1C3-2

構造データからの頻出多ポート項木パターン枚挙アルゴリズム

An Algorithm for Enumerating All Frequent Term Tree Patterns Having Multi-ports Variables without Duplication using Succinct Data Structures

糸川 裕子 *1　　内田 智之 *2
Yuko ITOKAWA　　Tomoyuki UCHIDA

*1広島国際大学
Hiroshima International University

*2広島市立大学大学院
Hiroshima City University

For analyzing tree-structured data such as web documents, it is necessary to develop a memory-efficient graph mining method that extracts characteristic tree structures. In order to express tree structured features of tree-structured data, we introduce a term tree pattern with multi-ports variables as an ordered tree pattern which has richer expressive. First we propose a succinct data structure for a term tree pattern with multi-ports variables by modifying a depth-first unary degree sequence (DFUDS), which is one of the succinct data structures for an ordered tree. Next we present a pattern matching algorithm, that uses the DFUDS succinct data structure, for determining whether or not a given tree-structured data has tree-structured features described by a given term tree pattern. Finally we give an algorithm for enumerating all frequent term tree patterns which represent tree-structured features common to given tree-structured data.

1. はじめに

効率的な木構造データからのデータマイニングツール構築のためには，高速かつメモリ効率のよいマイニングアルゴリズムが必要である．我々は 2 頂点から成る変数をもつ項木パターンに対する高速なパターンマッチングアルゴリズム [Itokawa 11] と頻出項木パターン枚挙アルゴリズム [Itokawa 14] を提案した．本研究では，これらを拡張し，より表現能力の高い $k(k \geq 2)$ 個の頂点から編成される変数をもつ木パターン (多ポート項木パターン) の簡潔データ構造を与え、多ポート項木パターンに対するパターンマッチングアルゴリズムおよび頻出多ポート項木パターン枚挙アルゴリズムを提案する．

2. 多ポート項木パターン

Λ と χ を $\Lambda \cap \chi = \emptyset$ であるような有限のアルファベットとする．V_t を頂点集合，$E_t \subseteq V_t \times \Lambda \times V_t$ をラベル付き辺集合とする．また，$H_t \subseteq V_t \times \chi \times V_t^+$ を変数集合とする．このとき、3 つ組 $t = (V_t, E_t, H_t)$ で表される辺ラベル付き順序木パターンを**多ポート項木パターン**という．辺 $e = (u, a, v) \in E_t$ において文字 $a \in \Lambda$ を e の辺ラベルという．変数 $h = (u, x, v_1, \cdots, v_k) \in H_t$ において，$x \in \chi$ を h の変数ラベル，u を h の**親ポート**，$v_1, \cdots, v_k$ を h の**子ポート**という．以後，変数を持たない多ポート項木パターンを**順序木**ということにする．図 1 に多ポート項木パターン $t = (\{v_0, \ldots, v_6\}, \{(v_1, a, v_2), (v_3, b, v_4), (v_3, c, v_5)\}, \{(v_0, x, v_1, v_3, v_6)\})$ を与える．

順序木に対する簡潔データ構造のひとつに **DFUDS**(depth-first unary degree sequence) **表現** [Munro 05] がある．順序木 T の DFUDS 表現は、各頂点を根から行きがけ順に走査し、次数個の (と 1 つの) で符号化して並べることでで得られる．つまり，辺数 m の順序木の DFUDS 表現は m 個の (と m 個の) から成る長さ $2m$ の括弧列である．また，) が表す頂点とその親との間にある辺ラベルを返すハッシュ関数を与えることで，辺ラベル付き順序木に対する DFUDS 表現に拡張できる [Itokawa 11]．ここで，根に対応する) は特別な辺ラベル # を返すこととする．多頂点からなる変数 h は h を示す 1 つの頂点 v_h(図 1 中では□で表される) および h の親ポートとの間のラベル $x \in \chi$ をもつ辺，各子ポート間のラベル"$" をもつ辺を対応させる．これにより，図 1 の多ポート項木パターン t の DFUDS 表現は ((# (((x ($ a (($ b c $，順序木 T の DFUDS 表現は (((# ((a (c a ((a b c b となる．

連絡先: 糸川 裕子，広島国際大学，東広島市黒瀬学園台 555-36，0823-70-4880，y-itoka@he.hirokoku-u.ac.jp

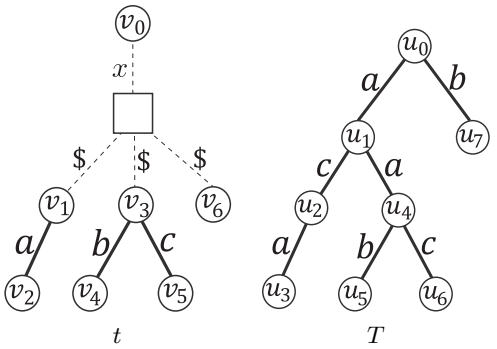

図 1: 多ポート項木パターン t と順序木 T

3. 多ポート項木パターン枚挙アルゴリズム

多ポート項木パターン t の変数を適切な順序木で置き換えることで、辺ラベル付き順序木 T が得られるとき，t と T は**マッチする**という．このとき，次のような問題を考える．

多ポート項木パターンマッチング問題

入力: 多ポート項木パターン t と順序木 T

問題: t と T がマッチするか否かを判定せよ．

データ構造として DFUDS 表現を用いた，多ポート項木パターンマッチング問題を解くアルゴリズム $MPTTP\_Matching$ をアルゴリズム 1 に示す．DFUDS 表現 D に対し，操作 $D.id(p)$ は D の p 番目の文字が表す木構造データの頂点を返し，操作 $D.next(p)$ は p 番目の文字が表す木構造データの頂点の，行きがけ順で次に現れる頂点を表

Algorithm 1 $MPTTP\_Matching$

Require: 多ポート項木パターン t の DFUDS 表現 $P[0:m-1]$，順序木 T の DFUDS 表現 $T[0:n-1]$
Ensure: t と T がマッチするとき true，そうでないとき false

```
1: i := 0; j := 0
2: while i < m ∧ j < n do
3:   if P[i] ≠ T[j] then
4:     if (P[i] ∈ Λ) ∧ (T[j] ∈ Λ) then
5:       return false
6:     end if
7:     P' = P.id(i) を根とする部分木の DFUDS 表現
8:     T' = T.id(j) を根とする部分木の DFUDS 表現
9:     if (P[i] ∈ Λ) ∧ (T[j] == "(") then
10:      if MPTTP_Matching(P', T') = false then
11:        return false
12:      else
13:        i := P.next(i); j := T.next(j)
14:      end if
15:    else
16:      if Variable_Matching(P', T') =false then
17:        return false
18:      else
19:        i := P.next(i); j := T.next(j)
20:      end if
21:    end if
22:  end if
23:  i++; j++
24: end while
25: if (m − i > 0) ∨ (n − j > 0) then
26:   return false
27: end if
28: return true
```

す D の文字の位置を返す．操作 $Variable\_Matching(t', T')$ は，多ポート変数の親ポート u を根とする t の部分木 t' と，頂点 v を根とする T の部分木 T' がマッチするか否かを判定する関数である．

データ構造として DFUDS 表現を用いることにより，マッチングアルゴリズム $MPTTP\_Matching$ は省メモリ化・高速化が図られている．長さ n の括弧列に対する DFUDS 表現 S において，次のように定義される $rank$ 関数と $select$ 関数は $n+o(n)$ ビットの領域を使って定数時間で実行可能である．(1)$rank_c(S,i)$ は $S[0\ldots i]$ 中での c の出現回数を返す関数である．(2)$select_c(S,i)$ は S 中で先頭から i 番目の c の位置を返す関数である．順序木に関する基本操作の多くはこの $rank$ 関数と $select$ 関数を使って定数時間で計算できる．

次に，多ポート項木パターンの枚挙問題について説明する．要素数 s の順序木の集合 S，多ポート項木パターン t，実数 $\sigma(0<\sigma\leq 1)$ に対し，t とマッチする順序木 $T\in S$ の数が q のとき，$q/s\geq\sigma$ ならば，t は σ **頻出**であるといい，σ を S に対する t の最小支持率という．このとき以下の問題を考える．

頻出多ポート項木パターン枚挙問題

入力: 順序木の集合 S，最小支持率 $\sigma(0<\sigma\leq 1)$

問題: S に関し σ 頻出多ポート項木パターンを全て枚挙せよ．

頻出多ポート項木パターン枚挙問題を解くアルゴリズム Enu_k-OTT をアルゴリズム 2 に示す．多ポート項木パターン t に対する関数 RightmostExpansion(t) とは、t の最右パス上の頂点から子ポート数 1 の変数を 1 つ末子に追加する、あるいは変数に子ポートを末子に追加した多ポート項木パターンを返すものである．また，枚挙された σ 頻出多ポート項木パターン F を σ 頻出多ポート項木パターンの枚挙手順を示す枚挙木で管理することにより，すべての σ 頻出多ポート項木パターンを効率的かつ重複なく正しく枚挙できる．

Algorithm 2 Enu_k-OTT

Require: 順序木の集合 S，　最小支持率 $\sigma(0<\sigma\leq 1)$
Ensure: S に関して σ 頻出な多ポート項木パターンの集合 F

```
1: 1 つの変数から成る項木パターンを p とする
2: F ← {p}, E ← ∅
3: while F が空でない間 do
4:   C ← ∅, D ← ∅
5:   for all t ∈ F do
6:     C ← RightmostExpansion(t)
7:     for all f ∈ C do
8:       if f は S において σ 頻出 then
9:         E ← E ∪ {f}, D ← D ∪ {f}
10:      end if
11:    end for
12:  end for
13:  F ← D
14: end while
15: A := F
16: while A ≠ ∅ do
17:   for t ∈ A do
18:     C ← t の 2 頂点からなる変数を辺で置き換えた項木パターン
19:     A := {t ∈ C | t はσ頻出 }
20:     F := F ∪ A
21:   end for
22: end while
23: return F
```

4. 終わりに

多ポート項木パターン t とマッチするすべての順序木の集合を $L(t)$ で表す．項木パターンの集合 S に対し，$S\subset L(s)\subsetneq L(t)$ となる多ポート項木パターン s が存在しない場合，t は S に関して**極小である**という．木構造データ D に関する効果的なデータマイニングシステムを構築するために，D に関して極小な多ポート項木パターンを枚挙する手法の開発が今後の課題である．

参考文献

[Munro 05] D. Benoit, E. D. Demaine, J. I. Munro, R. Raman: Representing Trees of Higher Degree. *Algorithmica,* 43(4):275-292, 2005.

[Itokawa 14] Y. Itokawa, M. Sano and T. Uchida: An Algorithm for Enumerating All Maximal Tree Patterns Without Duplication Using Succinct Data Structure. IMECS 2014, pp.156–161, 2014.

[Itokawa 11] Y. Itokawa, M. Wada, T. Ishii and T. Uchida: Pattern Matching Algorithm Using a Succinct Data Structure for Tree-Structured Patterns. LNEE 110. Springer, pp.349-361, 2011.