプログラム p を実行した際に得られる出力とすると

$$K_u(x) = \min_{p:u(p)=x} l(p)$$

で定義される. 与えられた入力 x に対するコルモゴロフ記述量 $K_u(x)$ は, 大まかに述べると x を復号可能な範囲で極限まで圧縮した際に得られる 2 進列のビット長のことである. 文献 [3] ではコルモゴロフ記述量は計算機が異なっても定数項分しか値が変わらないことが証明されている. そこで, 今後, 計算機の表記を省略して, x のコルモゴロフ記述量を $K(x)$ と表記する.

さらに, 一般的に y に対する x の相対コルモゴロフ記述量を $K(x|y)$ で表す. これは補助データ y を用いて復元することを想定したときの x の最小の圧縮列長である. y に含まれる情報を用いるため, $K(x|y)$ は $K(x)$ よりも小さくなる可能性がある. $K(x)$ と $K(x|y)$ の差 $K(x) - K(x|y)$ は y の中に含まれる x の情報量と考えられる. また文献 [3] では, $K(x) - K(x|y)$ と $K(y) - K(y|x)$ が小さい誤差の範囲で等しいことが証明されている. つまり, $K(x) - K(x|y)$ あるいは $K(y) - K\ (y|x)$ を $I(x,y)$ とすると $I(x,y)$ は x と y の中に含まれる両者の類似部分と考えることができる. 両者の類似部分を正規化した正規情報距離 (normalized information distance : NID) を以下の式で定義する.

$$NID(x,y) = \frac{\max\{K(x \mid y), K(y \mid x)\}}{\max\{K(x), K(y)\}} \quad (1)$$

しかし, 一般に, コルモゴロフ記述量は計算不可能である. そこで, 文字列 x を現実に使われている圧縮プログラムで圧縮した際に得られる x の圧縮列長 $C(x)$ を文字列 x に対するコルモゴロフ記述量 $K(x)$ の近似として用いる. 圧縮プログラムには相対コルモゴロフ記述量を定義するときに使った補助情報を用いた圧縮という概念がないため, 相対記述量 $C(x|y)$ を

$$C(x \mid y) = C(yx) - C(y) \quad (2)$$

と定義する. 直感的には, yx を記述するには y を記述し, それを用いて x を記述するのが順当である. この前者の記述量が $C(y)$, 後者が $C(x|y)$ である. 従って $C(yx) = C(y) + C(x|y)$ が成り立つ. コルモゴロフ記述量においても, この関係式が小さい誤差の範囲で成立する. $C(x)$ と $C(x|y)$ をコルモゴロフ記述量の代わりに用いると, $C(x) - C(x|y)$=$C(x) + C(y) - C(yx)$ が得られ, これを x と y の類似度の近似として用いる. 式 (2.2) を式 (2.1) に代入すると, 本論文で使用する類似距離である正規圧縮距離 (normalized compression distance : NCD) を得る.

$$NCD(x,y) = \frac{\max\{C(xy) - C(x), C(yx) - C(y)\}}{\max\{C(x), C(y)\}} \quad (3)$$

2.2 木距離

有限な点集合 V と辺集合 E からなる無効グラフを $G = (V, E)$ とする. G のある全域木 T と枝長 $\lambda\ _e (e \in E)$ が存在し, 任意の 2 頂点 x, y について

$$d(x,y) = \sum_{e \in P(x,y)} \lambda_e \quad (4)$$

が成り立つとき, 距離 d を木距離と呼ぶ. ここで $P(x,y)$ は x と y を結ぶの T 上の最短パスを表す.

2.3 4 点法

Cilibrasi と Vitanyi の考案した 4 点法 (quartet method)[2] を紹介する. 4 点法は, 入力として n 個のデータの距離行列を取り, n 個の葉と $n-2$ 個の内部頂点を持ち, 全ての内部頂点の次数が 3 である系統樹を得るための手法である. この系統樹は, 各葉にはある対象データが互いに近い距離にあるデータが互いに近い葉に来るようにラベルづけされる. より具体的には, 2.3.1 項で紹介する 4 点条件の非達成度を評価する目的関数を最小化木距離で与えられた距離行列を近似し, 系統樹を作成する. しかし, 最適解を求めるのは NP 困難であるため, Cilibrasi と Vitanyi はヒューリスティクスを使った近似アルゴリズムを提案した.

2.3.1 4 点条件 [1]

4 点法で用いる目的関数の設定のために以下の定理を紹介する.

定理 1 (4 点条件)**.** 有限集合を U とする. U 上の距離 d が式 (4) で定義される, 木距離となるための必要十分条件は, 任意の 4 点 $u, v, w, x \in U$ に対して

$d(u,v)+d(w,x) \leq \max\{d(u,w)+d(v,x)\ ,\ d(u,x)+d(w,v)\}$
$d(u,w)+d(v,x) \leq \max\{d(u,v)+d(w,x)\ ,\ d(u,x)+d(w,v)\}$
$d(u,x)+d(w,v) \leq \max\{d(u,v)+d(w,x)\ ,\ d(u,w)+d(v,x)\}$

が成り立ち, 少なくともその中の 2 つで等号が成り立つことである.

2.3.2 目的関数

上で紹介した 4 点条件の非達成度を評価するための目的関数を紹介する. データ集合を N とし, $|N|= n$ と仮定する. データ集合 N に対し, n 個のデータ間には距離行列が定義されているとする. この n 個のデータを葉に持ち, 内部頂点数が $n-2$ である木 $T=(V,E)$ を考える. このとき木 T 上の異なる 4 個の葉 u,v,w,x の位置関係は図 2.1 に示す 3 通りが考えられる. これらを 4 項組 (quartet) と呼ぶ. これらの中の 1 つは, 必ず 4 点条件を満たす. この中で 4 点条件を満たす 4 項組を consistent と呼ぶ.

図 2.2 で説明すると, 4 点条件を満たす組み合わせは葉 1,2,3,4 を選んだときには (1,3| 2,4) である. 葉 3,4,5,6 を選んだときには (3,5| 4,6) である. ここで,

$$W(u,v,w,x)=\begin{cases} d(u,v)+d(w,x) \\ (\text{uv|wx が consistent のとき}) \\ d(u,w)+d(v,x) \\ (\text{uw|vx が consistent のとき}) \\ d(u,x)+d(v,w) \\ (\text{ux|vw が consistent のとき}) \end{cases}$$

と定義する. この W を下式であらわすように n 個の葉の全ての組み合わせに対して計算することで木 T に対する目的関数を定義する.

$$W_T=\sum_{u,v,w,x\in N} W(u,v,w,x) \tag{5}$$

この目的関数に関する定理として, 文献 [4] で示される定理を紹介する.

定理 2. 与えられたデータ集合 N 上の距離 d が木距離と仮定する. このとき, 目的関数最小化問題の最適解は, 元の木に一致する.

上の定理で述べる目的関数最小化問題とは, 式 (5) の目的関数のコスト最小化問題

$$\min_T W_T \tag{6}$$

であり, 4 点法ではこの目的関数を最小化する全域木を用いて距離行列を木距離によって近似する.

2.3.3 評価関数

Cilibrasi と Vitanyi は式 (5) の目的関数を用いた評価関数を定義している. 任意の u,v,w,x に対しそれらの 3 つの quartet の中で最良コスト, 最悪コストを

$$m(u,v,w,x)=\min\{W_{uv|wx},W_{uw|vx},W_{ux|vw}\}$$
$$M(u,v,w,x)=\max\{W_{uv|wx},W_{uw|vx},W_{ux|vw}\}$$

とする. 目的関数の上界と下界は

$$m=\sum_{u,v,w,x\in N} m(u,v,w,x)$$
$$M=\sum_{u,v,w,x\in N} M(u,v,w,x)$$

となる. これを用いて木 T に対する目的関数を

$$S(T)=\frac{M-W_T}{M-m} \tag{7}$$

2.4 Cilibrasi と Vitanyi のアルゴリズム

式 (5) の目的関数の最適解を求めることは NP 困難であり Cilibrasi と Vitanyi は以下のヒューリスティックスを提案した.

- 葉に対象とする n 個のデータを持つランダムに作成された 3 正則木に対し $S(T)$ を計算する
- 木に対し以下に示す 3 つのオペレーションを確率的に行う
- 変形後の木を T' とする. $S(T)$<$S(T')$ ならば木を T' に更新し $S(T)\geq S(T')$ ならば元の木 T' を保持する

木に対するオペレーションとしては以下の 3 つである.

leaf swap ランダムに葉を 2 つ選び交換する
subtree swap ランダムに 2 つの部分木を選び交換する
subtree transfer ランダムに 1 つの部分木 (葉でも可) を選び別の辺に挿入する

この操作を,$S(T)$=1 とならない限り木の変形を一定回数繰り返す.

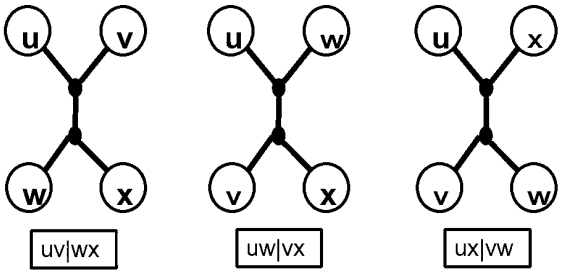

図 1 木 T 上の異なる 4 点 u,v,w,x で考えられる 3 通りの位置関係を示す. またそれぞれの位置関係の下に. 4 点条件を満たす u,v,w,x の組み合わせを示す.

3. 系統樹の枝長の定め方

本章では, 2 章で述べた 4 点法で得られた系統樹に枝長を割り当てる方法について述べる.

4 点法により, 正規圧縮距離を木距離に近似することで作成した系統樹に対し, 枝長を定義するアルゴリズムを紹介する. このアルゴリズムは, データ間の距離を定義した $n\times n$ 行列 D と, 4 点法により, 距離行列 D から木距離へと近似した系統樹 $T=(V,E)$ を入力として受け取り, 系統

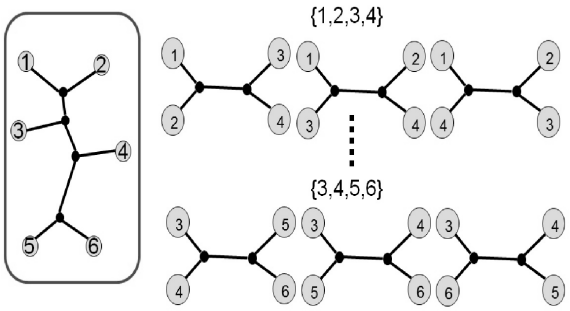

図 2 具体的な 3 正則木 T の例. T に対する目的関数値は全ての葉の 4 項組の回数 $W(u,v,w,x)$ を計算する. そのため木に対する目的関数値は $O(n^4)$ の計算が必要となる.

樹の頂点間の距離を隣接行列で表した $(2n-1)\times(2n-1)$ 行列 D' を出力する. $D(i,j)$ は D の i,j 成分を表す.

アルゴリズムの説明のために使用する言葉の説明をする. 系統樹 $T=(V,E)$ において, V の中で葉の部分の頂点の集合を L とし, L に属する頂点を葉と呼ぶ. 共通の親ノード v_p を持つ頂点 v_i, v_j を兄弟である頂点と呼ぶ. またアルゴリズムでは兄弟である頂点 v_i, v_j から親ノード v_p までの枝長 e_i, e_j を計算する. その後, 親ノード v_p から他の頂点に対して新たに頂点間の距離を計算する. この新たに他の頂点までの距離を計算した頂点 v_p のことを距離づけされたノード V_{dist} と呼ぶ. また親 v_p までの枝長 e_i, e_j が計算された葉, 内部頂点をそれぞれ, 探索済みの葉, 探索済みの内部頂点と呼ぶ. アルゴリズムの概要は以下である.

$D'=0$ とし,n 以下のすべての自然数 i,j に対し, $D'(i,j)$ に n データ間の距離行列の値 $D(i,j)$ の値を代入する. 兄弟である頂点 v_i, v_j を選択し, 各葉から親ノード v_p までの距離 e_i, e_j をそれぞれ下式 (7), (8) で計算する. 兄弟である頂点を選択する際, 葉同士, 葉と距離づけされた内部頂点, もしくは距離づけされた内部頂点同士でなければ e_i, e_j は計算できない. また親 v_p までの距離 e_i, e_j を重複して計算することを防ぐため, 葉の選択は未探索の葉, 内部頂点の選択は探索済みの頂点選択するものとする. e_i, e_j は以下の式で計算する.

$$e_i = \frac{D'(v_i,v_j) + \frac{\sum_{u\in L}\{D'(v_i,u)-D'(v_j,u)\}}{|L|}}{2} \tag{8}$$

$$e_j = \frac{D'(v_i,v_j) - \frac{\sum_{u\in L}\{D'(v_i,u)-D'(v_j,u)\}}{|L|}}{2} \tag{9}$$

式 (7), 式 (8) は葉 v_i, v_j 間の距離の和と差を, 和を $D'(v_i,v_j)$, 差を v_i, v_j から他の葉までの距離の差の総和の平均 $\frac{\sum_{u\in L}\{D'(v_i,u)-D'(v_j,u)\}}{|L|}$ を計算し e_i+e_j, e_i-e_j から e_i, e_j を計算している. その後, 親ノード v_p から系統樹全体の葉 L と, 距離づけされたノード V_{dist} へ新たに距離 $d(v_p,V_{dist})$,$d(v_p,L)$ を計算し距離行列 D' に追加していく. この時, 親ノード v_p から自分の子孫に属している葉 L に対して距離を計算する必要はない. $D'(v_p,L)$,$D'(v_p,V_{dist})$ は以下の式で計算する.

$$D'(v_p,V_L) = \frac{D'(v_i,V_L) - e_i + D'(v_j,V_L) - e_j}{2}$$

$$D'(v_p,V_{dist}) = \frac{D'(v_i,V_{dist}) - e_i + D'(v_j,V_{dist}) - e_j}{2}$$

上式は親ノードからある頂点までの距離を, 両子供からその頂点までの距離からそれぞれ上で計算した e_i, e_j を引いたものの和の平均をとり計算している.

この操作を兄弟となる頂点が系統樹の根の両子供となるまで繰り返す. 根の両子供が選択された場合には e_i, e_j の和しか計算できないため, $e_i=0$,, $e_j=D'(v_i,v_j)$ とする. このアルゴリズムの疑似コードは Algorithm1 である.

Algorithm 1 culculate_ distance

Input :葉間の距離を与える $n\times n$ 行列 D, D を近似した系統樹 T
n 以下のすべての自然数 i,j に対して
$D'(i,j)=D(i,j)$ とする.
L の中から任意の兄弟 v_i, v_j を選択する.
while $v_i, v_j \neq$ 系統樹の根の両子供 **do**
  v_i, v_j から v_p までの距離 e_i, e_j を計算する.
  v_p から, 各祖先までの距離を計算し D' に追加.
  未探索の葉もしくは, 探索済みの内部頂点かつ, v_p までの e_i, e_j の計算されてない兄弟 v_i, v_j を選択.
end while
根の 2 つのを v_i, v_j とする. $e_i=0, e_j=D'(v_i,v_j)$ として D' に追加.
return D'

4. おわりに

4.1 まとめ

本論文では, 圧縮度に基づく類似度である正規圧縮距離の木距離近似における枝長の定め方を提案した.

4.2 今後の課題

今後の課題として, 大きい n に対する系統樹作成を目標とし, 部分的に作成した系統樹を組み合わせることで, データ集合全体の系統樹を作成可能であるかを検討する.

参考文献

[1] P. Buneman, “A note on metric properties of trees,” *Jounal of Combinatorial Theory B*, 17, 48–50, 1974.
[2] R. Cilibrasi and P. Vitanyi, “Clusterring by compression”, *IEEE Trans Information Theory,* 51, 1523–1545, 2005.
[3] T.M. Cover and J.A. Thomas, *Elements of Information Theory, 2nd Edition*, Wiley-interscience, 2006.
[4] 久保 浩平, 正規圧縮距離を用いたクラスタリング法の実装と木距離への近似について, 九州大学理学部物理学科情報理学コース卒業論文, 2013.
[5] P. Vitanyi, 渡辺 治 (訳), “圧縮度に基づいた汎用な類似度測定法,” 数理科学, 519, 54–59, 2006.