有向グラフ上の到達可能性判定のための索引構造と大規模XMLデータへの応用

中村　有作†　　舞田　哲哉†　　坂本　比呂志††

† 九州工業大学 情報工学研究科
†† 九州工業大学 情報工学部
〒 820-8502 福岡県飯塚市大字川津 680-4
E-mail: {yuusaku, t_maita}@donald.ai.kyutech.ac.jp, hiroshi@ai.kyutech.ac.jp

あらまし　本研究では，巡回や合流などの構造を含む有向グラフで表現される半構造データに対して，任意の 2 ノード間の到達可能性を高速に判定することができる規模耐性の高い索引構造を提案する．有向グラフが木構造の場合，到達可能性の判定は，範囲ラベルなどの値をあらかじめ設定しておくことで高速に判定可能である．しかしながら，一般の有向グラフに対しては，このような単純な索引付けが困難である．そこで本研究では，一般の有向グラフに適用可能な到達可能性判定のための索引構造を提案し，その有効性を実験によって検証する．特に，比較的大規模な XML データにおいて，本研究の手法が他手法よりも，前処理に必要な主記憶量，計算時間および索引サイズにおいて優れていることを示す．

キーワード　XML, 範囲ラベル, 2Hop ラベル, 先祖子孫関係, 非巡回有向グラフ

Indexing for Reachability Test on DAGs and its Application to Large XML Data

Yuusaku NAKAMURA†, Tetsuya MAITA†, and Hiroshi SAKAMOTO††

† Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology
†† Faculty of Computer Science and Systems Engineering, Kyushu Institute of Technology
Kawazu 680-4, Iizuka-shi, Fukuoka 820-8502
E-mail: {yuusaku, t_maita}@donald.ai.kyutech.ac.jp, hiroshi@ai.kyutech.ac.jp

Abstract　We propose an efficient algorithm for deciding the reachability between any nodes on XML data represented by connected directed graphs. In case of tree structures, the reachability is testable in a constant time by preprocessing the range labels of all nodes. However, in case of general directed graphs, it is impossible to set a simple range label for the graph even if it is acyclic. In this study, we introduce a new index for the problem of the reachability test on the general graphs. we show that our algorithm answers almost queries in a constant time preserving the space efficiency and a reasonable preprocessing time compared with other methods.

Key words　XML, range label, 2Hop label, ancestor-descendant relationship, directed acyclic graph

1. はじめに

本研究では，有向グラフで表現される XML データ上の任意の 2 ノード間の到達可能性を高速に判定するための高い規模耐性を持つ索引構造を提案する．XML は制約の緩やかなフォーマットであるため，様々な場所で自由に記述・蓄積されており，このような XML から目的のデータを高速に取り出すために，XQuery [7] などの XML のための問い合わせ言語が提案されている．これらの XML データは順序木や有向グラフの構造を持っているため，問い合わせ言語においては，任意のノード間の先祖子孫関係や，より一般的なグラフ上の到達可能性を高速に判定する技術が必要である．そこでまず，これまでに提案されている，本研究と関連が深いグラフ構造に対するラベル付けの研究をまとめる．

1.1 範囲ラベル

木構造データ上の先祖子孫関係は，範囲ラベルと呼ばれる索引構造で高速に判定できる．範囲ラベルは，データとなる順序木の各ノードに前置順と後置順の順位の組を設定したものであ

る．このとき，2 つのラベル (x, y), (u, v) に対して，$x < u$ かつ $y > v$ であることと (x, y) が (u, v) の先祖であることは等しく，(x, y) は (u, v) を包含するという．このような範囲ラベルを前処理として計算することで，任意のノード間の先祖子孫関係を定数時間で判定できる．

これまでに，範囲ラベルを利用した様々なアルゴリズムが提案されている [2], [13], [14], [31]．一方，一般の XML データを記述するためには，順序木よりも広いクラスである非巡回グラフのクラスまで拡張する必要がある(注1)．しかしながら，非巡回グラフは辺の合流を含むので，単純な範囲ラベルでは非巡回グラフ上の無矛盾な先祖子孫関係を表現することができない．実際に，どのような範囲ラベルを用いても正しい到達可能性を表現することが不可能な非巡回有向グラフが存在する [22]．

このような問題点を解決するために，これまでに様々な工夫が提案されている．非巡回グラフの範囲ラベル付けとして [4] がある．この手法では，グラフの全域木を求め，それに対して範囲ラベルを設定する．そしてすべての葉から根に向かって木以外の辺も辿ることで範囲ラベルの伝播を行う．このとき，あるノード u のすべての範囲ラベルが，ノード v のある範囲ラベルに包含されるときまたそのときに限り v から u へ到達可能となる．この手法による判定時間は，各ノードに設定される範囲ラベルの最大数 p に依存する．グラフのノード数と辺数をそれぞれ n, m とすると，必要な前処理時間は $O(pm)$ であり，記憶領域は $O(pn + m)$ である．また，範囲ラベルがソートされていると仮定すると，1 回あたりの判定は $O(p)$ 時間となるが，最悪の場合 $p = O(n)$ となる．

この欠点を改善する手法として，重なり範囲ラベルと多次元分割による判定法が提案されている [29]．これは元の非巡回グラフをいくつかの連結成分に分解する手法である．これらの連結成分は，それぞれが単一の範囲ラベルで到達可能性を保持できる極大部分グラフとなっており，同じ成分内のノードについては定数時間で到達可能性を判定できる．この手法では，1 回あたりの判定時間は，多次元分割の個数 q に比例し，この値は，[4] の手法における p に比べて比較的小さいことが示されている [29]．この手法における前処理時間は $O(qn^2)$，記憶領域は $O(q(n + m))$ である．

[4] の伝播法は，前処理の時間は抑えられるものの，複雑なデータでは 1 ノードあたりの範囲ラベルの個数が急激に増加することが報告されている [29]．また [29] の手法では，判定コストは抑えられるがグラフの多次元分割に要するコストが高いため大規模データに対する実装が難しい．

前処理時間と判定コストの両方を妥当な範囲に抑えるために，[22] では到達可能性を記録する参照テーブルを分割する方法を提案している．この手法では，通常にみられるような深さがそれほど深くない XML データに対して有効で，すべてのノードの組に対して到達可能性を記録する方法と比較して，テーブルのサイズを 1%未満にすることに成功している．しかしながら，テーブルを分割してもそのサイズはデータサイズの 2 乗に比例して大きくなるので，データベースの大規模化は困難である．

1.2 2Hop ラベル

範囲ラベルは木構造データに対して有効な手法であるが，一般のグラフに対する到達可能性を判定する手法として，2Hop ラベル法が提案されている [12]．この手法でも各ノードにラベルを持たせるが，二つのノードが共通のラベルを持つか否かによって到達可能性を判定するという点が範囲ラベルとは異なる．すなわち，各頂点 v は 2 つのラベルの集合 v_{in}, v_{out} を持ち，これらを v の 2Hop ラベルと呼ぶ．ここで，v_{in} は v に到達可能なノードの集合であり，v_{out} は v から到達可能なノードの集合である．

いま，u, v の 2Hop ラベルが，可能なすべてのノードをラベルとして含んでいるとしよう．すなわち，v_{in} は v に到達可能なすべてのノードからなる集合である．このとき，u から v へ到達可能であることと $u \in v_{in}$ であることは等価である．しかしながらこのようなラベルを構成することは効率的ではない．2Hop ラベルの利点は，各頂点の 2Hop ラベルを適当に間引いても到達可能性が判定できる点にある．すなわち，ある適当な 2Hop ラベル u_{out}, v_{in} に対して，u から v へ到達可能であることと $u_{out} \cap v_{in} \neq \emptyset$ であることが等価となる．これは，u から v へ辿る途中のノードのうち少なくとも 1 つを u_{out} と v_{in} が共通に持っていればよいことを表している．与えられた有向グラフに対して，最小の 2Hop ラベルを計算する問題は NP-困難であるが，[12] では，この問題を近似的に解くアルゴリズムが提案されており，最適解に対して $O(\log n)$ 倍以内の 2Hop ラベルが得られることが保証されている．

[24] は，この手法を改良し，ファイルを分割することで，大規模なデータに対する実装が可能な HOPI アルゴリズムを提案している．判定時間はラベル数に比例するが，各ノードの平均ラベル数はそれほど大きくならないことが実験によって示されている．ところがこの手法では，分割した結果を統合することで処理時間が増加すること，あらかじめ到達可能なノードの組を候補集合として計算しておかなければならないことなどの主に計算時間と主記憶量に関する欠点がある．[24] では，データを適当に分割して，計算結果を統合することで全体の XML に対する 2Hop ラベルを構築し，これらの欠点をある程度補っている．

1.3 提案手法

本研究では，このような有向グラフの到達可能性判定問題に対して，これまでに述べた範囲ラベルと 2Hop ラベルを組み合わせた索引構造を提案し，前処理時間と主記憶量に関するコストを妥当な範囲に抑えながら，ノード間の到達可能性を高速に判定するアルゴリズムを実装する．特に本提案手法では，2Hop ラベルで必要な $O(n^2)$ サイズの候補集合全体をあらかじめ計算する必要がないため，データを分割することなく実用的な時間でラベルを構築可能である．

ここで，本研究が対象とする XML とそれが表す構造を説明する．XML は，より一般的な概念である半構造データに含まれる．半構造データは，形式的には，属性とその値のペアから

(注1)：巡回を含む連結な有向グラフは，強連結成分に分解することで非巡回グラフの場合に還元できる．これについては後述する．

なる列あるいは集合であり，値は文字列や数値などの基本データ型または半構造データであると定義される．本研究が対象とする XML では，このような構造は，次のようにラベル付きグラフで表現される．属性はグラフのノードに対応し，属性間の二項関係がノード間の辺に対応する．すなわち，属性が直近の入れ子関係にあるときや，特定の異なる属性が同じ値を持つときに，そのノード間に辺を定義する．XML では半構造データの属性のことを要素と呼ぶ．本論文では，要素間の入れ子によって定義される辺を実辺，要素の値によって定義される辺を参照辺と呼ぶ．

参照辺は要素に付加的に定義される特定の属性とその値の組によって定義される．例えば XMark [25] では，ある要素の ID 型属性と別の要素の IDREF 型属性が同じ値を持つことで，この 2 要素間の参照辺を実現している．この参照辺は，XLink で記述される単純リンクに相当する．

これらの XML の概念は，グラフの概念に自然に関連付けることができる．与えられた XML 全体 D がある有向グラフ G に対応し，D の実辺からなる構造は，G のある全域木 T に対応する．また，D の参照辺は，$G-T$ に含まれる辺に対応する．そこで以降では，特に断りがない限り，XML データ D とそれが表す有向グラフ G を同一視することにする．

本研究で提案する手法は，範囲ラベルと 2Hop ラベルを組み合わせた手法である．まず，与えられた有向グラフ G を強連結成分分解し，非巡回有向グラフ G' を計算する．G' の各ノードは，G の各強連結成分に対応するため，同じ成分内のノードは互いに到達可能な関係にある．したがって，G' のノードに対するラベル付けを行えばよい．そこで，G' に対して [4] と同様に，深さ優先探索などの標準的な手法で非巡回グラフの全域木を求めて，その全域木に対する範囲ラベルを計算する．そして，範囲ラベルで判定できないが互いに到達可能な関係にあるノードに対して 2Hop ラベルを計算する．この 2Hop ラベルの計算は, [24] のものとは異なり $O(\log n)$ 倍以内の近似を保証しないが，$O(n^2)$ のサイズの候補集合全体をあらかじめ計算する必要がないため，ファイルを分割する必要もなく，実データにおいては主記憶量，前処理時間および判定時間に対する効率がよいことが実験によって示される．

本論文の構成は以下のようになっている．2 節では基本的概念をまとめる．3 節では，本研究で提案するアルゴリズムとその正当性を示す．4 節では，本手法と HOPI を実装し，前処理時間と判定時間を比較し，有効性を示す．5 節では，本研究の成果と今後の課題をまとめる．

2. 準　　備

本研究では，連結な有向グラフ全体(注2)を対象としている．有向グラフ G に対するある全域木 T を求めたとき，$G \cap T$ の辺を T によって定まる G の実辺，それ以外の $G-T$ の辺を参照辺と定義する．一般に G の全域木は複数存在するが，混乱が生じない限り，単に G の実辺および参照辺と呼ぶことにする．

(注2)：連結でないグラフ間はあきらかに到達可能でないので，非連結グラフまで拡張する必要はない．

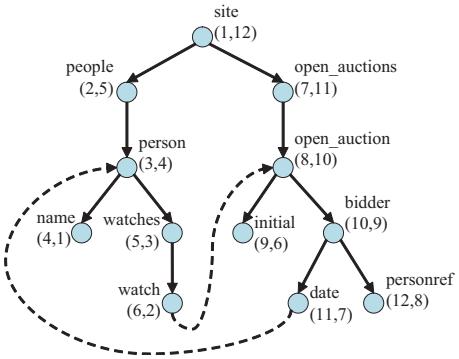

図 1　**連結な有向グラフとその全域木に設定された範囲ラベル**: グラフ中の実線が全域木の辺 (実辺) を表し，破線がそれ以外の辺 (参照辺) を表す．各ノードは，この全域木に対して定まる前置順・後置順のペア (範囲ラベル) を持っている．この範囲ラベルは全域木内の到達可能性を判定できる．

2.1 範囲ラベル

［**定義 1**］ 有向グラフ G の任意のノード v に，ある非負整数のペア $(x(v), y(v))$ を割り当てたとき，任意の v, u に対して，$x(v) < x(u)$ かつ $y(v) > y(u) \Leftrightarrow v$ が u の先祖であるとき，この割り当てを G に対する範囲ラベルという．

G が木や平面グラフなど単純な構造であるときには，このような性質を満たす範囲ラベルは高速に計算することができる．例えば木の場合には，そのノード v の前置順と後置順による順位のペア $(pre(v), post(v))$ を利用する手法がよく知られている．このようにしてラベル付けされた例を図 1 に示す．

より一般には，各 $pre(v)$ や $post(v)$ の間隔を空けて設定することで，範囲ラベルの条件を満たしながらデータの挿入に対応することができる [14]．したがって，それらすべてを広義の範囲ラベルと考える．このような範囲ラベルを設定することで，木の任意の 2 ノードについて，一方が他方の先祖であるか否かを 2 つの整数の比較のみで判定できる．

しかしながら，図 1 の例では，”watch” と”date” の 2 ノードが先祖子孫関係にあるにも関わらず，これらの範囲ラベルは定義 1 を満足しない．一般には，有向グラフに対する無矛盾な範囲ラベルは存在しないため，有向グラフの到達可能性の判定には別の手法が必要である [22]．次に，それらの関連研究について簡単に説明する．

2.2 非巡回グラフへの制限

巡回を含む連結な有向グラフに関する到達可能性の判定は，以下のように非巡回なものに制限した場合に還元できる．任意の有向グラフは，そのグラフに対する深さ優先探索を 2 回行うことで，そのグラフを強連結成分に分解できる (このアルゴリズムについては [5] が詳しい)．ここで強連結成分とは，互いに到達可能なノードだけで構成された部分グラフのことであり，ノード x, y が同一の強連結成分内に存在するならば，任意の z に対して，z, x 間の到達可能性と z, y 間のそれは等価である．

したがって，1 つの強連結成分に含まれるノード全体を 1 つの頂点とみなしてよく，再構成される有向グラフは非巡回である．このような例を図 2 に示す．

したがって，一般の有向グラフに対する到達可能性に関する索引付けは，まず与えられたグラフを強連結成分に分解し，その非巡回グラフ (DAG) に関する到達可能性についてのラベル付けを行えばよい．以降では，DAG 内における先祖子孫関係の概念を木の場合と同様に使用する．すなわち，x から y への有向パスが存在するとき，x は y の先祖，y は x の子孫という．

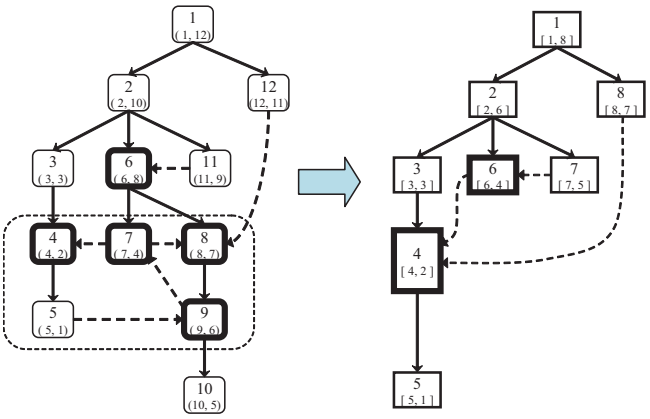

図 2 **被参照ノードを持つ有向グラフの強連結成分分解の例**: 左の有向グラフの破線で囲まれた強連結成分が 1 つのノードに縮約されて右の DAG に変換される．各ノードには深さ優先探索による ID と範囲ラベルが示されている．また，その探索によって実辺と参照辺が定まるが，破線矢印が参照辺を表し太枠ノードが被参照ノードを表す．

2.3 2Hop ラベル

範囲ラベルは木構造データに対して有効な手法であるが，一般の有向グラフに対する到達可能性を判定する手法として，2Hop ラベル法が提案されている [12]．

[定義 2] 有向グラフ $G = (E, V)$ の各ノード $v \in V$ に対して，ラベルの集合 $v_{in}, v_{out} \subseteqq V$ を決めたとき，$L(G) = \{(u_{in}, u_{out}) \mid u \in V\}$ を G の 2Hop ラベルという．このとき，任意の $u, v \in V$ に対して，u から v へ到達可能 $\Leftrightarrow$ $u_{out} \cap v_{in} \neq \emptyset$ であるなら，そのような $L(G)$ を G の 2Hop カバーという．

以降では，2Hop カバーの条件を満たす $L(G)$ のことを単に 2Hop ラベルと呼ぶことにする．任意の連結な有向グラフに対に対して，2Hop ラベルが存在することはあきらかである．なぜならば任意の $v \in V$ に対して，v_{in} を v に到達可能なすべてのノードからなる集合とし，v_{out} を v から到達可能なすべてのノードからなる集合とすればそのラベルは常に定義 2 を満足するからである．

しかしながらこのようなラベルを構成することは効率的ではない．2Hop ラベルの利点は，各ノードのラベルを適当に間引いても到達可能性が判定できることである．これは，u から v へ辿る途中のノードのうち少なくとも 1 つを u_{out} と v_{in} が共通に持っていればよいことを表している．各ノードがなるべく

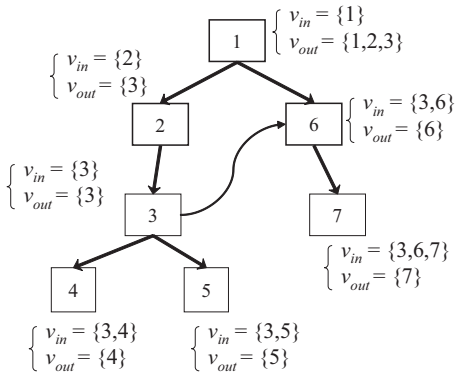

図 3 **自明でない 2Hop ラベルの例**: u から v へ到達可能 $\Leftrightarrow$ $u_{out} \cap v_{in} \neq \emptyset$ を満たし，不要なラベルをなるべく含まない．

小さい数のラベルを持つような 2Hop ラベルを求めることができれば，到達可能性を判定するための索引として都合がよい．図 3 にそのような 2Hop ラベルの例を示す．このグラフの各ノードには v_{in}, v_{out} が設定されており，これは定義 2 の条件を満たすが，例えばノード 1 はすべてのノードに到達可能であるにもかかわらず，その v_{out} はノード $1, 2, 3$ のみを含む．このように，可能な v_{in}, v_{out} からラベルを間引いても，2Hop ラベルの条件を満たすものが存在する．

しかしながら，与えられた有向グラフに対して，最小の 2Hop ラベルを計算する問題は NP-困難である．[12] は，この問題を近似的に解くアルゴリズムを提案し，最適解に対して $O(\log n)$ 倍以内のサイズの 2Hop ラベルを効率的に計算できることを示している．また，[24] では，ファイルを分割することで大規模データに適用した結果を示している．これらのアルゴリズムの詳細は，本研究で提案する手法とは直接関係しないため省略する．本研究ではこれらとは異なる 2Hop ラベル構築アルゴリズムを提案し，本手法と従来の範囲ラベルを組み合わせた新しい索引構造の有効性を実験によって示す．

3. 提案手法

本節では，提案アルゴリズムによるラベルの構築法とその改良を示す．

3.1 ラベル構築アルゴリズムと到達可能性の判定

本研究で提案する到達可能性判定のためのラベル構築アルゴリズムは，3 つのフェーズからなっている．まず第 1 フェーズでは，与えられた有向グラフ $G = (V, E)$ を強連結成分分解し，非巡回有向グラフ G' に変換する．ここで，同一成分内に含まれるノードに対しては，その成分情報を配列に記憶する．そのための時間計算量および領域計算量は $O(|E|)$ および $O(|V|)$ である．第 2 フェーズでは，G' を深さ優先探索し，その全域木 T を計算し，T の範囲ラベルを設定する．もし，$T = G'$ ならば索引付けは終了する．このための計算量は，第 1 フェーズと同じである．第 3 フェーズでは，各ノードの 2Hop ラベルを計算する．本研究の提案手法は，各ノード v に対して，v_{in}, v_{out} が被参照ノードのみを要素として持つ点が，これまでの手法と異なる．通常は被参照ノードは，ノード全体に対してあまり大

アルゴリズム $V_{IN}(T)$
入力：連結な非巡回有向グラフ $G = (V, E)$ の全域木 T
出力：任意の $v \in V$ に対するラベル集合 v_{in}
(1) T を深さ優先探索：(v をカレントノード，p をその親とする)
　(1-1) v が被参照ノードならば，$v_{in} \leftarrow v_{in} + \{v\}$.
　(1-2) $v_{in} \leftarrow v_{in} + p_{in}$, $v \rightarrow next$.
(2) v_{in} を出力.

図 4　v_{in} 構築アルゴリズム

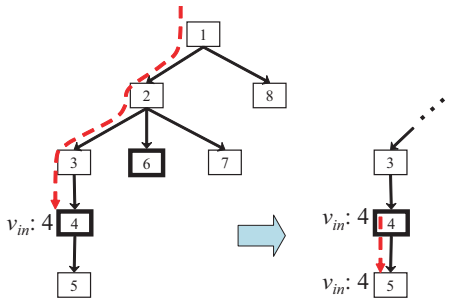

図 5　$V_{IN}(G)$ **の動作例**: 太枠ノードが被参照ノードを表す．参照辺は省略されている．ノード 4 を訪れたとき，被参照ノードであるので，自分自身を自分の v_{in} ラベルとして登録する (左図)．次のノードに行くと，それは被参照ノードではないが，親が v_{in} ラベルを持っているので，それを自分のラベルとして登録する (右図)．

アルゴリズム $V_{OUT}(T)$
入力：連結な非巡回有向グラフ $G = (V, E)$ の全域木 T
出力：任意の $v \in V$ に対するラベル集合 v_{out}
(1) T を深さ優先探索する：(v をカレントノードとする)
　(1-1) 参照辺 $(v, u) \in E$ があれば，$V_{OUT}(T_u)$ を呼び出す．
　ただし，T_u は u を根とする T の部分木を表す．
　(1-2) u が探索済みならば，その親 v へ戻り，
　　$(v, u) \in E$ が参照辺ならば，$v_{out} \leftarrow u_{out} + \{u\}$.
　　$(v, u) \in E$ が実辺ならば，$v_{out} \leftarrow u_{out}$.
(2) v_{out} を出力する．

図 6　v_{out} 構築アルゴリズム

きくないので，このよう制限することで，構築されたラベルのサイズも抑えられると予想される．ここでは，このように制限されたラベルによる 2Hop ラベル構築アルゴリズムとその正当性を示す．

図 4 にラベル v_{in} 構築アルゴリズムを示し，図 5 にその動作例を示す．このアルゴリズム $V_{IN}(T)$ は与えられたグラフの全域木の辺のみを深さ優先探索するので，計算時間は $O(\ell|V|)$ である．ただし，ℓ は被参照ノードの数，すなわち各ノードに継承されるラベルの最大数である．

同様に，図 6 にラベル v_{out} 構築アルゴリズムを示し，図 7 にその動作例を示す．このアルゴリズム $V_{OUT}(T)$ は，グラフの辺を高々一度探索するので，計算時間は $O(\ell|E|)$ である．

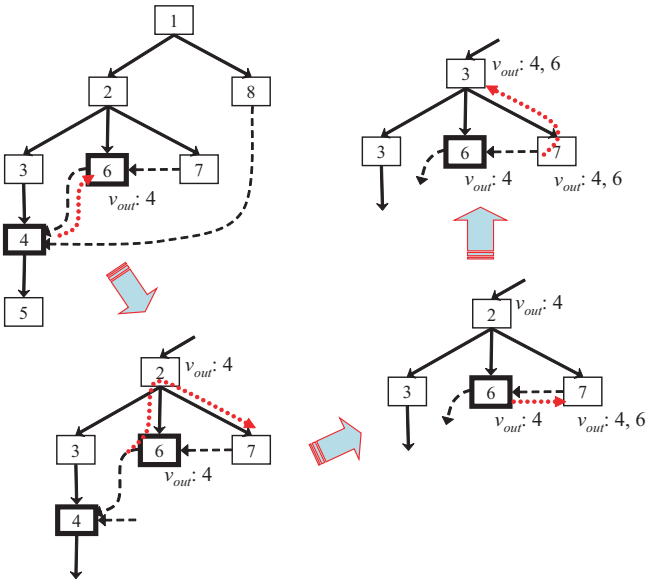

図 7　$V_{OUT}(G)$ **の動作例**: 太枠ノードが被参照ノード，破線矢印が参照辺を表す．点線矢印はアルゴリズムの探索経路を表し，各ステップで，子から親ノードへ必要なラベルの継承を行っている．

以上のアルゴリズム $V_{IN}(T)$ および $V_{OUT}(T)$ を用いて，非巡回有向グラフ $G = (V, E)$ の 2Hop ラベル $\{(v_{in}, v_{out}) \mid v \in V\}$ を計算する．このとき，G の 2 ノード間の到達可能性は，図 8 に示す手続きによって判定される．

到達可能性判定アルゴリズム $Reachability(u, v)$
入力：連結な有向グラフ $G = (V, E)$
前処理: G を強連結成分分解した G'，G' の全域木 T，
　$V_{IN}(T)$ および $V_{OUT}(T)$ の計算
出力: 以下のいずれかのときのみ $true$ (u から v へ到達可能)：
　(1) u, v が同一の強連結成分に含まれる．
　(2) u の範囲ラベルが v のそれを包含する．
　(3) $u_{out} \cap v_{in} \neq \emptyset$.

図 8　有向グラフ上の到達可能性判定アルゴリズム

次に，図 8 の到達可能性判定アルゴリズムの正当性を示す．

［**定理 1**］　任意の連結な有向グラフ $G = (V, E)$ と $u, v \in V$ に対して，$Reachability(u, v) = true \Leftrightarrow u$ から v へ到達可能.

[証明]　強連結成分分解後の非巡回有向グラフについて示せば十分である．

($\Rightarrow$) u の範囲ラベルが v のそれを包含する場合には，u から v へあきらかにパスが存在する．それ以外で，$u_{out} \cap v_{in} \neq \emptyset$ であるとしよう．$x \in u_{out} \cap v_{in}$ とすると，u から x へ到達可能でかつ x から v へ到達可能であるので，u から v へ到達可能である．

($\Leftarrow$) u から v へ到達可能ならば，それは実辺のみを辿るパスか，または少なくとも 1 つは参照辺含むパスのどちらかである．前者の場合はあきらかに $Reachability(u, v) = true$ である．後

者の場合，そのパスに含まれる v に最も近い被参照ノードを x としよう．x から v までの部分パスはすべて実辺からなり，それに含まれる x の子孫 x' は $x \in x'_{in}$ を満たすので，$x \in v_{in}$．また，x の任意の先祖 x' は $x \in x'_{out}$ を満たすので，$x \in u_{out}$．したがって，$u_{out} \cap v_{in} \neq \emptyset$ である． □

以上により，提案手法の正当性が示された．実際に 2Hop ラベルを計算した例を図 9 に示す．

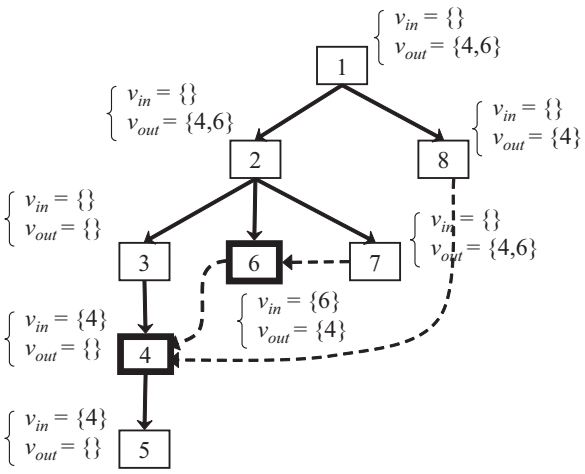

図 9　**提案手法による 2Hop ラベルの例**: 太枠ノードが被参照ノード，破線矢印が参照辺を表す．全域木内での先祖子孫関係が成立しているノードの組については，範囲ラベルで判定可能であり，それ以外の任意の到達可能性をこの 2Hop ラベルで判定できる．

3.2　提案手法の改良

直感的には，v_{in} は v の先祖の被参照ノードの集合を表し，v_{out} は v の子孫の被参照ノードの集合を表す．したがって，浅く広い構造である XML データのようなものほど，v_{out} のサイズが極端に大きくなる傾向がある．これは判定時間の増加に直接影響するため，v_{out} をさらに削減する手法を導入する．まずアルゴリズム $V_{OUT}(T)$ を改良するためのアイデアを説明する．

アルゴリズム $V_{OUT}(T)$ において，あるノード x から実辺や参照辺を辿って親 y に戻るとき，x_{out} のラベルをそのまま y_{out} に継承していた (図 7 を参照)．ここで，y から x_{out} のあるラベルへ到達可能であることが範囲ラベルで判定可能ならば，そのラベルは y_{out} に継承する必要がない．そこで，アルゴリズム $V_{OUT}(T)$ を図 10 のものに改良し，その正当性を示す．

[定理 2]　アルゴリズム $V_{OUT}(T)$ を $V'_{OUT}(T)$ で置き換えても $Reachability(u, v)$ は到達可能性を判定できる．

[証明]　範囲ラベルは変化せず，$V'_{OUT} \subseteqq V_{OUT}$ であるので，x から y へ到達不可能であれば，それはどちらの場合でも判定できる．したがって，x から y へ到達可能であるが，範囲ラベルでは判定できない場合のみを示せば十分である．あるラベル u が存在して，変更前のラベルに対して $u \in x_{out} \cap y_{in}$ が成り立つ．ここで，変更後のラベル x'_{out} に対して，“$u \in x'_{out} \Leftrightarrow u$ は x の子孫の被参照ノードでかつ x の範囲ラベルが u のそれを包含しない” であるので，この場合は $u \in x'_{out}$ が成り立つ．よって，変更後のラベルによっても判定可能である． □

アルゴリズム $V'_{OUT}(T)$
入力：連結な非巡回有向グラフ $G = (V, E)$ の全域木 T
出力：任意の $v \in V$ に対するラベル集合 v_{out}
(1) T を深さ優先探索する：(v をカレントノードとする)
　(1-1) 参照辺 $(v, u) \in E$ があれば，$V_{OUT}(T_u)$ を呼び出す．
　ただし，T_u は u を根とする T の部分木を表す．
　(1-2) u が探索済みならば，その親 v へ戻り，
　　$(v, u) \in E$ が参照辺ならば，v の範囲ラベルが包含しない
　　すべての $x \in u_{out} + \{u\}$ を $v_{out} \leftarrow \{x\}$ とする．
　　$(v, u) \in E$ が実辺ならば，v の範囲ラベルが包含しない
　　すべての $x \in u_{out}$ を $v_{out} \leftarrow \{x\}$ とする．
(2) v_{out} を出力する．

図 10　改良版 v_{out} 構築アルゴリズム

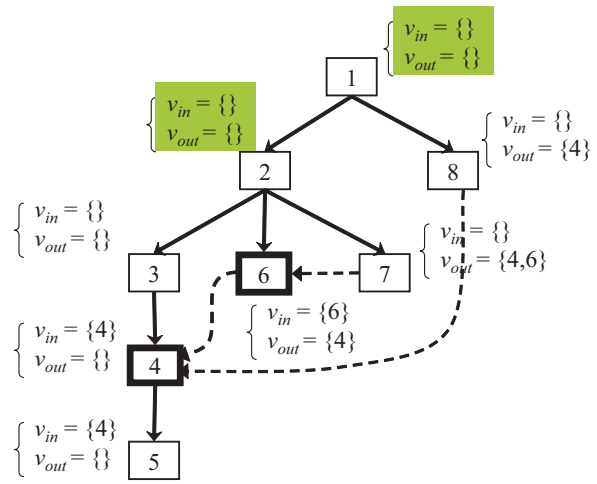

図 11　**提案手法の改良による 2Hop ラベルの例**: 図 9 と同じグラフに対して，改良版のラベル付けを行った例．網掛け部分のラベルが減少している．

改良した v_{out} 構築アルゴリズムを図 10 に示し，図 9 の例と同じグラフに対するラベル付けの実行例を図 11 に示した．この例で削減されたラベル数はあまり大きくないが，特に XML のような幅が広いデータに対して効果が期待できる．このことは実験によって示される．

本節では最後に提案手法と他手法の計算量を比較し，結果を表 1 に示す．ここで，n はノード数，m は辺数，p は [4] における各ノードが持つラベルの最大数，q は [29] におけるグラフの多次元分割数である．ℓ は，被参照ノード数であり，L は [24] による 1 ノードあたりの平均ラベル数 (最悪は $O(n)$) である．これらの変数の大小比較については，一般に，$n \leqq m$ であり，p は木の深さに依存し，$q < p$ であることが実験によって示されている [29]．これ以外の関係はデータの複雑さに依存する．本手法では，1 ノードあたりの最大ラベル数は ℓ であり，辺を巡回しながらラベルを先祖や子孫に継承するため，前処理時間は $O(n + \ell m)$ であり，領域量は $O(\ell n + m)$ である．次の節では，本手法の効果を実験によって確認する．

表 1　計算量の比較：[4], [24], [29] と本手法 (オーダー記号は省略)

	伝播法	多次元法	HOPI	提案手法
前処理時間	$n + pm$	$qn^2 + m$	$n^3 + m$	$n + \ell m$
領域量	$pn + m$	$q(n + m)$	$n\sqrt{m}$	$\ell n + m$
判定時間	p	q	L	ℓ

4. 実　　験

本手法の有効性を実験によって検証する．使用したデータは，XMark [25] によって生成した XML データ (120KB,1.2MB,12MB) と，XML Data Repository (注3) から取得した実データ Mondial(1.7MB) である．実験環境は，Celeron 3.2GHz，992MB メモリ上の WindowsXP であり，参照辺を含む XML の DOM 木を C++による双方向リストで実装した．

実験は，本手法における前処理時間および到達可能性を判定する時間を測定した．また，HOPI [24] を実装し，本手法との性能比較を行った．

表 2 に，使用した XML データの基本的な情報を示す．この表のノード数は，強連結成分分解後の非巡回有向グラフのノード数を表している．この結果から，XMark は，ノード全体のうちおよそ 3%が被参照ノード (すなわち親を 2 つ以上持つノード) であり，Mondial は，全体の 5%以上が被参照ノードである．平均入次数はその有向グラフの各ノードが持つ親の平均個数を表す．グラフが木の場合は，根以外がちょうど 1 つの親を持つので，平均入次数はほぼ 1 に等しい．表 2 の 1.19 という値は，平均して 5 ノード中 1 つが被参照ノードであることを表す．Mondial の平均入次数は XMark よりも大きいので，Mondial の方がより複雑なデータであると考えられる．[24] の手法では，この値が判定時間に大きな影響を与えている．

次に，提案手法で得られた 2Hop ラベルのサイズを，1 ノードあたりの平均ラベル数と最大ラベル数について HOPI のものと比較した結果を表 3 に示す．また，3.2 節で行った改良の効果を検証するため，改良前の値も示している．この結果から，平均・最大ラベル数ともに改良の効果が確認できる．また，HOPI との比較において，本手法の方が平均ラベル数が小さいことも示されている．ところが実際の判定は，HOPI は 2Hop ラベルのみで行うが，本手法では範囲ラベルの判定も必要である．よって実際の判定時間は，提案手法と HOPI ではそれほど差はないと考えられ，後で示す実際の測定結果もそれを裏付けている．さらに，表 3 より，データの増加に対して平均ラベル数は，本手法では増加傾向にあるが，HOPI ではほとんど変化していない．このことから，さらに大規模なデータに対する判定時間については，HOPI の方がやや有利であると考えられる．

表 2　XML データの概略

データの種類	サイズ (KB)	総ノード数	被参照ノード数	平均入次数
XMark	116	1665	45 (2.7%)	1.19
	1155	16529	471 (2.8%)	1.19
	11597	167865	4583 (2.7%)	1.19
Mondial	1740	20642	1064 (5.2%)	1.89

表 3　ノードあたりのラベルサイズの比較

データの種類	サイズ (KB)	平均ラベル数 提案手法 (改良前)/HOPI	最大ラベル数 提案手法 (改良前)/HOPI
XMark	116	1.2(1.2) / 5.3	23(26) / 345
	1155	1.4(1.9) / 5.3	246(283) / 2170
	11597	1.9(3.6) / 4.7	2515(2857) / 7950
Mondial	1740	2.3(3.3) / 4.8	200(202) / 1686

(注3)：http://www.cs.washington.edu/research/xmldatasets/

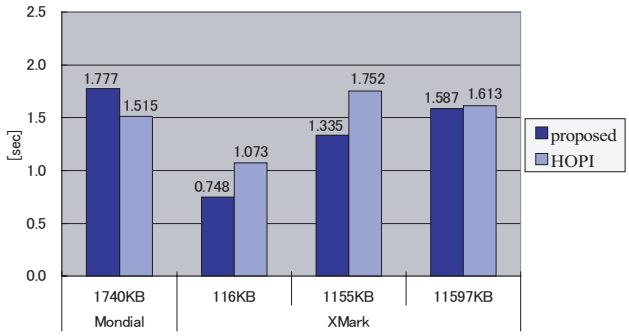

図 12　**判定時間の比較**: 100 万回の判定の実行時間

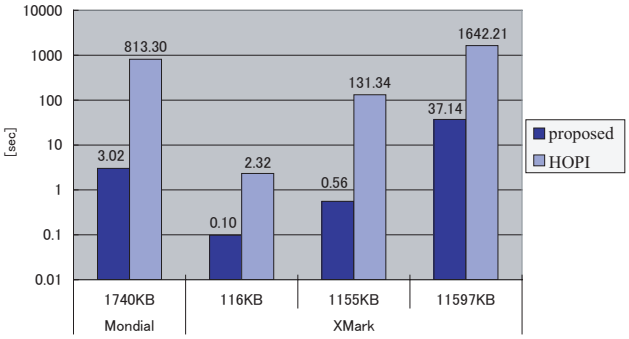

図 13　**前処理時間の比較**: 提案手法は範囲ラベル構築時間を含む

最大ラベル数は表 3 に示すように，有意な差がみられる．HOPI は漸近的には良い値を示すが，数十 MB 程度のデータでは，提案手法の方が良い結果が得られている．また，用いた実行環境では，HOPI がファイルを分割せずに実行可能なデータサイズは 1MB 程度であり，それ以上のデータは適当にファイルを分割して実験を行っている．本提案手法は，$O(n^2)$ のテーブルを構築する必要がないため，大規模なデータに対してもファイルを分割することなく直接ラベルを構築できる．このことは，次に示す前処理時間の比較に現れている．

最後に，提案手法と HOPI における前処理時間と到達可能性判定時間の比較結果を示す．図 12 は，ランダムに生成した 100 万組のノードに対する到達可能性判定時間を測定した結果である．これより，XMark では提案手法の方が高速であり，Mondial では，HOPI の方が高速であるが，判定した回数を考慮に入れるとこれらの判定時間に顕著な違いはない．しかしながら，データの増加に対して，判定時間の差は縮まっており，前述の平均ラベル数における考察が裏付けられている．

図 13 は提案手法と HOPI の前処理時間の比較結果である．この結果より，本研究で提案する範囲ラベルと 2Hop ラベルを組み合わせたラベル付け手法は，従来の手法に比べて十分高速に構築することができ，前節の計算量の比較と併せて，必要な主記憶量も抑えられていることがわかる．前処理時間は，XMark では最大 230 倍 (1.2MB のデータ)，より複雑な Mondial では 260 倍以上効率的である．また，その結果得られた索引サイズと判定時間も従来手法と比較して遜色ないものといえる．以上により，本手法の有効性が確かめられた．

5. まとめ

本研究では，木構造よりも複雑な XML データに対して，高速にノード間の到達可能性を判定するための効率的なアルゴリズムを提案し，その効果を実験によって検証した．実験では，最適なラベルに対するよい近似が保証されている 2Hop ラベル構築アルゴリズム [24] と比較して，本提案手法の有効性を示した．本手法は，データサイズに対する規模耐性が高く，特に，他手法との比較において前処理時間の削減率が著しく向上している．しかしながら，比較対象の HOPI[24] では，ラベルの更新法も提案しており，ラベルを効率的に再構築する手法は本研究の今後の課題である．一方で，RDF のようなさらに複雑な構造に対する到達可能性問題へ本手法を応用することも重要な課題である．

謝辞 本研究の一部は，平成 18 年度科学研究費補助金 (基盤研究 (A) 課題番号 17200011 課題名” 大規模半構造データからの高速知識発見システムの開発”) の支援を受けて行われた．また，論文の執筆にあたり査読者をはじめとする様々な方から有益なコメントを頂いた．ここに記して謝意を表す．

文　献

[1] S. Abiteboul, P. Buneman, D. Suciu, Data on the Web, Morgan Kaufmann, 2000.

[2] S. Abiteboul, H. Kaplan, T. Milo, Compact labeling schemes for ancestor queries, In SODA 2001, pp. 547-556, 2001.

[3] A. Aboulnaga, A. R. Alameldeen, J.F. Naughton, Estimating the Selectivity of XML Path Expressions for Internet Scale Applications, In VLDB 2001, pp. 591-600, 2001.

[4] R. Agrawal, A. Borgida, H.V. Jagadish, Efficient Management of Transitive Relationships in Large Data and Knowledge Bases, In SIGMOD 1989, pp. 253-262, 1989.

[5] A. V. Aho, J. E. Hopcroft, J. D. Ullman, Data Structures and Algorithms, Addison-Wesley, 1987.

[6] A. Arion, A. Bonifati, G. Costa, S. D'Aguanno, I. Manolescu, A. Pugliese, Xquec: pushing queries to compressed XML data, In VLDB 2003, pp. 1065-1068, 2003.

[7] S. Boag, D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, J. Simeon, XQuery 1.0: An XML Query Language, http://www.w3.org/TR/2002/WD-xquery-20020816, 2002.

[8] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, Extensible Markup Language (XML) 1.0 W3C Recommendation, http://www.w3.org/TR/REC-xml, 1998.

[9] L. Chen, A. Gupta, M. E. Kurul, Efficient Algorithms for Pattern Matching on Directed Acyclic Graphs, In ICDE 2005, pp. 384-385, 2005.

[10] C.-W. Chung, J.-K. Min, K. Shim, APEX: Adaptive Path Index for XML Data, In SIGMOD 2002, pp. 121-132, 2002.

[11] J. Clark, S. DeRose, XML Path Language (XPath) Version 1.0 http://www.w3.org/TR/xpath, 1999.

[12] E. Cohen, E. Halperin, H. Kaplan, U. Zwick Reachability and Distance Queries via 2-Hop Labels, In SODA 2002, pp. 937-946, 2002.

[13] E. Cohen, H. Kaplan, T. Milo, Labeling Dynamic XML Trees, In PODS 2002, pp. 271-281, 2002.

[14] 江田毅晴, 天笠俊之, 吉川正俊, 植村俊亮, XML 木のための更新に強い節点ラベル付け手法, DBSJ Letters, Vol. 1, No. 1, pp. 35-38, 2002.

[15] M. F. Fernandez, D. Suciu, Optimizing Regular Path Expressions Using Graph Schemas, In ICDE 1998, pp. 13-23, 1998.

[16] R. Goldman, J. Widom, DataGuides: Enable Query Formulation and Optimization in Semistructured DataBases, In VLDB 1997, pp. 436-445, 1997.

[17] T. J. Green, A. Gupta, G. Miklau, M. Onizuka, D. Suciu, Processing XML Streams with Deterministic Automata and Stream Indexes, ACM TODS, vol. 29, no. 4, 2004.

[18] W.-Y. Lam, W.-N. Peter, M. Levene, XCQ: XML Compression and Querying System, In WWW 2003 (posters), 2003.

[19] T. Maita, H. Sakamoto, A Simple Extension of Queriable Compression for XML Data, In AMT 2005, pp. 91-95, 2005.

[20] T. Milo, D. Suciu, Index structures for path expressions, In ICDT 1999, pp. 277-295, 1999.

[21] J.-K. Min, M.-J. Park, C.-W. Chung, XPRESS: A Queriable Compression for XML Data, In SIGMOD 2003, pp. 122-133, 2003.

[22] 中村有作, 舞田哲哉, 坂本 比呂志, 参照構造を持つ XML 上の高速な到達可能性判定, 人工知能学会論文誌, Vol. 22, No.2, pp. 191-199, 2007.

[23] C.-W. Park, J.-K. Min, C.-W. Chung, Structural Function Inlining Technique for Structually Recursive XML Queries, In VLDB 2002, pp. 83-94, 2002.

[24] R. Schenkel, A. Theobald,G. Weikum, Creation and Incremental Maintenance of the HOPI Index for Complex XML Document Collections, In ICDE 2005, pp. 360-371, 2005

[25] A. Schmidt, F. Waas, M. Kersten, M. Carey, I. Manolescu, R. Busse, Xmark: A benchmark for XML data management, In VLDB 2002, pp. 974-985, 2002.

[26] T. Schwentick, XPath Query Containment, SIGMOD Record, Vol. 33, No. 1, pp. 101-109, 2004.

[27] D. Srivastava, S. Al-Khalifa, H. V. Jagadish, N. Koudas, J. M. Patel, Y. Wu, Structural joins: A primitive for effcient XML query pattern matching, In ICDE 2002, pp. 141-152, 2002.

[28] P. M. Tolani, J. R. Haritsa, XGRIND: A Query-friendly XML Compressor, In ICDE 2002, pp. 225-234, 2002.

[29] 鳥井修, 白井智, 金井達徳, 非巡回グラフのための拡張レンジラベリング手法, DBSJ Letters, Vol.5, No.1, pp.157-160, 2006.

[30] Y. Wu, J. M. Patel, H. V. Jagadish, Structural Join Order Selection for XML Query Optimization, In ICDE 2003, pp. 443-454, 2003.

[31] C. Zhang, J. Naughton, D. DeWitt, Q. Luo, G. Lohman, On Supporting Containment Queries in Relational Database Management Systems, In SIGMOD 2001, pp. 425-436, 2001.