LA_001

繰り返し表現木最小化アルゴリズム
Algorithm for Minimizing Repetition Representation Trees

斉藤 智哉* 中村 篤祥* 工藤 峰一*
Tomoya Saito Atsuyoshi Nakamura Mineichi Kudo

概要

本論文では、ラベル付順序木の繰返し構造を明示的に表す繰返し表現木を提案し、与えられた節点数 n の木に対し、節点数最小の繰返し表現木を求める問題を $O(n^3)$ 時間で解くアルゴリズムを示す。最もコンパクトな形による解釈は、本質的な繰返しの構造を捉えている可能性が高いものと期待できる。

1. はじめに

シーケンスデータにおいて、連続的に現れる繰返しは特別な意味をもつことが多い。例えば、楽曲においては、連続的な繰返しは印象に残る部分であり、DNA シーケンスにおいては tandem repeat と呼ばれ、ある遺伝病と関係があることが知られている。また、Web ページに埋め込まれたデータ集合は、テキスト部分をすべて同じ1文字に置換した場合、連続的な繰返し部分となっていることが多い。したがって、シーケンスから連続的な繰返しを抽出することは、様々な分野で役に立つと考えられる。

シーケンスから連続的な繰返しを抽出する場合、1つの問題点はどの部分を繰返しとみなすか複数の解釈が存在することである。どの解釈が尤もらしいかは応用に依存するが、複数のデータが埋め込まれた Web ページから1データずつ抽出する問題では、ヒューリスティクス等を用いて局所的に判断して尤もらしい解釈の繰返し構造を抽出する方法が開発されてきた [1–3]。それに対し、本論文では繰返し構造の自然な表現である繰返し表現木を提案し、その最小形を求めることにより、大局的に判断して尤もらしい解釈の繰返し構造を抽出する方法を提案する。最もコンパクトな形による解釈は、本質的な繰返しの構造を捉えている可能性が高いものと期待できる。

本論文で提案する繰返し表現木は、ラベル付き順序木の繰返し構造の明示的な表現である。つまり、提案する手法はラベル付き順序木における繰返し構造を把握するためのものである。Web ページからデータを抽出する場合、HTML や XML のタグ構造をラベル付き順序木である DOM(Document Object Model) 木として表現できるので、本手法をそのまま適用することができる。また、シーケンスに対しては、シーケンスを高さ1のフラットな順序木とみなすことにより本手法を適用することが可能である。

本論文では、最小繰返し表現木を求める問題を、木の節点数 n に対して時間計算量 $O(n^3)$ で解くアルゴリズムを示す。また、抽出される繰返し構造について、Web からの繰返し構造抽出法として提案された、ヒューリスティクス等を用いて局所的に判断する2つの既存法 [2,3] との違いを考察する。

*北海道大学大学院情報科学研究科, Hokkaido University

2. 問題設定

2.1 繰返し表現木

根付き木で、同じ親をもつすべての子の節点間に全順序が与えられた木を順序木という。木を図示するとき、子は左から右に順序付けされているとする。本論文では、全ての節点にラベルが与えられたラベル付順序木を扱う。以後、ラベル付順序木を単に木と呼ぶ。

連続的な繰返し構造をもつ木をコンパクトに表現するために、繰返し情報節点というものが導入された木を考える。繰返し情報節点 v_1 とは、繰返し回数 $r \geq 2$ でラベル付けされた節点であり、v_1 のすべての子 $c_1, c_2, ..., c_k$ を根とする部分木列が v_1 の親 p の子を根とする部分木列として r 回繰返された木を表現しているものとする (図1)。繰返し情報節点 v を、それを用いずにそれが表現し

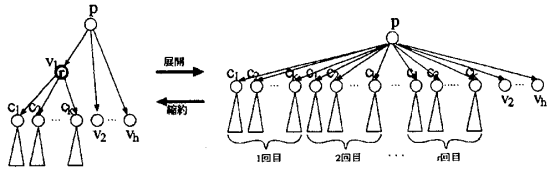

図 1: 繰返し情報節点とその展開

ている木に変換する操作を v の展開といい、その逆の操作を縮約という。根と葉以外の節点として繰返し情報節点を0個以上もつ木を繰返し表現木という。繰返し表現木 R に含まれるすべての繰返し情報節点を展開すると、繰返し情報節点を1つも含まない木 T が得られる。このとき、「繰返し表現木 R は、木 T を表現する」という。

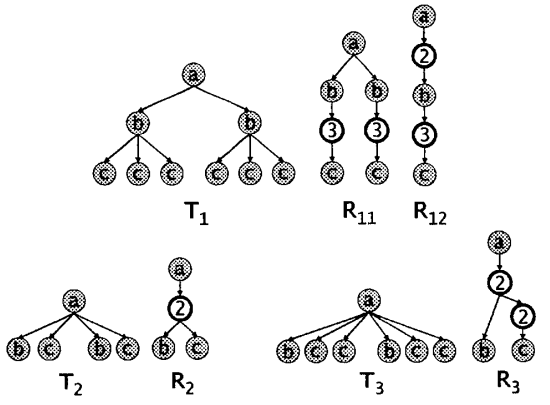

図 2: 繰返し表現木の例

例 1 図 2 の T_1, T_2, T_3 は一般的な木であり、それ自身を表現する繰返し表現木でもある。R_{11} と R_{12} は T_1 を、R_2 は T_2 を、R_3 は T_3 を表現する繰返し表現木である。

2.2 最小繰返し表現木算出問題

本論文では、以下のような問題を扱う。

問題 1 (節点数最小繰返し表現木算出問題) 与えられた木 T を表現する節点数最小の繰返し表現木 R を求めよ。

ただし、ここでいう節点数とは、繰返し情報節点を含めたすべての節点の数のことである。

例 2 図 2 の木 T_1, T_2, T_3 を表す節点数最小の繰返表現木はそれぞれ R_{12}, R_2, R_3 である。このとき節点数はそれぞれ、$9 \to 5, 5 \to 4, 7 \to 5$ というように減っており、よりコンパクトな表現になっている。

注意 1 節点数最小の繰返し表現木は複数存在する場合があるが、それらの内 1 つを求めればよいこととする。

3.2 節で説明するように、最小繰返し表現木算出問題は、以下のように定義される「高さ 1 の木に対する節点重み和最小の繰返し表現木算出問題」を繰返し解く問題に帰着できる。

問題 2 (高さ 1 の木に対する節点重み和最小の繰返し表現木算出問題) ラベルの集合を Σ とし、Σ の各要素 l に実数重み $w(l)$ が与えられているとする。Σ 上の木で高さ 1 の任意の木 T に対し、T を表現する節点重み和最小の繰返し表現木 R を求めよ。ただし、繰返し情報節点の重みは 1 とする。

3. アルゴリズム

この節では、まず、高さ 1 の木に対する節点重み和最小の繰返し表現木算出問題を解くアルゴリズムを示し、その後、そのアルゴリズムを用いて節点数最小繰返し表現木算出問題を解くアルゴリズムを示す。

3.1 高さ 1 の木に対する 節点重み和最小の繰返し表現木算出問題を解くアルゴリズム

まず、記述の簡略化のため、以下のような表記を導入する。高さが 1 で葉節点の数が n の木 T に対し、T の葉節点を i 番目から j 番目までに限った木を $T[i,j]$ で表す。$T[i,j]$ を表現する節点重み和最小繰返し表現木を $R[i,j]$ とする。同じラベル l の根をもつ木 R_1 と R_2 に対し、R_1 を前 (左) にして 2 つの木の根を重ねてできる木を $R_1 + R_2$ とする (図 3 右から 2 番目の木参照)。任意の自然数 h と繰返し表現木 R に対し、R の根と深さ 1 の節点列の間にラベル h の繰返し情報節点を挿入した形の木を $h \times R$ とする (図 3 右端の木参照)。

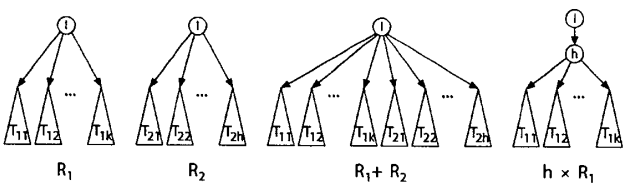

図 3: 根のラベルが等しい 2 つの繰返し表現木の和と繰返し表現木のスカラー倍

MinH1Tree % [Minimize Height-One Tree]
入力 l: 根節点のラベル、$(l_1, m_1), (l_2, m_2), ..., (l_n, m_n)$: 高さ 1 の木 T の葉の (ラベル, 重み) の列
出力 (N_R, m_R): T を表現する節点重み和最小の繰返し表現木 R の根節点 N_R とその節点重み和 m_R

```
1: m[i,i] ← m_i for i = 1,2,...,n
2: m[i,j] ← ∞ for i = 1,2,...,n, j = 2,3,...,n, i < j
3: for k = 1 to n do
4:   for i = 1 to n − k + 1 do
5:     for j = 0 to k − 2 do
6:       if m[i,i+j]+m[i+j+1,i+k−1] < m[i,i+k−1]
         then
7:         m[i,i+k−1] ← m[i,i+j]+m[i+j+1,i+k−1]
8:         type[i,i+k−1] ← (2,j)
9:       end if
10:    end for
11:    h ← 2
12:    while i + hk − 1 ≤ n do
13:      if (l_{i+(h−1)k}, l_{i+(h−1)k+1}, ..., l_{i+(h−1)k+k−1})
           ≠ (l_i, l_{i+1}, ..., l_{i+k−1}) then
14:        break
15:      end if
16:      if m[i,i+k−1]+1 < m[i,i+hk−1] then
17:        m[i,i+hk−1] ← m[i,i+k−1]+1
18:        type[i,i+hk−1] ← (1,k)
19:      end if
20:      h ← h + 1
21:    end while
22:  end for
23: end for
24: m_R ← m[1,n] + 1
25: N_R ← ConstructMinTree(l, l_1, ..., l_n, type)
26: return (m_R, N_R)
```

図 4: アルゴリズム MinH1Tree

命題 1 繰返し表現木 R が、高さ 1 で葉の数が $n \geq 2$ の木 T を表現する節点重み和最小繰返し表現木であれば、以下の *Case 1* または *Case 2* のどちらかが成り立つ。

Case 1 $kh = n$ を満たすある自然数 k と h が存在し、$R = h \times R[1,k]$ を満たす。

Case 2 $1 \leq j < n$ を満たすある j が存在し、$R = R[1,j] + R[j+1,n]$ を満たす。

すなわち、Case 1 か Case 2 を満たす候補のうち、節点重み和が最小の木が R である。いずれも、$j-i+1 < n$ を満たす $R[i,j]$ を用いて重み和を求めることができるので、$R[i,i] = T[i,i]$ という事実を使って、動的計画法により R を求めることができる。

図 4 に、高さ 1 の木に対する 節点重み和最小の繰返し表現木算出問題を解くアルゴリズム MinH1Tree を示す。このアルゴリズムでは、$k = 1, 2, ..., n$ の順に、$1 \leq i \leq n-k$ を満たす i に対する $R[i, i+k-1]$ の根節点を除いた節点重み和 $m[i, i+k-1]$ を求める。$k = 1$ の場合は、$R[i,i] = T[i,i]$ なので $m[i,i] = m_i$ となる (1 行目)。$k \geq 2$ の場合は、命題 1 のいずれかの Case を満たす候補木の中から節点重み和が最小となるものを探す。

Case 2 を満たす候補木は、5 行目から 10 行目で求められる。k が小さいものから計算しているので、Case 2 を満たす候補木の節点重み和はひとつあたり $O(1)$ で得られる。候補木の節点重み和が現在の $m[i, i+k-1]$ より小さければ、$m[i, i+k-1]$ を更新する。

Case 1を満たす候補木は、11行目から21行目で求められる。ここではi番目の節点から始まるk節点を単位としたh回の繰返しが存在するかを調べる(13行目)。$T[i,i+k-1]$が$k'(<k)$節点ずつの繰返しになっているかは既に計算されており、またCase 2の処理が済んでいるので、この時点で$R[i,i+k-1]$の節点重み和が$m[i,i+k-1]$に入っている。よってCase 2を満たす候補木の節点重み和はひとつあたり$O(1)$で得られる。候補木の節点重み和が現在の$m[i,i+hk-1]$より小さければ、$m[i,i+hk-1]$を更新する。

また、それぞれ$m[i,j]$を更新する際に、選択したCaseと、いくつの節点を繰返しの単位としたか、またはどの位置で分割したかをtype$[i,j]$に保持しておく。

最終的に、$m[1,n]$には$R=R[1,n]$の、根を除く節点重み和が設定される。選択された繰返し単位・分割位置をtype$[1,n]$から遡ることで、節点重み和$m[1,n]$を達成する繰返し表現木Rを$O(n)$時間で構成できる。このようなアルゴリズムをConstructMinTreeとする。最後に$m[1,n]+1$をm_R、ConstructMinTreeで求まったRの根節点をN_Rとして(m_R,N_R)を出力する。このようにして、MinH1Treeは与えられた節点重み付き木Tに対し、Tを表現する節点重み和最小の繰返し表現木Rの節点重み和m_RとRの根節点N_Rの組を出力する。

命題 2 *MinH1Tree*の時間計算量は$O(n^3)$である。

(証明) 5行目から10行目の計算量は$O(n)$である。12行目から21行目までのループ回数は高々n/k回であり、13行目で必要なラベルの比較はk回なので、計算量は$O(n)$である。よって、3行目から23行目までの計算量は$O(n^3)$である。他の処理はすべて$O(n)$であるから、全体として$O(n^3)$で計算できる。 □

3.2 最小繰返し表現木算出問題を解くアルゴリズム

この節では、与えられた木Tを表現する節点数が最小の繰返し表現木Rを求めるアルゴリズムConv2MinRRTree (図5)を説明する。

Conv2MinRRTree % [Convert to the Minimum Repetition Representation Tree]
入力 N_T: 木Tの根節点
出力 (N_R,m_R): Tを表現する節点数最小の繰返し表現木Rの根節点N_Rとその節点数m_R

1: **if** N_Tが子を持たない **then**
2: return $(N_T,1)$
3: **else**
4: **for** $i=1$ to N_Tの子の数 **do**
5: $(N_i,m_i) \leftarrow$ Conv2MinRRTree(N_Tのi番目の子)
6: **end for**
7: $(l_1,l_2,...,l_k) \leftarrow$ Labeling$(N_1,N_2,...N_k)$
8: $(N_R,m_R) \leftarrow$MinH1Tree$(l,(l_1,m_1),(l_2,m_2),...,(l_k,m_k))$
9: 根がN_Rの木Rの葉節点でラベルがl_iの節点をN_iに置換
10: **end if**
11: return (N_R,m_R)

図5: アルゴリズムConv2MinRRTree

Conv2MinRRTreeは、Tの根節点N_Tが入力されると、Tを表現する最小節点数の繰返し表現木Rの根節点N_Rとその節点数m_Rの組を出力する。

根だけからなる高さが0の木Tが与えられた場合は、Tを表現する繰返し表現木がTそのものしかないため、$(N_T,1)$を出力する。高さが1以上の木Tの根節点N_Tが入力された場合は、N_Tの子を根とする部分木$T_1,T_2,...,T_k$に対応する葉節点$L_1,L_2,...,L_k$をもち、それぞれの葉節点が以下の条件(1)を満たすラベル$l_1,l_2,...,l_k$をもつ高さ1の木T'を求める。

$$l_i=l_j \Leftrightarrow T_i=T_j \text{ for } (i,j)\in\{1,2,...,k\}^2 \quad (1)$$

このようなラベル付けを行なうのが図5のLabelingである(7行目)。Labelingの引数としてConv2MinRRTree(5行目)で求めたT_iを表現する最小繰返し表現木R_iの根節点N_iの列を渡しているが、Conv2MinRRTreeは決定的アルゴリズムなので$T_i=T_j$と$R_i=R_j$は等価となることに注意されたい。求めたラベルl_iに対して、実数重み$w(l_i)$をR_iの節点数m_iとする。また、T'の根節点のラベルはN_Tのラベルlとし、$w(l)=1$とする。Tからこのように構築された高さ1の木T'とラベルの重みwに対する「高さ1の木に対する節点重み和最小の繰返し表現木算出問題」をMinH1Treeで解き(8行目)、得られたT'に対する節点重み和最小の繰返し表現木Rの根節点N_Rとその節点重み和m_Rを得る。最後にRの葉節点各々を、ラベルがl_iであればT_iを表現する最小繰返し表現木R_iの根節点に置換する(9行目)。Conv2MinRRTreeは、こうしてできたRの根節点N_Rと節点重み和m_Rの組を出力する。

命題 3 木Tに対し、*Conv2MinRRTree*の出力が(N_R,m_R)であったとする。このとき、N_Rを根とする繰返し表現木RはTを表現する節点数最小の繰返し表現木であり、その節点数はm_Rである。

(証明) 木Tの高さhに関する数学的帰納法によって証明する。$h=0$のとき、明らかに$R=T$であり、節点数は1であるから、命題は成り立つ。

高さが$h-1$以下の木に対して命題が成り立っていると仮定する。N_Tのi番目の子を根とする部分木をT_iとすれば、5行目で得られるN_iはT_iを表現する最小繰返し表現木R_iの根節点であり、その節点数はm_iである。Tを表す最小繰返し表現木のひとつをR^*とする。R^*の根の子に繰返し情報節点があればそれを展開する。根の子に繰返し情報節点が無くなるまでこれを繰返して得られる木をQとする。このときQの根N_Qのi番目の子を根とする部分木R_i^*はT_iを表現する最小繰返し表現木である。R_i^*をR_iに置き換えてできる木Q'に、QからR^*への変換と同じ縮約を施すことでできる木R'もまた、Tを表現する最小繰返し表現木である。つまり、Q'に縮約を施すことでできる節点数最小の繰返し表現木は、Tを表現する最小繰返し表現木である。ここで、各R_iはそれ以上縮約されないので、節点数を重みとしたラベルを持つ単一の節点とみなすことができ、高さ1の木に対する節点重み和最小の繰返し表現木算出問題に帰着できる。8行目のMinH1Treeによって得られた節点重み和m_RはR^*の節点数に等しく、MinH1Treeの出力するN_Rを根とする木において、ラベルl_iの葉節点をR_iで置換した木Rは節点数がm_Rの繰返し表現木である。よってTの高さがhの場合も命題が成り立つ。 □

命題 4 節点数 n の任意の木に対し、*Conv2MinRRTree* の時間計算量は $O(n^3)$ である。

(証明) 入力木 T の高さ h に関する数学的帰納法で証明する。$h = 0$ のとき、明らかに $O(1)$ である。$h \geq 1$ のとき、$h < h_0$ を満たす高さ h の木に対する ConvMinRRTree の計算量がある定数 c に対して高々cn^3 時間であると仮定する。T の高さを $h = h_0$ とする。T の深さ 1 の節点を根とする部分木を $T_1, T_2, ..., T_k$、T_i の節点数を n_i とする。仮定より T_i に対する Conv2MinRRTree の計算量はある定数 c_2 に対して高々cn_i^3 時間である。Labeling の計算量はある定数 c_2 に対して高々c_2nk 時間であり、葉節点 k 個の木に対するMinH1Tree の計算量はある定数 c_3 に対して c_3k^3 時間であり、葉節点 k 個の置換にかかる計算量はある定数 c_4 に対し高々c_4k 時間であるから、T に対する Conv2MinRRTree の計算量 time は高々 $\sum_{i=1}^{k} cn_i^3 + c_2nk + c_3k^3 + c_4k$ 時間である。$c \geq c_2, c_3, c_4$ とすると、$\sum_{i=1}^{k} n_i^3 + nk + k^3 + k \leq n^3$ より、time$\leq cn^3$ が成り立つ。よって、ConvMinRRTree の時間計算量は $O(n^3)$ である。 □

4. 考察

本節では、繰返し構造を発見する既存手法と提案手法の違いについて考察する。

本手法は、全体の節点数の最小化という指標を用いて繰返し構造を発見するため、局所的な指標を用いた既存手法に比べ、大局的に尤もらしい繰返し構造が発見できると考えられる。

南野らは、タグ列から繰返し構造を発見することで Web ページの構造化を行う手法を提案した [3]。この手法では、範囲の重複した繰返し構造が複数発見された場合に、範囲の重複しない組み合わせのうち繰返し回数の和が最大となるものを選択する。ただし、入れ子状の繰返し構造に対して、内部に繰返し構造を含まないものから順に発見するという処理を繰返すことで構造を発見するため、最終的な繰返し回数の和の最大化は保証されず、その点で局所的な判断による手法といえる。具体例として、入れ子状の繰返し構造を持つ Web ページ (図 6) を考える。南野らの手法では、入れ子の外側の繰返し構造より先に、境界部分 (Price1 と Name2) が繰返し構造として選択される。そのため、外側の 2 回の繰返しを発見できなくなるという問題がある。一方、本論文で提案したアルゴリズムは、全体の節点数が最小であることを保証した完全に大局的な指標を用いるため、このような入れ子状の繰返し構造も正しく発見することが可能である。

Liu らの手法 [2] は、Web ページの DOM 木に対し、類似度を用いた繰返し構造の発見を行い、構造化された情報を抽出する手法である。この手法では、範囲の重複した複数の繰返し構造が得られた場合、繰返し構造が広い範囲にわたっているものを優先的に選択する。例として、子のラベルが順に abaababaab となっている高さが 1 の木を考える。この場合、abaab が二回繰返されているとみることができ (図 7(1))、また aba が二回繰返されたものとみることもできる (図 7(2))。全体の繰返し構造の把握という観点からは前者がより尤もらしい解釈であると考えられる。一方、さらに baab が追加された abaababaabbaab という子を持つ木を考えると、同様に二通りの解釈が考えられるものの (図 8)、この場合は aba の二回の繰返しと baab の二回の繰返しという解釈 (図 8(2)) のほうが尤もらしいと考えられる。Liu らの手法では、繰返し範囲の広いものを優先して選択するため、どちらの場合でも abaab の二回の繰返しという解釈が選択される。これに対し、本手法では節点数の最小化という指標より、それぞれ図 7(1)、図 8(2) が選択される。このように、部分的には同じ構造でも大局的にみると尤もらしい解釈が異なる場合があり、節点数最小化という指標を用いればそのような場合にも本質的な繰返し構造の発見が期待できると考えられる。

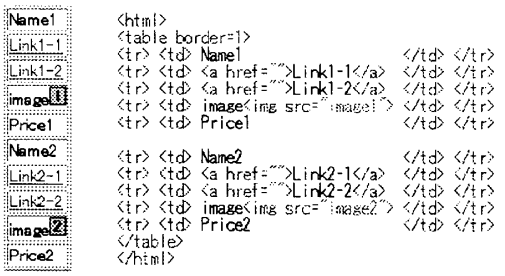

図 6: 入れ子状の繰返し構造を持つ Web ページ

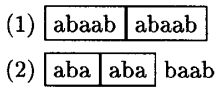

図 7: 例 1

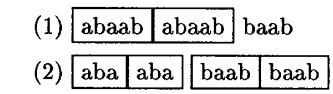

図 8: 例 2

5. まとめ

本論文では、木の繰返し構造を簡潔かつ明示的に表す表現を提案した。また、節点数 n の木に対して、その表現による節点数最小の木を $O(n^3)$ で求めるアルゴリズムを示した。Web ページの DOM 木に対して最小繰返し表現木を算出する予備実験によると、日本の主な検索サイトや電子商取引サイトでは平均して節点数は約 45%に減少し、目視によって繰返し構造とみなしたもののうち約 78%を発見することができた。

既存手法 [1–3] では類似した構造の繰返しも発見できるが、本手法では完全一致の繰返し構造しか発見できない。応用のためには類似した繰返し構造の把握も必要であると考えられ、そのための手法の検討が今後の課題として挙げられる。

参考文献

[1] C-H. Chang and S-C. Lui, IEPAD: Information Extraction Based on Pattern Discovery. *Proceedings of the 10th International Conference on the World Wide Web*, 2001, 681–688.

[2] B. Liu, R. Grossman and Y. Zhai, Mining Data Records in Web Pages. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, 601–606.

[3] 南野朋之, 齋藤 豪, 奥村 学, 繰返し構造に基づいた Web ページの構造化. 情報処理学会論文誌, **45**, 9(2004), 2157–2167.