# 有向グラフ上の最短経路問題に対する効率的な索引付け

## Scalable Indexing for Reachability Problem on Directed Graphs

原口 新平* 中村 有作† 坂本 比呂志‡

Shinpei HARAGUCHI Yusaku NAKAMURA Hiroshi SAKAMOTO

**本研究では，有向グラフ上の任意の２ノード間の距離を高速に計算できる規模耐性の高い索引構造を提案する．本研究で提案する手法は，有向グラフ上の最短距離計算として，ノード間の距離計算の結果をあらかじめ隣接行列に格納しておく方法がある．しかしながら，この方法では時間計算量と領域計算量の両方のコストが高く，大規模なグラフ構造に適用する事は困難である．そこで本研究では，一般の有向グラフに適用可能な距離計算のためのラベル付けを提案し，その有効性を実験により検証する．特に，XML データにおいて，前処理に必要な主記憶量，計算時間および索引サイズを示す．**

**We propose an efficient algorithm which reports the length of a shortest path between any two nodes of a directed graph in constant time. In usual method, we can obtain the constant time response for any query in this problem by using an adjacent matrix for the graph. However, this method requires huge memory space and it is difficult to apply it to lage database. So we introduce more practical method for this problem and evaluated the efficency by experiments.**

## 1. はじめに

本研究では，有向グラフのラベル付けを応用した任意の 2 ノード間の最短距離を高速に計算する索引付けを提案する．XML データは，リンク構造をグラフの有向辺と見なすことで順序木や有向グラフと同等である．そこで，半構造データ上の問い合わせ言語において，任意のノード間の先祖子孫関係を高速に判定する技術が提案されている．本研究では，そのようなラベル付けを拡張し，比較的大規模なデータに対して，ノード間の距離すなわち最短の経路長を実用的な時間で計算できる手法を提案する．本研究では有向グラフ全体を対象としている．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．

### 1.1 最短経路問題

グラフにおける最短経路問題は，グラフ $G$ が与えられたとき，ノード $u, v$ の条件によって次のように分類される．

1. 単一始点最短経路：固定された $v$ からの最短経路
2. 単一点対間最短経路：固定された $(u, v)$ 間の最短経路
3. 全点対間最短経路：すべての $(u, v)$ 間の最短経路

*九州工業大学 大学院情報工学府
s_haraguchi@donald.ai.kyutech.ac.jp
†日立製作所 ソフトウエア事業部
yuusaku.nakamura@gmail.com
‡正会員 九州工業大学 大学院情報工学研究院
hiroshi@ai.kyutech.ac.jp

ここではじめの二つの問題に本質的な違いはない．本研究で取り扱うのは最後の問題であり，1 質問あたり定数時間で答えを得る手法として，あらかじめすべての組み合わせに対して値を計算して表の形式で保存しておくダイクストラ法を応用したものが提案されている [5].

ところがこれらの手法は，ダイクストラ法を素朴に実行した場合は $O(n^2 m)$ 時間かかり，巧妙なデータ構造で改善した場合でも $O(n^2 \log n + nm)$ 時間かかる．また，表の形式で値を保存するため，領域は常に $\Omega(n^2)$ である．ここで，$n$ はグラフのノード数であり $m$ は辺数を表す．本研究の手法では，要求された距離を定数時間で得ることをあきらめて，より大規模なグラフに対しても実行可能な距離計算のためのラベル付けを行う．

### 1.2 グラフのラベル付け

有向グラフ $G$ が木構造であるとき，各ノードに対して前置順と後置順による順位付けを考える．あるノード $v$ の前置順および後置順による順位をそれぞれ $pre(v), post(v)$ とする．このとき，整数の組 $(pre(v), post(v))$ は，以下の性質を満たす．

木の任意のノード $u, v$ に対して，$pre(u) < pre(v)$ かつ $post(u) > post(v)$ であるときまたそのときに限り $u$ は $v$ の先祖である．また $(pre(u), post(u))$ は $(pre(v), post(v))$ を包含するという．

このような性質を満たすラベルを範囲ラベルと呼ぶ．この範囲ラベルを使う事で実辺上の先祖子孫関係かどうかを判定できるが，参照辺を介した先祖子孫関係については判定できない．

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 [13]．そこで，この範囲ラベルに加え，次のように 2Hop ラベルをつける．2Hop ラベルとは一般の有向グラフに対する到達可能性を判定する手法である．

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 $v_{in}, v_{out} \subseteq V$ を決めたとき，$L(G) = \{(u_{in}, u_{out}) | u \in V\}$ 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．各ノードがなるべく小さい数のラベルを持つような，到達可能性を判定するための索引として都合がよい．しかし，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．

その他，効率的なグラフのラベル付け手法として，R2Hop[14] が知られており，本研究の DR2Hop はこの手法に基づいている．R2Hop は，有向グラフ $G = (V, E)$ を強連結成分分解した非巡回グラフ $G'$ に対し，深さ優先探索で全域木 $T$ を計算することで $T$ に範囲ラベルを設定する．$T$ に対しては通常の範囲ラベルが計算可能であるが，それ以外のノードの到達可能性に対してのみ 2Hop ラベルを計算することで，全体のラベルサイズを抑えている．

### 1.3 提案手法

本研究では，有向グラフ上の全点対間最短経路問題にグラフのラベル付けのための手法を応用し，距離計算を高速かつ省スペースで実現する索引付けを提案する．特に，疎なグラフに対してはほぼグラフのサイズ (辺数) に比例する時間で索引付けを実行できる．本提案手法では，$O(n^2)$ サイズの領域が必要ないため比較的大規模なグラフに対しても妥当な時間と領域で索引を構築可能である．

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 R2Hop を応用した手法である．まず，与えられた有向グラフ $G$ に対して，深さ優先探索などの標準的な手法で $G$ 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．このとき，範囲ラベルとして木の深さも持たせることで，木の先祖子孫間の距離が計算できる．そして，範囲ラベルで判定できないが互

**アルゴリズム** $DRV_{OUT}(G)$
**入力**：連結な有向グラフ $G=(V,E)$ とその全域木 $T$
**出力**：任意の $v \in V$ に対するラベル $v_{out}$
(1) $T$ を $v$ から深さ優先探索する
　(1-1) $v$ が被参照ノードならば先祖へ幅優先探索:
　　$u$ をカレントノード,
　　$i$ を $v$ から $u$ までの幅優先探索による距離とし,
　　$u$ が以下の (a),(b),(c) すべてを満たすならば
　　$u_{out} \leftarrow u_{out} + \{(v:i)\}$ とする
　　(a) $v \notin u_{out}$
　　(b) $v$ の先祖かつ，$u$ から $v$ の実辺距離が $k > i$
　　(c) $u \neq v$
　(1-2) $v$ からの深さ優先探索に戻る
(2) $v_{out}$ を出力

**図 1.** $v_{out}$ **構築アルゴリズム**
**Fig. 1.** Algorithm for $v_{out}$

いに到達可能な関係にあるノードに対して 2Hop ラベルを計算し，そのラベルにも距離を持たせる．ただし，到達可能性の判定では，ある二つのノードが共通のラベルを持つだけでそれが判定できたのに対して，最短経路問題では，ノードラベルとして常に最短の距離を与えるものを保証しなければならない．そこで，ダイクストラ法を部分的に適用することでこの問題を解決する．本研究の索引付けは，大規模な表を必要としないため，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．

## 2. 提案手法による効率的な距離計算

### 2.1 DR2Hop のラベル構築アルゴリズム

本研究で提案する最短距離計算は，グラフの全域木 $T$ の各ノード $v$ に深さ情報を持たせる事と $v_{out}$ 内の各ラベル $u$ に $v$ から $u$ までの距離 $i$ を付加させる事により実現される．この最短距離計算に必要な $v_{out}$ のラベル構築アルゴリズムと最短距離の計算アルゴリズムを示す．なお，距離計算に必要なもう一方のラベル $v_{in}$ の構築は，[14] で提案された手法をそのまま用いる．すなわち，あるノード $v$ のラベル $v_{in}$ とは，全域木における $v$ の先祖である被参照ノード全体の集合である．

図 1 に $v_{out}$ 構築アルゴリズム，図 2 にその動作例を示す．本手法で構築する $v_{out}$ のラベルの集合は，$v$ の子孫の全ての被参照ノード (実辺パスのみで辿れる，かつ，実辺パスの方が距離が短いノードは除く) の集合である事を意味している．本手法では，グラフ内の各被参照ノードを始点に幅優先探索によりラベルと距離を付けしているので，ラベル内の距離情報は最短距離であることを保証する．$n$ はノード数，$\ell$ は被参照ノードの数，$m$ は辺の数とした時，本手法では，1 ノードあたりの $v_{out}$ の最大ラベル数は $\ell$ であり，ラベル付けは各被参照ノードごとにグラフの辺を巡回しながら追加していくので，$v_{out}$ 構築にかかる時間計算量と領域計算量はそれぞれ $O(\ell m)$ と $O(\ell n)$ である．

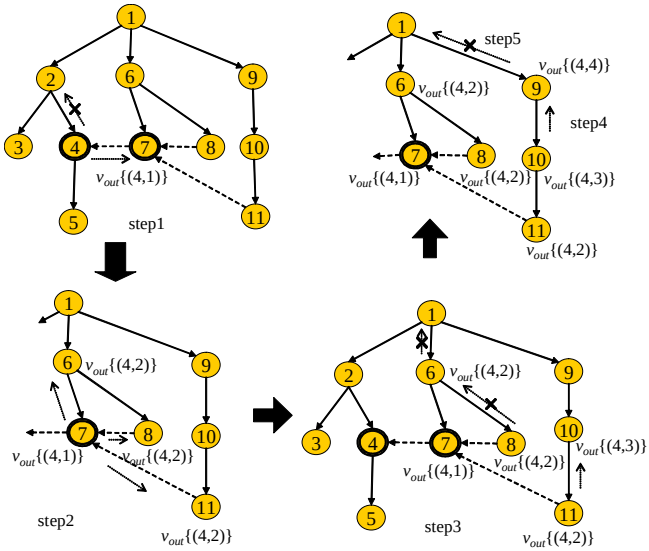


**図 2.** $DRV_{OUT}(G)$ **の動作例**
**Fig. 2.** An example run of $DRV_{OUT}$

### 2.2 最短距離計算方法

次に，距離計算についての説明とその計算アルゴリズムを提案する．任意のノード $v$，$u$ が実辺上で先祖子孫関係の場合，この実辺パスの距離を計算は，それぞれの木における深さを用いる．そして，この 2 つの深さの差をそのまま距離とみなせる．任意のノード $v$，$u$ が参照辺を通り到達可能な場合は，距離情報付きの 2Hop ラベルで距離計算を行う．$(w:i) \in v_{in} \cap u_{out}$ の場合，これは $u$ から距離 $i$ 先にある $w$ を経由して $v$ に到達可能であることを示している．

1. $u$ から $w$ までの最短距離が $i$ であることは，$v_{out}$ アルゴリズムで保証されている．
2. $v_{in}$ は，$v$ の先祖でかつ被参照ノードの集合であるので，$w$ と $v$ は全域木における先祖子孫関係があり，その距離は深さの差で計算できる．

よって，$u$ から $v$ までの距離は，上記いずれかの小さい方の値として計算可能である．深さの差と距離情報付き 2Hop ラベルでの距離計算の例を図 3 で示す．

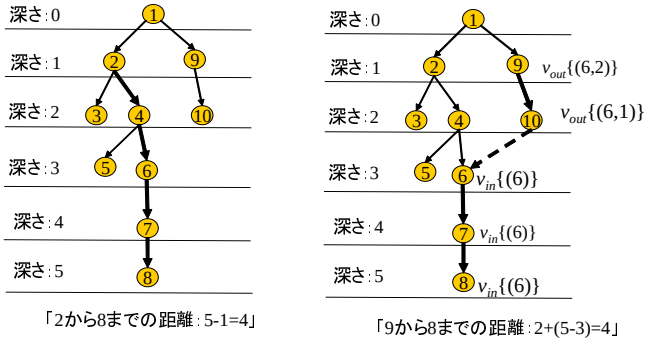


**図 3. 2 ノード間の距離計算**
**Fig. 3.** A computation of the distance of two nodes

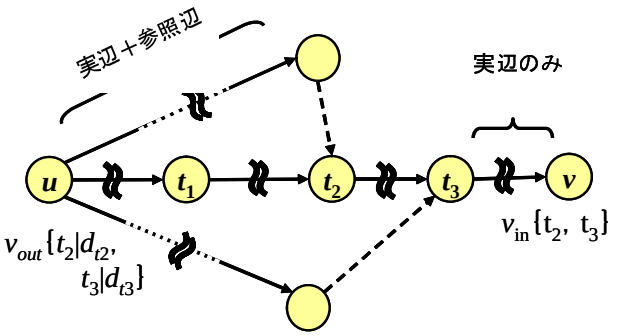


**図 4. 最短経路の保証：実辺だけを辿る経路と参照辺を辿る経路のそれぞれの最小値を求めることが可能であるので，全体として最短の経路を計算できる**

**Fig. 4.** The guarantee of the shortest path

### 2.3 最短距離の保証

ノード $v$ の持つ $v_{in}$ 内のラベル $u$ は，$v$ と $u$ が実辺のみで接続されていることを意味するので，$v-u$ 間の実辺経路の距離は全域木の深さで計算できる．そして，$v_{out}$ 内のラベル $u'$ と対になっている距離の値は，アルゴリズム $DRV_{OUT}(G)$ の幅優先探索により，$v-u'$ 間の最短距離を保証している．ノード $v$ の持つ $v_{out}$ は，$v$ から参照辺を介し到達可能な被参照ノードのラベルをすべてを持っている (ただし，実辺のみでも到達可能でかつ，実辺のみの経路の距離が短い場合は除く)．これらの事実から，本手法による計算が最短距離を計算することを示す．

ノード $v, u$ の最短距離を以上の手法で計算したとき，この距離が最短でないと仮定する．最短でない場合とは，求めた経路 $p$ よりも短い経路 $q$ でたどり着けることである．この時，$q$ が実辺のみの経路でない事はあきらかであるので，$q$ は参照辺を少なくともひとつは通る経路である．参照辺を通る経路は必ず被参照ノード $x$ を経由するので，$v$ の $v_{out}$ には $x$ が含まれる．しかし，参照辺を通る経路の距離は幅優先探索による計算ですでに得られており，その最小値と全域木における経路の距離の比較がすでに行われているので，$x$ を経由した経路 $q$ が最短距離になるという事はありえない．したがって，求めた $p$ の距離が 2 点間の最短距離である．図 4 にこの証明の概略を示す．

## 3. 実験

本研究で提案する手法の有効性を実験によって確認する．使用したデータは，XMark [16] によって生成した 3 つの XML データ (503KB,1026KB,3044KB) 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最短距離を判定する時間を測定した．また，HOPI [15]，強連結成分分解によって巡回を無くしたグラフに対する R2Hop ラベル付け手法 [14]（以下 (scc)R2Hop)，巡回を含んだまま R2Hop ラベル付けする手法の 3 手法を実装し，前処理時間や平均ラベル数などの比較を行った．ただし，この 3 手法は到達可能性を判定できる距離計算はできない．

表 1 に，使用した XML データの基本的な情報と，提案手法で得られた 2Hop ラベルのサイズを他の 3 手法のものと比較した結果を示す．ノード数とは XMark を巡回グラフで表したときのノード数を表す．XMark では，全体のノードのおよそ 3.5%が被参照ノード (すなわち 2 つ以上の親を持つノード) である．

平均ラベル数は 1 ノードあたりの 2Hop ラベルの個数を表す．提案手法は HOPI と (scc)R2Hop にはラベル数で劣っている．しかし XML データをそのまま表現したグラフに対しての R2Hop と比べると，ラベル数は変わらないにも関わらず最短距離計算の機能を追加できている．この要因のひとつとして，XMark においては，あるノード $x$ からあるノード $y$ へと参照辺を通過して到達する経路があったとき，$x$ が $y$ の実祖先である可能性が極めて低いことが挙げられる．オーダ表記では差が見えなくとも，平均ラベル数・最大ラベル数は判定時間と強い相関関係があるので，実際のラベル数の差が判定時間の差となると考えられ，後で示す測定結果からもそれが正しいことが分かる．巡回を含むグラフに対する R2Hop と提案手法のラベル数の増加が大きいのは，データサイズの増加に伴って大きくなる巡回経路上のノードがすべて等しく $v_{out}$ のラベルを持つためである．

最後に，提案手法と他 3 手法における前処理時間と，判定時間の比較結果を示す．図 5 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 (提案手法については最短経路判定) 時間を測定した結果である．ラベル数に違いがないが，実行時間で提案手法の方が R2Hop よりもわずかに時間が掛かっているのは，R2Hop が 2 ノード間に到達可能な経路を発見した時点で処理を終えて良いのに対し、提案手法は最も距離数の少ない経路を探すために最後のラベルまで追っているためと考えられる．

**表 1. データの概略と各手法のノードサイズ比較**

**Table. 1.** Outline of data and comparison of node size

| サイズ (KB) | ノード数 | 被参照ノード数 | 手法 | 平均ラベル数 | 最大ラベル数 |
|---|---|---|---|---|---|
| 503 | 7526 | 267 (3.6%) | [15] | 4.87 | 469 |
| | | | [14] | 1.68 | 116 |
| | | | R2Hop | 20.81 | – |
| | | | 提案手法 | 20.81 | 263 |
| 1026 | 14974 | 525 (3.5%) | [15] | 4.78 | 723 |
| | | | [14] | 1.65 | 217 |
| | | | R2Hop | 44.05 | 521 |
| | | | 提案手法 | 44.05 | 521 |
| 3044 | 44170 | 1546 (3.5%) | [15] | 4.57 | 1811 |
| | | | [14] | 1.67 | 699 |
| | | | R2Hop | 148.68 | – |
| | | | 提案手法 | 148.68 | 1543 |

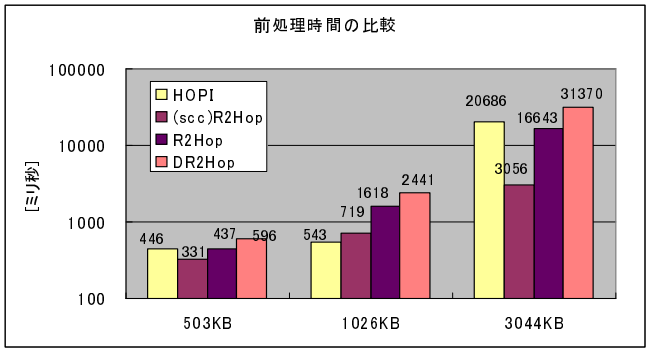


**図 5. 100 万回の判定の実行時間**

**Fig. 5.** Computation time for 1M queries

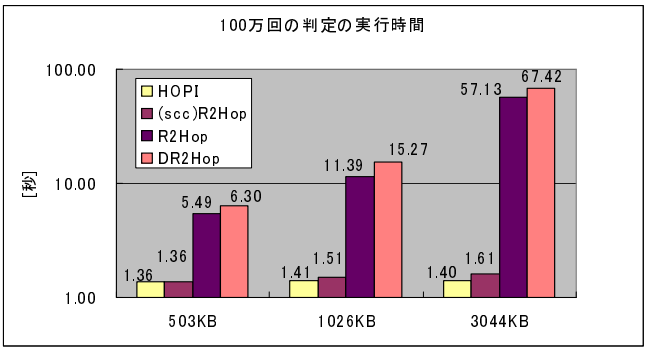


**図 6. 前処理時間の比較**

**Fig. 6.** Comparison of preprocessing time

図 6 は提案手法と他 3 手法の前処理時間の比較結果である．提案手法の前処理時間は，503KB および 1026KB のとき R2Hop の 2 倍以内，3MB のとき HOPI および R2Hop の時間の 2 倍以内に収まっている．この結果より，本研究で提案する R2Hop に距離情報を持たせたラベル付け手法は，2 点間の最短距離を計算できない従来の手法に比べて妥当な速度で構築することが結論できる．

## 4. おわりに

本研究では，比較的大規模で疎なグラフ構造を持つ XML データに対して，前処理によって 2 点間の最短距離を計算するアルゴリズムを提案した．実験では，HOPI と本手法の基礎となる (scc)R2Hop と巡回を含むグラフへの R2Hop とを比較した．本手法では，強連結成分分解をしない場合，ラベル数や判定時間が増加したが，妥当な計算時間で最短距離に対するラベル付けが可能であることが確認された．今後は，本手法を応用したパターン発見等の枠組みを提案することが課題である．

## [謝辞]

本研究の一部は，平成 19 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．

## [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] 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.

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

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, Introduction to Algorithms, MIT Press, 1990.

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

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

[8] 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.

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

[10] 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.

[11] D. B. Johnson, Efficient algorithms for shortest paths in sparse networks, Journal of the ACM, vol. 24, no. 1, pp.1-13, 1977.

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

[13] 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.

[14] 中村有作, 舞田哲哉, 坂本 比呂志, 高速な到達可能性判定のための規模耐性の高い索引付け, DBSJ Letters, Vol. 6, No. 1, pp. 77-80, 2007.

[15] 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

[16] 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.

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

[18] 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.

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

**原口 新平　Shinpei HARAGUCHI**
2007 年九州工業大学情報工学部知能情報工学科卒業．現在，同大学院情報工学府知能情報工学研究系に在学．高速な XML データベース，大規模な半構造データの索引付けに関する研究に従事．

**中村 有作　Yusaku NAKAMURA**
2006 年九州工業大学情報工学部知能情報工学科卒業．2008 年同大学院情報工学府知能情報工学研究系修了．現在，(株) 日立製作所に勤務．

**坂本 比呂志　Hiroshi SAKAMOTO**
九州工業大学情報工学部准教授．1998 年九州大学大学院システム情報科学研究科情報理学専攻博士後期課程修了 (日本学術振興会特別研究員)．博士 (理学)．1999 年から 2003 年 7 月まで九州大学大学院システム情報科学研究科助手．機械学習，半構造データからの知識獲得および半構造データに対する索引付けに関する研究に従事．2006 年度人工知能学会論文賞などを受賞．電子情報通信学会，人工知能学会，情報処理学会，日本データベース学会各会員．