非巡回グラフのための拡張レンジラベリング手法

Extended Range Labeling Technique for Acyclic Graphs

鳥井 修♣ 白井 智♣
金井 達徳♦

Osamu TORII Satoshi SHIRAI
Tatsunori KANAI

レンジラベリングは，XML 文書などツリーで表現されるデータの各ノードに，レンジと呼ぶツリーの位相情報を表現する数字の組を付与する手続きのことであり，レンジを用いることで XML 文書などへの問い合わせ処理の効率的な実現が可能となる．しかしながら，従来のレンジラベリング手法は，ツリーしか考慮していない．本論文では，2 種類の拡張により，複雑なデータ構造への問い合わせ処理を効率的に行うことへのニーズの高まりに答える．

Range labeling is the procedure that gives a pair of figures called range to each node of a tree such as XML document. The range represents the topological information of the tree, and efficient query processing of XML documents can be achieved based on range labeling. But current range labeling takes only trees into consideration. In this paper, we will extend the range labeling technique for acyclic graphs.

1. はじめに

XML [2] 文書は，近年，電子化された情報の標準的なフォーマットになりつつあり，様々な情報が XML 文書形式で記述，交換，保存されている．このため，XML 文書の問い合わせ処理を効率的に行うことは，膨大な情報が溢れている今日において，目的に合った情報を獲得する際に必須である．

レンジラベリング [5] は，ツリーで表現されるデータにおいて，レンジと呼ぶ始点と終点からなる数字の組を，各ノードに 1 組ずつ付与する手続きのことである．レンジラベリングによって各ノードに付与されるレンジはツリーの位相情報を表現しており，任意の 2 個のノードの先祖子孫関係は，これら 2 ノードに付与されたレンジのみを比較することで判定可能である．このため，レンジラベリングの結果を利用すると，多くの種類のツリーノード検索を高速に実行することが可能である．

XML 文書はツリーとみなせるので，レンジラベリングを行えば，要素間の先祖子孫関係を高速に判定することが可能である．このため，レンジラベリングは XML 文書の問い合わせ処理の効率的な実現に必要な基盤技術の 1 つである．しかしながら，従来のレンジラベリング手法はツリーしか考慮していないため，以下の理由で十分とは言い難い．第 1 に，問い合わせによっては XML 文書をツリーとみなした場合には問い合わせ処理を行う際に扱いづらく，グラフとみなす必要があり，第 2 に，XML 文書が標準的なフォーマットになった今日であっても，XML 文書やツリーでは表現しにくく，グラフによってのみ適切に表現可能な情報が数多く残っている．これらのことから，グラフへの問い合わせ処理を効率的に行うための技術へのニーズは，これから益々高まってくる．

本論文では，グラフ，特に非巡回グラフへの問い合わせ処理を効率的に行うことを可能にするために，従来のレンジラベリング手法を「重なりレンジラベリング」と「グラフの多次元分割」を用いて拡張する．

2. XML とレンジラベリング

XML (eXtensible Markup Language) はマークアップ言語の 1 つであり，タグを用いて階層構造を持つ情報を表現する．また，XML 文書はエレメントやテキストをノードとするツリーとして扱うことが可能であり，このツリーを DOM (Document Object Model) [3] ツリーと呼ぶ．

グラフの各ノードに，以下の条件を満たすレンジと呼ぶデータを付与する手続きを考える．

レンジ数条件: レンジは始点，終点の 2 個の数字の組であり，各ノードに付与されるレンジ数は 1 個である．

包含条件: 任意の 2 ノード $u(u0, u1)$, $v(v0, v1)$ に対して，グラフ上で u が v の先祖になっている時，かつこの時に限り，レンジ $(u0, u1)$ はレンジ $(v0, v1)$ を包含する．ただしここで，2 個のレンジ (a, b), (c, d) が $a < c, d < b$ の関係を満たす時，レンジ (a, b) はレンジ (c, d) を包含するという．また，グラフ上でノード e からノード f へのパスが存在する時，e を f の先祖 (f を e の子孫) と呼ぶ．

重なり条件: 任意の 2 個のノード u $(u0, u1)$, v $(v0, v1)$ に対して，レンジ $(u0, u1)$ とレンジ $(v0, v1)$ が部分的に重なりを持つことはない．ただしここで，互いに包含関係にないノード a のレンジ $(a0, a1)$ とノード b のレンジ $(b0, b1)$ の両方に包含されるレンジ $(c0, c1)$ を持つノード c が存在する場合，レンジ $(a0, a1)$ とレンジ $(b0, b1)$ とは部分的に重なりを持つという．

レンジを付与する対象のグラフがツリーであれば，上記条件をすべて満たすレンジが必ず存在するので，この時上記手続きをレンジラベリングと呼ぶ．

レンジラベリングを行う最大の目的は，2 ノードの先祖子孫関係を高速に判定し，これを用いて様々な問い合わせを効率よく処理することである．任意の 2 ノードの先祖子孫関係は，上記「包含条件」から，これら 2 ノードに付与されたレンジのみを比較することで判定可能であり，判定時間は O(1) である．これらのことから，レンジラベリングの結果を利用すると，例えば，XPath [4] で表現される問い合わせ「A // B」など，多くの種類のツリーノード検索を高速に実行することが可能である．

レンジラベリングを行うアルゴリズムには，ツリーのルートから深さ優先探索を行い，各ノードの探索を開始する順番 (プレオーダー) を始点に，各ノードの探索を終了する順番 (ポストオーダー) を終点に与える方法がある．

ツリーのノード数を n, エッジ数を m とすると，レンジラベリングの計算時間は，深さ優先探索に必要な計算時間，つまり O (m), レンジに必要な記憶容量は O(n) である．

3. 非巡回グラフとレンジラベリング

本論文ではレンジラベリングを非巡回グラフ向けに拡張し，非巡回グラフのノード間の先祖子孫関係を高速に判定にする目的に適したラベル体系を構築する．非巡回グラフはループを持たないグラフのことであり，以下において，特に混乱のない場合には，単にグラフと記述した時は非巡回グラフのことを指すものとする．

♣ 正会員 (株)東芝 研究開発センター
♦ (株)東芝 研究開発センター

従来のレンジラベリングをグラフ向けに拡張する手法の 1 つに Agrawal の方法がある．一般に 1 次元レンジによって任意のグラフの位相情報を表現することは不可能である．そこで Agrawal の方法では，グラフの各ノードが多次元レンジを保持することを許容する．Agrawal のアルゴリズムは，第 1 に，グラフから部分位相グラフ (「G' の任意の 2 ノード u, v に対して，u が v の先祖である場合には u は G においても v の先祖である」という関係が成り立つ時，G' は G の先祖子孫関係のうち一部を表現していることから，G' を G の部分位相グラフと呼ぶ) であるツリーを抽出して，このツリーの各ノードに従来の方法でレンジラベリングを行い，第 2 に，グラフの子から親にレンジを再帰的に伝播し，子のレンジのうち親のレンジに包含されていないもの「のみ」を親のレンジリストにマージする．

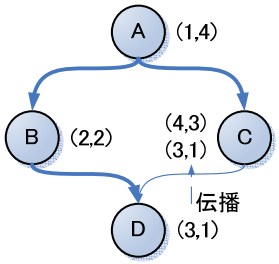

図 1 Agrawal の方法
Fig. 1 Agrawal's Algorithm

図 1 に，4 個のノードからなるグラフに Agrawal の方法によってレンジ付けを行った結果の 1 例を示す．ただし [6] では，レンジの始点として子孫ノードの最小ポストオーダーを用いているが，ここではプレオーダーを用いている．両者に本質的な違いはない．

Agrawal の方法によって付与されたレンジは，グラフの位相情報を表現しており，従来のレンジラベリングと同様，任意の 2 個のノードの先祖子孫関係を，これら 2 ノードに付与されたレンジのみを比較することで判定可能である．具体的には，2 ノード u, v のレンジリストを取得し，ノード v のレンジリスト中のすべてのレンジが，ノード u のレンジリスト中のいずれかのレンジに包含される場合には，ノード u はノード v の先祖であると判定，それ以外の場合には，ノード u はノード v の先祖ではないと判定する．

Agrawal の方法によってグラフのリーフノードに付与されるレンジの個数は必ず 1 個である．しかしながら，前述の通り Agrawal の方法はレンジの伝播を含んでいるため，インターナルノードには複数，場合によっては膨大な数のレンジが付与されるという欠点を持つ．ノードに付与されるレンジの個数が多いと，先祖子孫関係を判定する際に調べなければいけないレンジの個数が多くなり，結果として検索効率の低下につながる．

4. 重なりレンジラベリングと多次元分割

4.1 重なりレンジラベリング

本節では，Agrawal の方法の欠点を踏まえ，これと異なるレンジラベリング拡張を行う．

第 1 の拡張として，「重なりレンジラベリング」手続きを導入する．

グラフの各ノードに，第 2 章で述べたレンジ数条件，包含条件のみを満たすレンジを付与する手続きを考え，この手続きを「重なりレンジラベリング」と，この手続きによって位相情報が表現可能であるデータ構造を「重なりレンジラベリング可能グラフ」とそれぞれ呼ぶ．従来のレンジラベリング手法から重なり条件を除く，つまりレンジが部分的に重なりを保持してよいとする拡張を行うことで，ツリーよりも一般的なデータ構造の位相情報をレンジによって表現することが可能となる．

アルゴリズムの詳細は以下の通りである．

第 1 に，グラフからツリーを抽出して従来手法で初期レンジを付与する．Agrawal の方法では，ルートから各ノードへの経路長が最長になるツリーを抽出しており，本アルゴリズムでも同じ方法を用いる．

第 2 に，ノードを始点の数字でソートしたリスト Ls と終点の数字でソートしたリスト Le の 2 種類を準備し，下記 SWAPS, SWAPE を，入れ替え可能な隣接ノードがなくなるまで繰り返す．

SWAPS: リスト Ls において隣接する 2 ノード $u(u0, u1)$, $v(v0, v1)$ (u, v の順番に出現する) が，(1) v が u の先祖である，かつ (2) $u1 < v1$ である，場合に，2 ノードの始点の値を入れ替え，$u(v0, u1)$, $v(u0, v1)$ とする (リスト Ls において u, v の順番が入れ替わる)．

SWAPE: リスト Le において隣接する 2 ノード $u(u0, u1)$, $v(v0, v1)$ (u, v の順番に出現する) が，(1) u が v の先祖である，かつ (2) $v0 < u0$ である，場合に，2 ノードの終点の値を入れ替え，$u(u0, v1)$, $v(v0, u1)$ とする (リスト Le において u, v の順番が入れ替わる)．

上記第 2 の手続きにおいて，始点 (または終点) が連続する 2 ノードに関して「のみ」始点 (または終点) の入れ替えを行っているので， 1 度の入れ替えによって包含関係に変化が生じるのはこの 2 ノード間「のみ」である．従って，入れ替えにより，レンジによって表現される先祖子孫関係は (1 ずつ) 単調増加する．

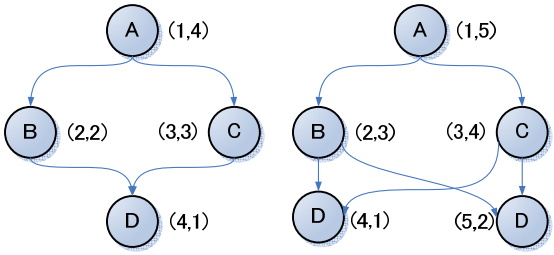

図 2 重なりレンジラベリング
Fig. 2 Overlapped Range Labeling

上記アルゴリズムの実行結果は，単調性より Agrawal の方法で抽出されるツリー (部分位相グラフ) で表現される先祖子孫関係をすべて包含し，極大な重なりレンジラベリング可能グラフ (部分位相グラフ) である．

図 2 左に 図 1 と同一の 4 個のノードからなるグラフから，図 2 右に 5 個のノードからなるグラフから，それぞれ重なりレンジラベリング可能グラフを抽出し，抽出結果に重なりレンジラベリングを行った具体例を示した．これらの具体例では，抽出前のグラフと抽出したグラフが同一になっている．

上記アルゴリズムは，SWAPS や SWAPE 中に 2 ノードの先祖子孫の判定を含む．この判定を高速に行うために，アルゴリズムの前処理としてグラフに Agrawal のアルゴリズムを用いて仮レンジ付与を行う．

ノード数が n, エッジ数が m であるグラフに Agrawal のアルゴリズムによってグラフにレンジ付与を行った結果，1 ノードに付与されるレンジ数が最大 l' であるとする．このとき，重なりレンジラベリング可能グラフ抽出のアルゴリズムの計算時間は $O(l'n^2)$ である．また，ラベル保持に必要な記憶容量は $O(n)$, 抽出したグラフにおいてノード間の先祖子孫関係を判定する計算時間は $O(1)$ であり，従来のレンジラベリング手法と同一である．

4.2 多次元分割

グラフの中には重なりレンジラベリング可能グラフ以外のグラフも含まれているため，一般のグラフへの問い合わせ処理を効率的に行うためには，更なる拡張が必要である．Agrawal の方法と同様に，本論文でも各ノードが複数のレンジを保持することを許容するが，Agrawal の方法の欠点を改善するために，伝播とは別の方法が必要である．

そこで，第 2 の拡張として「グラフの多次元分割」手続きを導入する．グラフの多次元分割は，グラフを複数の部分位相グラフに分割し，部分位相グラフの和集合によってグラフを表現する手続きのことであり，より厳密には，ノード数 n のグラフ G が与えられた時，k 個のグラフ $G1, G2, \ldots, Gk$ が以下の条件を満たす時，$(G1, G2, \ldots, Gk)$ のことをグラフ G の多次元分割と呼ぶ．

部分集合条件: $G1, G2, \ldots, Gk$ のノードは，すべて G のノードの部分集合である．

位相条件: グラフ G の任意の 2 ノード u, v に対して，

1. G で u が v の先祖になっていれば，$G1$ から Gk の少なくとも 1 個のグラフにおいて，u が v の先祖になっている．
2. G で u が v の先祖になっていなければ，$G1$ から Gk のいずれのグラフにおいても，u は v の先祖になっていない．

上記 $G1, G2, \ldots, Gk$ は G の部分位相グラフである．グラフ G の部分位相グラフには様々な種類が存在するが，ツリー，重なりレンジラベリング可能グラフなどに加えて G 自身，G の 1 本のエッジのみからなるグラフなども G の部分位相グラフである．

多次元分割を用い，任意のグラフに位相情報を表現するレンジを付与するアルゴリズムは以下の通りである．

第 1 に，グラフを多次元分割する．第 2 に，個々のグラフのノードにレンジを付与する．ここで i 番目 $(0 < i \leqq k)$ のグラフにおいて，ノード u に付与されたレンジを，ノード u の第 i 次元のレンジと呼ぶ．第 3 に，ノード u の第 1 次元のレンジから第 k 次元のレンジまでの k 個をまとめてノード u のレンジリストとする．

アルゴリズム全体でグラフの差分計算に $O(kn^2)$ の計算量が必要である．また，レンジを保持するために必要な記憶容量は $O(kn)$ である．さらに，多次元分割を用いた方法では，判定時に各次元 1 回のみレンジ比較を行えば十分であり，判定に必要な計算時間は $O(k)$ である．

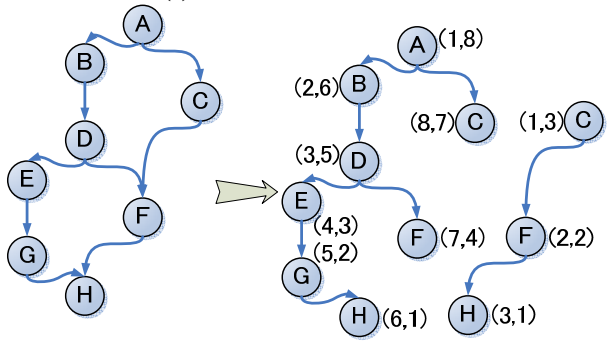

図 3 グラフの多次元分割
Fig. 3 Multi-Dimensional Graph Partition

図 3 にグラフを 2 個のグラフに多次元分割して，レンジを付与した結果が示してある．

4.3 拡張の組み合わせ

一般に，レンジを用いて任意のグラフの位相情報を表現するためには多次元のレンジが必要である．このため，グラフから部分位相グラフを抽出して 1 次元レンジを付与し，この結果を活用して，グラフ全体の位相情報を表現する多次元レンジを付与するという方法によって，従来の 1 次元のレンジラベリングを多次元のレンジラベリングに拡張する．

この際，グラフから抽出する部分位相グラフをツリーにする (Agrawal の方法で採用) か，重なりレンジラベリング可能グラフ (第 1 の拡張) にするか，2 通りの選択肢がある．また，1 次元レンジの結果を活用して，多次元レンジを付与する方法には，伝播を用いる方法 (Agrawal の方法で採用) と，グラフの多次元分割を用いる方法 (第 2 の拡張) の 2 通りの選択肢がある．

表 1 拡張レンジラベリング
Table 1 Extended Range Labeling

		多次元化方式	
		伝播	多次元分割
部分位相グラフ	ツリー	(TP)	TC
	重なりレンジラベリング可能グラフ	GP	GC

表 1 にこれら 2 種類の拡張の組み合わせバリエーションを示した．任意のグラフは上記 TP, TC, GP, CG の 4 種類どれを用いてもラベル付け可能である．TP は Agrawal の方式と同一，TC, GP, GC の 3 種類が本論文独自の方式である．

ノード数が n, エッジ数が m であるグラフに上記 4 種類の方法を用いてレンジ付与した時，1 ノードに付与されるレンジ数の最大数が，TP では l', GP では l'', TC では k', GC では k'' であるとする．単調性より，これら最大レンジ数の間には，$l'' \leqq l'$, $k'' \leqq k'$ の関係が必ず成り立つ．

ノードに付与されたレンジを用いてグラフ中の 2 ノード間の先祖子孫関係を判定するために必要な計算時間は，TP では $O(l')$, GP では $O(l'')$, TC では $O(k')$, GC では $O(k'')$ である．また，レンジを格納するために必要な記憶容量は，TP では $O(l'n)$, GP では $O(l''n)$, TC では $O(k'n)$, GC では $O(k''n)$ である．

さらに，レンジを付与するために必要な計算時間の観点から比較すると以下の通りである．TP に必要な計算時間は $O(l'm)$ である．GP では，重なりレンジラベリング可能グラフの抽出に $O(l'n^2)$, レンジの伝播に $O(l''m)$ の計算時間がかかるため，アルゴリズム全体では $O(l'n^2 + l''m)$ の計算時間がかかる．TC では，ツリーの抽出に $O(k'm)$, グラフの差分計算に $O(k'n^2)$ の計算時間がかかるため，アルゴリズム全体では $O(k'n^2)$ の計算時間がかかる．GC では，$O(l'n^2)$ の計算時間がかかる重なりレンジラベリング可能グラフの抽出と $O(n^2)$ の計算時間がかかるグラフの差分計算をそれぞれ k'' 回ずつ行うため，アルゴリズム全体では $O(k''l'n^2)$ の計算時間がかかる．

TP とその他のアルゴリズムを比較すると，$k'' \leqq k'$, $l'' \leqq l'$ の関係に加えて $k' < l''$ であることが期待されるため，記憶容量 (付与レンジ数) や，検索に必要な計算時間 (レンジ比較回数) の観点から，本論文で導入した方式は TP を改善すると期待される．ただし，レンジ付与に必要な計算時間の観点からは TP が最適である．

5. 計算機実験

5.1 データ準備

本計算機実験では，ODP [1] のカテゴリデータのうち，ノード「Top/World/Japanese/アート」をルートとし，このノードから到達可能なすべてのノードからなる部分グラフ，ノード「Top/World/Japanese/ビジネス」をルートとし，このノードから

到達可能なすべてのノードからなる部分グラフの 2 種類の部分グラフを用いる．ただし，これら 2 種類のサブグラフは，いずれもループを含んでいるので，強連結成分分解を行った上で，個々の強連結成分を縮約し，非巡回グラフに変換する前処理を行う．

5.2 付与レンジ数

表 2 合計・平均・最大レンジ数
Table 2 Range Numbers

アート	合計レンジ数	平均レンジ数	最大レンジ数
TP	2196	1.51	83
GP	2089	1.44	53
TC	3489	2.41	9
GC	2416	1.67	6

ビジネス	合計レンジ数	平均レンジ数	最大レンジ数
TP	3474	1.52	40
GP	3202	1.40	32
TC	4973	2.17	11
GC	3703	1.61	5

表 2 に，グラフ全体に付与される合計レンジ数，1 ノードに付与される平均レンジ数，1 ノードに付与される最大レンジ数を示した．TC, GC における最大レンジ数は，グラフの多次元分割における分割次元数に等しい．第 1 に合計 (平均) レンジ数の比較を行う．TP と GP を比較すると，TP に比べて GP のレンジ数が必ず小さくなる．2 種類の実データに適用したところ，それぞれ 4.9%, 7.8% レンジ数減少を実現した．しかしながら，TC は TP に比べて 59%, 43% レンジ数が増加，GC は TP に比べて 10.0%, 6.6% レンジ数が増加している．第 2 に最大レンジ数の比較を行う．実験の結果，GP, TC, GC すべての場合に最大レンジ数が減少している．減少幅は，GC が 93%, 88% と一番大きく，TC の 89%, 72%, GP の 36%, 20 % と続く．

5.3 レンジ比較回数

レンジ比較回数の比較実験を以下の方法で行った．2 種類のカテゴリそれぞれについて，グラフ中すべてのノードを，(1) ルートノードからから 2 本以下の枝を辿ることで到達可能なノード集合 U と，(2) それ以外のノード集合 V に分類する．ノード集合 U 中の 1 個のノード u，ノード集合 V 中の 1 個のノード v のすべての組合せに対して，u, v が先祖子孫の関係になっているかどうか判定を行い，判定に必要なレンジ比較回数をカウントする．

表 3 平均レンジ比較回数
Table 3 Average Range Comparison Number

	TP	GC
アート	1.56	1.24
ビジネス	1.79	1.27

表 3 に示したものは，1 組のノードの先祖子孫関係を判定するために，平均して何回のレンジ比較が必要であるかをまとめたものである．アートカテゴリ，ビジネスカテゴリの実験結果ともに GC の方が TP よりもレンジの比較回数が少なく抑えられる結果 (21%, 29% 削減) が得られている．

表 4 最大比較レンジ数
Table 4 Maximum Range Comparison Number

	TP	GC
アート	114	6
ビジネス	52	5

表 4 に示したものは，1 組のノードの先祖子孫関係を判定するために，最大何回のレンジ比較が必要であるかをまとめたものである．アートカテゴリ，ビジネスカテゴリの実験結果ともに GC の方が TP よりもレンジの比較回数が少なく抑えられる結果 (95%, 90% 削減) が得られている．

5.4 考察

第 1 に記憶容量 (付与レンジ数) に注目すると，「重なりレンジラベリング」のみを用いた拡張方式である GP が最善であり，TC, GC は TP より劣っていることが分かる．これは，「重なりレンジラベリング」はレンジの全体数削減効果をもたらすのに対して，「グラフの多次元分割」は効果がないことを意味している．

第 2 に検索速度に注目すると，「重なりレンジラベリング」と「グラフの多次元分割」の 2 つを用いた拡張方式である GC が，平均検索時間の面でも，最悪検索時間の面でも最善であることが分かる．

以上のことから，記憶容量，検索時間の両方で Agrawal の方式を改善するためには，重なりレンジラベリングのみを用いた拡張が適している．改善効果は小幅ではあるが，理論的にも，必ず良くなることが保証されており，確実な効果が得られる．また，検索時間の面で Agrawal の方式を大幅に改善するためには重なりレンジラベリングとグラフの多次元分割の両方を用いた拡張が適している．ただし，この方式は記憶容量の面で多少 Agrawal の方式より劣っている．拡張を行う際には，用途に応じて方式を使い分けるとよい．

6. おわりに

本論文では，非巡回グラフのための拡張レンジラベリング手法に関する研究成果を述べた．拡張のポイントは，「重なりレンジラベリング」の導入と，「グラフの多次元分解」の導入の 2 点である．今後は，様々なデータを用いて，本拡張の有効性の検証を行う予定である．

[文献]

[1] Open Directory Project, available at http://dmoz.org/.

[2] Extensible Markup Language (XML), available at http://www.w3.org/XML/.

[3] Document Object Model (DOM), available at http://www.w3.org/DOM/.

[4] XML Path Language (XPath), available at http://www.w3.org/TR/xpath.

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

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

鳥井 修 Osamu TORII

株式会社東芝 研究開発センター コンピュータ・ネットワークラボラトリー．1995 東京大学大学院工学系研究科計数工学専攻修士課程修了．データベースとデザインオートメーションの研究･開発に従事．情報処理学会，日本データベース学会会員．

白井 智 Satoshi SHIRAI

株式会社東芝 研究開発センター コンピュータ・ネットワークラボラトリー．2001 京都大学大学院情報学研究科知能情報学専攻修士課程修了．データベースとデザインオートメーションの研究・開発に従事．日本データベース学会会員．

金井 達徳 Tatsunori KANAI

株式会社東芝 研究開発センター コンピュータ・ネットワークラボラトリー．1989 京都大学大学院工学研究科情報工学専攻博士後期課程研究指導認定退学．データベースとデザインオートメーションの研究・開発に従事．情報処理学会，電子情報通信学会会員．