スキーマに基づいた XML ビュー更新可能性判定

数理第 7 研究室 66223 松崎幸太郎
指導教員 胡振江 准教授

1 はじめに

XML (eXtensible Markup Language) は汎用的なデータ記述言語として様々な分野で用いられている．最近では計算機の高速化に伴い，大規模なデータを扱う XML データベースが普及している．データベースの一部を取り出してユーザに見やすいように加工したデータをビューという．ユーザがデータベースを更新したい場合，ユーザがデータベースを直接書き換えることは安全性の面などで大きな問題があるので，ビューを更新しそれをデータベースに反映する方法をとりたい．この様に，ビューの更新からデータベースの更新を求める問題がビュー更新問題である．図 1 はビュー更新問題を簡単な図式で表したものであり，実線の矢印の経路で得られるビューと点線の矢印で得られるビューが一致する場合，ビューの更新とデータベースの更新が対応している．

多数のユーザがビュー更新を行おうとすると計算コストの面で問題である．あらゆるビューの更新がデータベースに反映できる，つまりビュー更新可能とは限らないことから，前処理としてビュー更新可能かどうかを予め軽量に判定できれば便利である．データベースの構造を規定するスキーマのみを用いてビュー更新可能性を判定する方法が Dayal と Bernstein [1] によって関係データベースの上で議論された．Wang ら [2] は関係データベースから XML のビューを取り出す場合において Dayal と Bernstein の議論を発展させた．しかし XML データベースにおける議論は我々の知る限り存在しない．

本研究では，XML データベースにおけるビュー更新可能性判定を考える際，更新操作を部分木の削除と挿入に限定し，スキーマに対しても制限を加えた枠組みにおいて考える．この枠組みにおいてスキーマに基づいた XML データベースの更新可能性判定問題の定式化と判定方法を提案する．

本研究の貢献は以下の 2 点にまとめられる．1 点目は，ビュー更新可能性判定問題を XML データベースの上で扱う枠組みを定式化したことである．2 点目は，一定の制約を加えた中でビュー更新可能性判定のアルゴリズムを与え，その正しさを示したことである．

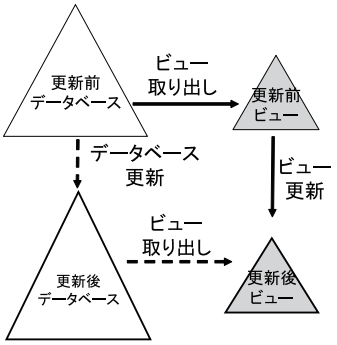

図 1　ビュー更新問題の模式図

2 木構造データとスキーマ木

本研究では，XML 文書を木構造データで表す．XML 文書を表す木構造データは中間ノード (Node) か葉ノード (Leaf) で構成される．木 t を構成する各ノードとそのノードをルートノードとする木 t の部分木が同一視する．

スキーマ木は木構造データにおける構造の制限を表す．スキーマ木はユニークノード (Unique)，スターノード (Stared)，葉ノード (Leaf) 構成される．スキーマ木の表す木の構造の概略は次の通りである．

- スキーマ木のノードとそれに対応する木構造データのノードは要素名が同一である．
- ユニークノードは木構造データにおいて子ノードとして必ず 1 個だけであることを表す．
- スターノードは木構造データにおいて子ノードとして 0 個以上あることを表す．
- 葉ノードは木構造データにおいても葉ノードで，中身は任意の文字列である．

スキーマ木の例を図 2 で表す．

スキーマ木と木構造データの対応として，インスタンスを定義する．スキーマ木のノード s の木構造データ t におけるインスタンスの集合 $instances(s,t)$ はノード s に対応する木 t のノードの集合であり，以下の式

$$instances(s,t) = \{t' \mid t' \trianglelefteq t\} \cap vtrees(s)$$

で定義する．ここで $vtrees(s)$ はスキーマ木 s によって規定される木構造データの集合である．

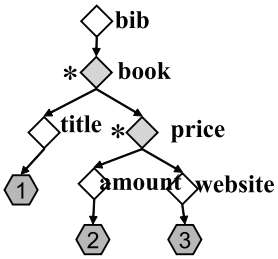

図 2　スキーマ木の例

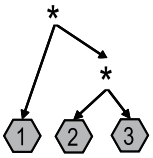

図 3 図 2 のノード “book” のクロージャ

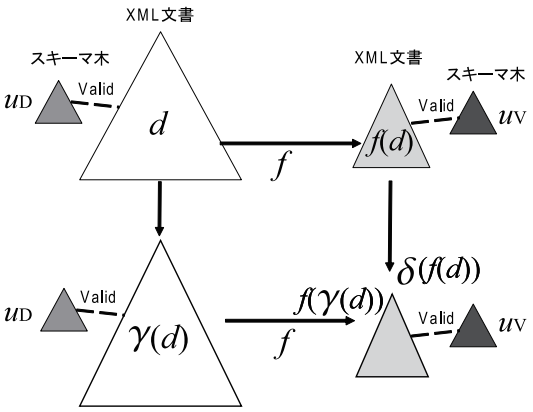

図 4　更新可能性判定の図式

3　ビューとデータベースの対応

本研究では具体的に特定のクエリ言語を扱わず，ビューの葉ノードがデータベースのどの葉ノード から取り出されたのかを表す関数 gen が既知であることを仮定する．この関数 gen を対応関数と呼ぶ．実際には与えられたクエリから対応関数 gen を導くが，本論文では具体的な方法は触れない．関数 gen には幾つかの制約を設けるが，本稿では触れない．

関数 gen からスキーマの上での対応関数 $dgen$ を導く．データベースを d，ビュー定義関数を f として，ビューのスキーマ木の葉ノードからデータベースのスキーマ木の葉ノードへの単射な関数 $dgen$ が，ビューのスキーマ木における任意の葉ノード s_{V} に対し，以下の式

$$\forall v' \in instances(s_{\mathrm{V}}, f(d)),\\ gen(v') \in instances(dgen(s_{\mathrm{V}}), f(d))$$

を満たすように定義できるとする．このとき，関数 $dgen$ は，ビューの $dtree$ の各葉ノード s_{V} に対し，集合 $instances(s_{\mathrm{V}}, f(d))$ から適当に 1 つ取ることで前述の制約式から定義できる．

4　更新可能性判定問題の定式化

図 4 は更新可能性判定問題を図式化したものである．ビューが更新操作 δ で更新可能であるとは，$f(\gamma(d)) = \delta(f(d))$ を満たすようなデータベースの更新操作 γ が存在することである．本研究では，ビュー更新 δ とデータベースの更新 γ は部分木 1 つの削除または挿入であるとする．

5　スキーマ木における構造の比較

本論文の主要な結果がスキーマ木の構造を比較することで更新可能性が判定できるというものである．スキーマ木そのものでは比較に必要のない構造の情報が含まれるので，比較に必要な構造であるスターノードと葉ノードを抽出したものとしてクロージャを導入する．例えば図 2 のスキーマ木において，ノード `book` のクロージャは図 3 になる．ノード n のクロージャを $closure(n)$ と記述する．

ビューのスキーマ木内のあるノードにおけるクロージャを C_{V} とする．クロージャ C_{V} に現れるデータベースのスキーマ木の葉ノード全てに関数 $dgen$ を適用し置換した新たなクロージャを $dgen(C_{\mathrm{V}})$ とする．クロージャ $dgen(C_{\mathrm{V}})$ とデータベースのスキーマ木のノードにおけるクロージャ C_{B} が一致する場合 $C_{\mathrm{V}} \overset{dgen}{\mapsto} C_{\mathrm{B}}$ と記述する．

6　更新可能性判定定理

本節では削除可能性判定定理のみ紹介する．挿入可能性の定理は割愛する．

定理　(削除可能性判定定理)　データベースを d_0, ビューを $f(d_0)$，データベースのスキーマ木の部分木を s_{D}, ビューのスキーマ木の部分木を s_{V} とする．この時，スターノード s_{D} が

$$closure(s_{\mathrm{V}}) \overset{dgen}{\mapsto} closure(s_{\mathrm{D}})$$

を満たす u_{D} において最も深いノードである時，ビュー $f(d)$ ノードの集合 $instances(s_{\mathrm{V}}, f(d))$ の任意の要素をルートノードとした木は削除可能である．　□

参考文献

[1] Umeshwar Dayal and Philip A. Bernstein. On the correct translation of update operations on relational views. *ACM Transactions on Database Systems*, Vol. 7, No. 3, pp. 381–416, 1982.

[2] Ling Wang, Elke A. Rundensteiner, and Murali Mani. Updating XML views published over relational databases: towards the existence of a correct update mapping. *Transactions on Knowledge and Data Engineering*, Vol. 58, No. 3, pp. 263–298, 2006.