# A4-8　階層構造を識別可能な木節点の番号付け

佐藤 隆士，里本 智彦，小畑 喜平，潘 洪涛
大阪教育大学大学院総合基礎科学専攻数理情報コース
〒582-8582 大阪府柏原市旭ヶ丘 4-698-1 Tel: 0729-78-3671
{sato,tomo,kobata,han}@ss.osaka-kyoiku.ac.jp

**概要**　XML-DB における経路式質問を効率的に処理するための索引の提案である．従来，XML データの木構造の節点に振った番号を索引に格納し，索引のみで節点間の先祖子孫関係を知る方法が提案されている．本報告では，これを進め，木構造中での絶対的な位置を識別可能な番号付けを提案する．具体的には，レベルごとに異なる基数を用い，経路を整数にコーディングしている．

**キーワード**　XML-DB，階層構造，木節点の番号付け，半構造データ

# Numbering Scheme which Identifies Tree Structures

Takashi SATO, Tomohiko SATOMOTO, Kihei KOBATA, Taohan HONG
Course of Mathematical and Information Science, Division of Pure and Applied Science, Graduate School of Education, Osaka Kyoiku University

## １．はじめに

XML (拡張可能なタグ付き言語)は，インターネットの標準的なデータ交換手段として用いられるようになってきた．これに伴い，XML データを効率よく格納および検索する必要性が増している．XML 文書は木で表現できる階層構造をなしている．このため，階層をたどる経路式質問と呼ばれる特徴的な質問がある．

経路式質問を効率よく処理するため，経路に基付く索引が提案されている [1,2,3,4]．経路の途中は任意で階層の上位と下位を指定する正規経路式質問で効果的な索引も提案されている．木構造の節点に対応する要素，属性を表す節点に番号を振り，索引には，節点ごとに分解して番号とともに格納する．検索時には，付けられた節点番号から節点間の先祖子孫関係などの情報を得ることができる索引である [5,6]．しかし，Li ら [5] の方法では，節点対が，先祖子孫関係であるかどうかが分かるが，その関係が親子であるか，あるいは何世代離れた関係であるかに答えることはできない．また，$k$-ary 完全木に基づく番号付けによる Lee ら [6] の方法は，節点に付けられた番号から，木における絶対的な位置がわかるので，先祖子孫関係でなく，世代の間隔や兄弟節点間の位置関係も答えることができるはずである．しかし，番号を幅方向に振っているので親子の関係以外の計算は容易ではない．また，多くの利用されない仮想節点に番号を使うため，arity と木の高さが大きくなると，現実的でないことが指摘されている．

本稿では，節点の木における絶対的な位置を記憶し，且つ様々な節点間の関係を容易に計算できる方法を提案する．しかも，木のレベルごとに異なる arity を使用することにより仮想節点による番号の消費を少なくし，現実的に実装可能な番号付け手法を提案する．

## ２．層構造を識別可能な木節点の番号付け

木の節点に対して行う新しい番号付け手法を提案する．

### ２．１　数字の組による節点の位置表現

まず，節点の絶対的な位置を数字の組で表す方法である．絶対的な位置が分かれば，任意の節点間の先祖子孫関係だけでなく，具体的に先祖子孫間の経路長も知ることができる．また，兄弟，従兄弟の関係についても，どの程度離れているか正確に知ることができる．具体的には以下のように行う．高さ $h$ までの木について，$h$ 組の数字からなる番号を付ける（$h$ を越えるレベルにある節点については，２．３[A]参照）．レベル $n$ にある節点の，下から $h$-$n$ 個は 0 とする．従って，根は $h$ 個の数すべてが 0 である．レベル $n$ にある節点 $s$ の親を $p$ とするとき，$s$ の位置を表す数字の組の先頭の $n$-1 組は $p$ を受け継ぐ．$n$ 番目の数字は，$p$ の子を左から数えた番号を入れる．本の章，節などに付ける番号の組と同じであると考えれば分かり易い．

[例１]図１の実線部分の木について，数字の組による節点の位置表現を（ ）内に記入している．

□

### ２．２　節点のコード番号

数字の組による表現方法は，分かりやすいが冗長なため，記憶スペースの点では不利である．また，２節点間の位置比較のために，組を構成する個々の数字の比較を必要とし，手間がかかる．そこで，構成する数字の組と一対一に対応する単一の整数にコーディングすることを考える．木における任意の節点の子の数の最大値を $k$ とするとき，２．１の数字の組を $k$ を基数として単一の数にコーディングすればこの条件を満たすことができる．しかし，木の高さを $h$ とするとき，$k^h$ 以上の大きさの数になってしまう．例えば，$k$ および $h$ が 10 以上の大きさの木を仮定すると，32bit 整数には収まらない．

そこで，本稿では，レベルごとに異なる基数によるコーディングを行い，番号の大きさを抑える方法を提案する．レベル $i(0 \leqq i \leqq h\text{-}1)$ の基数を $k_i$，あるレベル $n(\leqq h)$ にある節点の数字の組を $(a_1, \cdots, a_n, 0, \cdots, 0)$ とするとき，

$$((\cdots(a_1k_1+a_2)k_2+\cdots+a_n\text{-}1)k_{n\text{-}1}+a_n)k_n\cdots k_{h\text{-}1} \quad （１）$$

とコーディングする．

ここで，各レベル $i$ にある節点の子の数をあらかじめ調べておくこととし，レベル $i$ の基数 $k_i$ は，ほとんどの子の数が，$k_i$-1 以下になるように決める．１が引かれているは，先頭から $i$ 番目の数が 0 は，レベル $i$-1 より浅いレベルの節点に使うことにし，レベル $i$ における子の番号を１から始めたためである．

[例２]図 １ の実線部分の木において，$k_0$=4, $k_1$=8 としたとき，コーディングされた番号を( )の右と下にそれぞれ 10 進数と 2 進数で書いている．

□

### ２．３　オーバフローフラグ

番号を $m$ bit でコーディングする．木の形状によりオーバフロー生じたことを表すフラグとして先頭 1bit を用意するため，実質 $m$-1 bit をコーディングに使用できる．

レベル $i$ の節点の子の数が $k_i$-1 を超えると，コード番号の一意性で保証されず，数字の組による表現法と一対一の関係が崩れる．この状態を子の数のオーバフローと呼ぶ．オーバフローを生じないようにするには，$k_i$ をレベル $i$ の子の数の最大値+1 になるようにすればよいが，特異的に大きな数の子をもつ節点に引きずられ，$k_i$ を大きくし過ぎると深いレベルの節点は非常に大きなコード番号を必要とすることになり，結果的にレベル方向のオーバフローを起こすことになるため得策でない．

そこで，稀なケースとして子のオーバフローは認めることとする．オーバフローフラグの立った部分については必要に応じ，当該XMLデータに直接アクセスして確認する，または別索引を用意するなどの対策を採るものとする．

[A]レベルのオーバフロー

レベル $n$ が，$\log_2(k_0k_1k_2\cdots k_{n\text{-}1}) \leqq m\text{-}1$ の場合は，（１）式で計算するが，それ以上深いレベルの節点は，オーバフローフラグを立て，その親と同じコードを割り当てる．

[B]子の数のオーバフロー

レベルのオーバフローは生じていないが，レベル $i$ の子の数が $k_i$-1 を超える場合は，オーバフローフラグを立て，$k_i$-1 番目の子と同じコードを付ける．その子の子孫についてもオーバフローフラグを立て，$k_i$-1 番目の対応する位置に子孫がある場合と同じコードを割り当てる．

[例３]レベル３以上でオーバフローを生じると

して, 図１のレベル３の破線部分にレベルオーバフローを示した．また, $k_0$=4 としたので，図１の右破線部分に子の数のオーバフローを示した．`*'マークはオーバフローフラグが立っていることを示している． □

## ３．位置関係の計算

節点に付けられた番号から，節点間の位置関係を知る方法を説明する．簡単のため，レベル $i$ の基数が２のべき乗で，$k_i=2^{w_i}$ と表される場合について説明するが, そうでない場合も同様な考えで計算できる．

### ３．１　絶対位置の復元

節点に与えられた番号の２進数表現の下位から，bit 幅 $w_{h-1}, w_{h-2}, \cdots, w_1, w_0$ を順に取り出し，数字に変換したものをそれぞれ，$r_h, r_{h-1}, \cdots, r_2, r_1$ とするとき，それらを逆順に並べて組にした $(r1, r2, \cdots, r_{h-1}, r_h)$は，その節点の数字の組による位置表現になる．

### ３．２　節点間の関係

節点に付けられた番号は, 木の pre-order 探索順で昇順に並ん(extended pre-order)でおり，且つ任意の節点の子孫の最大番号も簡単な計算で分かるので，２節点が，左，右，先祖あるいは子孫のいずれの関係であるか，これら節点に付けられた番号だけで簡単に計算できる．

提案の番号付けでは, 節点間の更に詳細な位置情報を計算できる特徴を有す．以下に例を示す．

[A] 最も近い共通先祖 (LCA: lowest common ancestor)

２つの節点の番号の bit パターンを比較すると，節点の共通の先祖が分かる．即ち，bit パターンの先頭から比較してレベル $n$ に相当する bit パターンまでが同じで，$n$+1 に相当する次の $w_n$ bit は異なるとき，LCA の番号は，レベル $n$ までの bit パターンの後に 0 をコーディング幅に届くまで追加したものになる．

[例４] 図１において, $k_0$=4 すなわち $w_0$=2 なので，01001，01010 と番号付けられた２つの節点のレベル１，即ち先頭の 2 bit 分が一致していることが分かる．従って，LCA の番号は 01000 となる． □

[B]先祖子孫関係にある節点間の距離

bit パターンを先頭から調べ，レベル $n$ に相当する bit パターンまで 0 がなく，$n$+1 に相当する次の $w_n$ bit が 0 の場合，その節点はレベルは $n$ にあることが分かる．先祖子孫関係にある節点について，レベルの差をとれば，節点間の距離が分かる．距離が１なら，親子関係である．

[例５] 図１において，$w_0$=2, $w_1$=3 なので， 10000 と 10010 と番号付けられた２つの節点のレベルはそれぞれ１および２であることが分かる. レベル差が１，即ち親子の関係にあることが分かる． □

[C]兄弟および従兄弟関係

２節点が同じレベルにある場合, LCA のレベルから節点間の兄弟あるいは従兄弟の関係がわかる．

以上は, オーバフローフラグの立っていない場合である．フラグが立っている場合は，最大レベルまで達している番号の場合は, その番号は木の深さ方向に縮退していること, あるレベルで子の数を表す数がそのレベルの基数-1 に達している場合は, 右方向に縮退していることに配慮しながらある程度節点間の情報を得ることはできる. 正確な情報を知るには，２．３で述べた方法によるものとする．

## ４．おわりに

XML-DB における経路式質問を効率的に処理するため, 索引に文書の階層構造を反映した番号を格納する．本稿では，XML 文書を表す木節点への新しい番号付け手法を提案した．これを用い，節点間の左右あるいは先祖子孫関係だけでなく, より詳細な兄弟あるいは親子関係などの情報を知ることができた. 各レベルの基数の選び方の詳細, 番号のオーバフローの具体的処理方法, 更新に伴う番号の付け替えなどの課題がある．

## 参考文献

[1]Brian F. Cooper, Neal Sample, Michael J. Franklin, Gisli R. Hjaltason and Moshe Shadmon: "A Fast Index for Semistructured Data", in *Proceedings of the 27th VLDB Conference*, Roma, 2001.

[2]山本陽平,絹谷弘子,吉川正俊,植村俊亮：XML 文書のための逆経路索引とその応用 ,(http://db-www.aist-nara.ac.jp/papers/xdd3.pdf).

[3]Chun Zhang, Jeffrey Naughton, David DeWitt, Qiong Luo: On Supporting Containment Queries in Relational Database Management Systems, in

*Proceedings of the 201 ACM SIGMOD Conference*, Santa Barbara, CA, May 2001.
[4]吉川正俊：XML エンジン, in *Proceedings of the DBWeb2001*, Kyoto, December 5-7 2001, pp. 33-34.
[5]Quanzhong Li and Bongki Moon: “Indexing and Querying XML Data for Regular Path Expressions”, in *Proceedings of the 27th VLDB Conference*, Roma, 2001.
[6]Yong Kyu Lee, Seong-Joon Yoo and Kyougro Yoo: “Index Structures for Structured Documents”, in *Proceedings of the ACM 1st Conference on Digital Libraries*, Bethesda, MD, March 1996, pp.91-99.

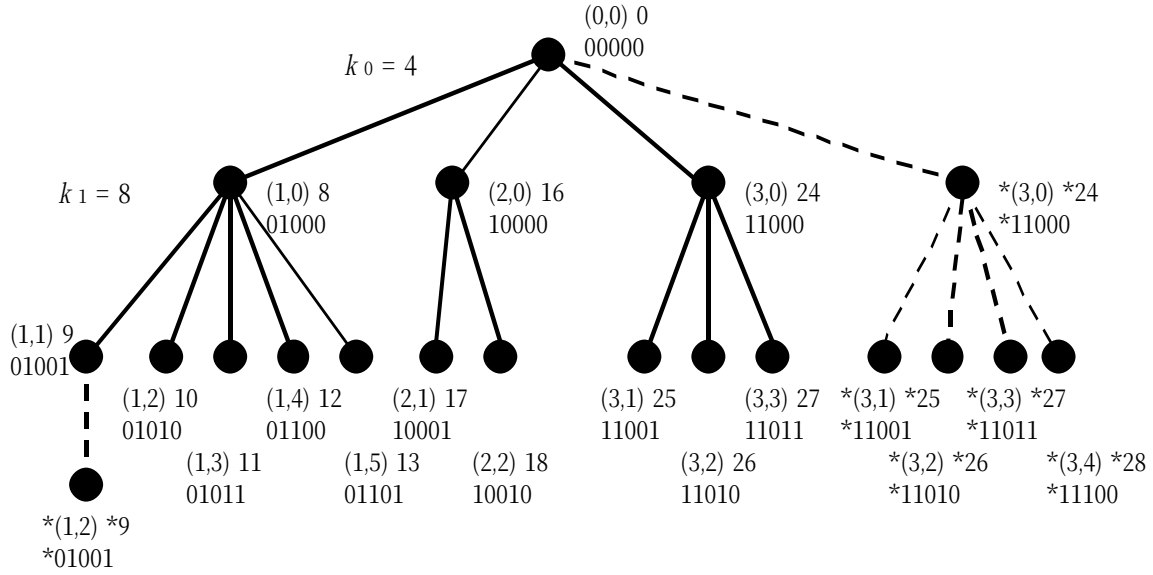


**図１　木節点の番号付け**