データベースシステム 121－19
情 報 学 基 礎 58－19
（2000. 5. 26）

スキーマのない 多様なXML文書のリポジトリに対する 問合せ処理について

絹谷 弘子　吉川 正俊　植村 俊亮
奈良先端科学技術大学院大学
情報科学研究科
〒631-0101 奈良県生駒市高山町 8916-5
Tel : 0743-72-5336
E-Mail : {hiroko-k, yosikawa, uemura} @is.aist-nara.ac.jp

概要

XMLでは応用が進むにつれ従来の構造化文書にはなかった利用がなされるようになってきている．たとえば，SGML文書には厳格な固いスキーマが必須であったが，XMLではスキーマを持たない整形式の文書も許されている．本稿ではスキーマのない多様な不定形のXML文書のリポジトリに対する問合せにおける問題点を明らかにする．さらにスキーマについての情報を持たずに標準の名前空間に属する要素や属性とキーワードを用いた利用者の問合せに対し，必要な論理単位となる部分文書を得るための方法について提案する．

On Query Processing of Miscelaneous Well-formed XML Documents

Hiroko Kinutani　Masatoshi Yoshikawa　Shunsuke Uemura
Graduate School of Information Science
Nara Institute of Science and Technology (NAIST)
8916-5,Takayama, Ikoma, Nara 630-0101,Japan
Tel : 0743-72-5336
E-Mail : {hiroko-k, yosikawa, uemura}@is.aist-nara.ac.jp

Abstract

Progressing and spreading applications of XML, we have new sorts of XML documents diffrenet from traditional strucutred documents. For example, SGML documents have to comply with their schemas. XML documents don't require to have their schemas. In this paper, we clarify the issues related on query processing of miscelaneous well-formed XML documents. Then we define a new function which serves as a basis for identifying appropriate XML subdocuments as results of such queries.

1 はじめに

XML(eXtensible Markup Language)[12, 6] では，応用が進むにつれて従来の構造化文書では想定されない利用方法が出現している．W3C で制定中の多くのデータフォーマット[1]，SVG(Scalable Vector Graphics), MathML(Mathmatical Markup Language), SMIL(Synchronized Multimedia Integration Language), RDF(Resource Description Framework), XHTML(eXtensible Hyper Text Markup Language) も XML に基づいている．XML 名前空間 [15] によって文書中の要素型や属性を識別する方法は，スキーマ[2]を持たない文書においても利用可能なので，様々な XML 文書中で利用されている．名前空間を利用する要素集合は，URI[14] で識別する．例えば，SVG は "http://www.w3.org/2000/svg-20000303-stylable"，MathML は "http://www.w3.org/1998/Math/MathML"，SMIL は "http://www.w3.org/TR/REC-smil"，RDF は "http://www.w3.org/1999/02/22-rdf-syntax-ns#"，XHTML は "http://www.w3.org/1999/xhtml" で識別する．名前空間は業界標準の要素集合を識別するためにも使われている．例えば Dublin Core[2] では 15 の要素集合を定義している．我々はこの 15 の要素型を XML 文書作成において Dublin Core の意味において利用することができる．従って，文書の一部の構造に外部の名前空間で定義されている要素型や属性を利用することで，文書全体としてはスキーマのない整形式でありながら下部構造に共通性のある XML 文書が多数出現している．本稿では図 1 のように，文書の一部に共通の名前空間の要素を持っている XML 文書を対象とする．我々は，従来の構造化文書で必要であったスキーマ設計作業は難しく，XML の利用者にとってスキーマ設計を行なう代わりに名前空間という枠組を利用するほうが容易であることから，このような XML 文書は急増すると考える．従って，スキーマのない多様な XML 文書を集めたリポジトリに対する効率的な管理，検索，格納方法が必要となる．

共通の名前空間を利用している XML 文書についての問合せでは，名前空間で限定されている要素型と文字列を指定して検索することができる．従って，文書の要素を単位とした部分文書の検索が可能である．しかし，従来の検索手法では，検索条件に適合する要素を抽出するか文書全体を抽出する．XML

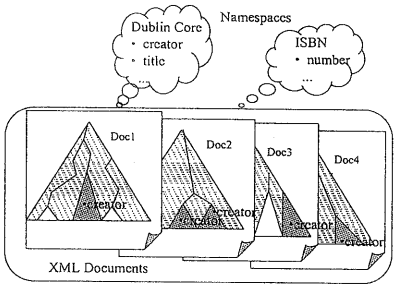

図 1: 共通の名前空間を利用している XML 文書

の構造から検索条件に関連する部分文書を自動的に抽出することで，利用者にとってより検索要求を満たす情報提供ができると考えられる．利用者がこのような XML 文書リポジトリに問合せをする場合に必要な方法は次の通りである．

1. 利用者が名前空間についての知識だけで，文書の論理構造の知識なしに問合せできる方法．
2. 少数の単語の入力から利用者の欲しい文書内容を探して利用者に提供する方法．
3. 問合せに用いた単語や要素型，属性の出現回数や構造上の出現位置を考慮した結果の表示と結果の絞り込み．

本稿では，利用者が XML 文書についてのスキーマについての情報を持たずに標準の名前空間に属する要素や属性といくつかの単語を用いて問合せを行い，必要な文書内容を含んでしかも名前空間の要素や属性について文書中で文脈としての論理単位となる部分文書を得るための方法を提案し，部分文書を特定するアルゴリズムの有効性を検証した実験について報告する．第 2 章では本研究の位置づけと関連研究について述べる．第 3 章では名前空間を利用した問合せについて明らかにし，問合せを満たす部分文書の特定方法について定義する．第 4 章では定義した文脈ノードを求める実験について報告し第 5 章ではまとめと今後の課題について述べる．

2 本研究の位置づけ

2.1 XML における文書検索とデータ検索の性質

XML は文書とデータという二つの異なった性質を持っている．文書としての性質を持つ XML インスタンスは，人間が読んだり印刷するために利用される．一方，データとしての性質を持つ XML インスタンスは，プログラムによって解釈するために利

[1]XML を用いて設計されたマーク付け言語を "XML アプリケーション" あるいは "語彙 (Vocabulary)" と呼ぶ

[2]現状では，名前空間を利用した XML 文書はすべて整形式となり，DTD では名前空間を指定できない．XML のスキーマ定義を XML で記述する方法 (XML Schema) が提案されているが，いまだ XML の勧告にはいたっていない．

用される．この二つの異なった性質を合わせ持つXMLインスタンスも存在する．スキーマを持たないXML文書検索は，データ構造が定義されているデータベースの検索と全く論理構造のない文書検索の中間に位置する．すなわち各々のXML文書についてXMLプロセッサを利用して文書構造を解析することで，文書構造を考慮した検索が可能となる．

2.2 関連研究

XMLの基となっているSGML[3, 4, 5]が構造化文書を記述するための言語であったため，従来の構造化文書データベースの研究では，SGML文書を対象として扱ってきていた．SGML文書では文書型定義（DTD）が必須であったため，DTDに従っているSGML文書を対象としていた．

スキーマを持たないXML文書インスタンスからスキーマを抽出する研究がある[10, 11]．しかし，各XML文書インスタンスごとにスキーマが異なる場合は，文書インスタンスとスキーマの数が等しくなり，スキーマから共通の構造を抽出することも困難となる．

一方Web上の多量の文書から必要な文書を探すためのサーチエンジンは，HTML文書を対象としている．HTML文書は，テキストのビューについてのタグ付けを行っているだけなので文書内容について記述する方法がない．従って，問合せ結果は問合せに用いた語句を利用する一般的な文書検索の方法で求める．しかし，これらの問合せの限界は，文書の一部を対象とした問合せを認めないことである．さらに，問合せ結果の単位（粒度）も文書単位で固定されている．

XMLサーチエンジンもいくつか公開されている[1, 8]．XMLサーチエンジンでは，XMLのタグを利用して検索部分の指定，検索結果の利用法の指定ができるので従来のHTMLを対象とした検索エンジンに比べ，より正確な情報の提示が可能となる．XSet[1]は主記憶上のデータベースとサーチエンジンの組み合わせでXMLをデータ格納言語として利用している．XSetの問合せは，次のような整形式のXML文書で与える．

```
<PERSON><FIRST>Ben</FIRST>
    <OFFICE CLEAN='NO'>443</OFFICE>
</PERSON>
```

問合せ結果は，問合せの構造を少なくとも含んでいるXML部分木となる．従って次のXML部分文書が結果となる．

```
<PERSON><FIRST>Ben</FIRST>
        <LAST>Zhao</LAST>
 <OFFICE CLEAN='NO' WINDOW='YES'>443</OFFICE>
</PERSON>
```

このシステムでは，タグの値はタグの順序の文脈で意味が異なることに基づいている．例えば，PERSON → HOME → ADDRESS と BUSINESS → CONTACTINFO → SHIPPING → ADDRESS では，ADDRESS要素の値の意味が異なるのでタグを含めた問合せを行う．しかしこのシステムでは，あらかじめデータベース中の文書構造がわかっていなければ問合せができない．従って利用者が問合せ時に文書構造についての知識を持っているようなXML文書についての問合せが対象となっている．

一方，XML文書検索に情報検索の技術を導入し文書の各要素の特徴量を文書の葉にあるテキストに索引づけをして，上位構造の要素ノードは下位ノードの出現値を積算するBUS(Bottom Up Scheme)[7]がある．BUSでは，文書中の任意の要素を単位として利用者の問合せと要素の特徴量から関連性の高い部分文書を取り出すことができる．この応用として，XMLサーチエンジンXRSが公開されている[8]．このサーチエンジンでは，構造に関する問合せはDTDをもとに行っている．また，問合せ条件に関連する祖先要素も取り出すことができる．例えば，PARAGRAPH要素内に文字列"XML, search"を含むという条件を与えた場合，"Get the CHAPTER whose PARAGRAPH contains 'XML, search' "という問合せで，PARAGRAPH要素の祖先要素であるSECTION, CHAPTERも取り出すことができる．しかし，取り出す部分を明示的に指定しなければならない．

すでにXML文書への問合せ言語も多数提案されているが，W3Cの勧告とはなっていない[13]．現在提案されているXML文書への問合せ言語では，構造と内容に基づく問合せが可能である．しかしこれらは構造や内容についての問合せ条件を満たす文書や部分文書をすべて取り出すが，該当する部分文書を自動的に特定したり，問合せと文書との関連性をランキングする機能は想定されていない．

我々は，利用者が文書構造についての情報を持たずに，標準の名前空間に属する要素や属性といくつかの単語を用いる問合せに焦点を絞り，問合せに関連する部分文書を自動的に特定する方法を提案している[9]．本稿では，その有効性を検証し，問題点について考察する．

3 名前空間を利用したXML文書への問合せ

```
<?xml version="1.0"?>
<bookstore xmlns:dc="http://purl.org/dc/elements/1.0/"
     xmlns:uri="http://url.org/elements">
  <name uri:loc="http://db-www.aist-nara.ac.jp">
     NAIST Publishing Inc.</name>
  <book>
   <creators>
    <dc:creator>Hiroko</dc:creator>
    <dc:creator>Yohei</dc:creator>
   </creators>
   <dc:title>XML</dc:title>
   <section>
    <dc:title>SGML</dc:title>
    <body>It is ....</body>
   </section>
  </book>
  <book>
   <creators>
    <dc:creator>Take</dc:creator>
   </creators>
   <dc:title>Image DBs</dc:title>
  </book>
  <book>
   <dc:creator>Taka</dc:creator>
   <dc:title>Multimedia DBs</dc:title>
  </book>
  <proceedings>
   <editors>
    <dc:creator>Masa</dc:creator>
   </editors>
   <dc:title>DB Workshop</dc:title>
   <paper>
    <authors>
     <dc:creator>Hiro</dc:creator>
    </authors>
    <dc:title>Video DBs</dc:title>
   </paper>
   <paper>
    <authors>
     <dc:creator>Masa</dc:creator>
    </authors>
    <dc:title>XML DBs</dc:title>
   </paper>
  </proceedings>
</bookstore>
```

図 2: 電子書店の XML 文書例

XML 名前空間の仕様では，XML のデータモデルを拡張して，URI で要素型名と属性名を限定する．名前空間を含んだ要素型名と属性名をそれぞれ拡張要素型名，拡張属性名と呼ぶ．図 2 は，二つの名前空間を利用している電子書店の XML 文書例である．図 3 は，その木構造表現である．各中間ノードは要素型名か属性名のラベルがつけられている．また，葉ノードは，ラベルが内容であるようなテキストノードを示している．文書構造上の 1 から 51 までのノード番号は，後の参照のためのものである．

3.1 単純 XML 問合せ

標準の名前空間の要素集合を使っているが共通のスキーマのない XML 文書のリポジトリから情報を取り出す場合，利用者が行なう問合せは，拡張要素型（あるいは属性）名とキーワードの組の連言であると想定する．この問合せの一部は，XPath[16] で記述することができる．しかし，問合せから論理単位となる部分文書を自動的に特定するような記述能力はない．ここでは，経路式の表現と述語表現に XPath で定義されている記法を利用する．

e, θ と v をそれぞれ拡張名，演算子と文字列とする． また，θ を '=' あるいは，'contains' とする．単純 XML 問合せは，述語 $(e\ \theta\ v)$ として表すことができる．ここで，$(e\ \theta\ v)$ は，次のことを満たすノード集合と解釈する．i) ノード名が e; ii) 内容が 文字列 v と等しい（θ が '=' の場合）か文字列 v を含む（θ が contains の場合）．このノード集合は，XPath では，"$//e[.//*[\theta\ v]]$" と表すことができる．

例 1 図 2 に示した電子書店の XML 文書例において次に示す単純 XML 問合せを考える：

```
(dc:title contains 'DB')                    (Q1)
```

この問合せ (Q1) では，i) dc:title 要素以下の部分文書内容が文字列 'DB' を含んでいるような適当な部分文書を取り出す という意味を持つものとする．

XPath では，XML 文書の根ノードの子孫（子あるいは子の子..）ノードの dc:title ノード以下のノードの内容に "DB" に含んでいるような部分木の根ノードを得る問合せは，次のように表すことができる：

```
//*[.//dc:title[.//*[contains(text(),'DB')]]]
```

図 3 中のノード {1,6,7,8,14,16,18,19,20,28 ,30} が部分文書を表す木の根ノードになる．しかし，図 3 中の 2,3,4,5 を根とする部分木は，この問合せの条件をみたす要素が全く存在しない部分木であり，問合せ内容とは文脈を異にすると考えられる．従ってノード 1 は部分文書の根ノードとしては冗長な情報が含まれるので問合せ結果としては不十分である．

単純 XML 問合せの結果となる部分文書を特定するためには，部分文書の根ノード（ここでは文脈ノードと呼ぶ）を特定する必要がある．そしてこの部分文書をこの単純 XML 問合せの論理的な単位とする．我々はこの文脈ノードを XML 文書の根ノードと単純問合せの述語を満たすノードの間のノードの中から特定する．(テキストノードは除外する．)

3.2 問合せを満たす論理単位となる部分文書の同定

ここで (dc:title contains 'DB') を満たすノードについて文書構造から他の部分木と文脈が異なることが識別可能な部分木を求める．この述語を満たすノードは，図 3 ではノード番号 14,16,18,28,30 が該当するノードである（これらを

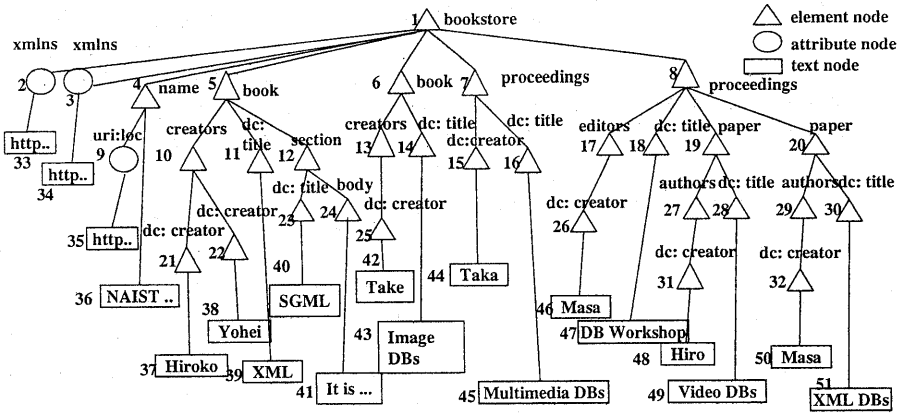

図 3: 電子書店の XML 文書例の木構造表現

カレントノードと呼ぶ）．文書中の経路は，それぞれ/bookstore/book[2]/dc:title, /bookstore/proceedings[1]/dc:title, /bookstore/proceedings[2]/dc:title, /bookstore/proceedings[2]/paper[1]/dc:title, /bookstore/proceedings[2]/paper[2]/dc:title である[3]．この経路から，dc:title 要素は，proceedings と paper の 2 種類のタイトルの意味で使われていることがわかる．そのため，proceedings のタイトルの部分文書と paper のタイトルの部分文書の範囲を文書構造から同定する．このカレントノードを含む部分木の根ノードは，それぞれの経路式を根に向かってたどる時の中間ノードである．文書の根からの経路式が構造上の内容を反映していると考えると，部分木内の上位ノードの経路式[4]に重複のないものが論理単位とみなせる．そこで，カレントノードの親ノードを最小の論理単位として[5]，カレントノードの祖先ノードについての部分木が重複経路式を持たないかどうかを判定し，重複経路式のない極大の部分木の根ノードをカレントノードの同一文脈部分木ノードとする．言い換えると，同じ経路式を複数含む場合は要素の繰り返しを含んでいることを意味しているので，段落や章の繰り返しに相当する可能性と考えられ，文脈の変り目と解釈できる．これは，カレントノードから一代づつ祖先をたどって祖先ノードの兄弟ノードに祖先ノードと同じ名前のノードがないような極大の部分木をつくることと同義である．我々は，この極大の部分木の根ノードを文脈ノードと定義する．

定義 1 (文脈ノード $c(n)$) テキストノード以外（中間ノード）の *XML* 文書 D 中の各ノード n に対して，D 中の文脈ノード $c(n)$ は次の条件を満たす：

(1) $c(n)$ は，カレントノード n と D の根ノード（n_d と書く）を結ぶ経路式上のノードである．$n = n_d$ の場合は $c(n) = n_d$ とする．

(2) T を $c(n)$ を根とする D の部分木とする場合，すべての $c(n)$ と n の中間ノード $m(\neq n, c(n))$ について，T 中に経路式 $/n_d/\ldots/c(n)/\ldots/m$ がただ *1* つであり，m と同じ名前の兄弟ノードを持たない．

例 2 表 1 は，図 3 中の各ノード n と文脈ノード $c(n)$ の対応を表す．この例では D 中の中間ノードに対応する文脈ノードの種類は要素の出現順を考慮すると 7 種類となる．また，要素の出現順を考慮しないと 4 種類となる．

表 1: 文書中のノードと対応する文脈ノード

ノード	文脈ノード	文脈ノードまでの経路式
1,2,3,4,5,6,7,8,9	1	/bookstore
10,11,12,21,22 23 24	5	/bookstore/book[1]
13,14,25	6	/bookstore/book[2]
15,16	7	/bookstore/proceedings[1]
17,18,19,20,26	8	/bookstore/proceedings[2]
27,28,31	19	/bookstore/proceedings[2]/paper[1]
29,30,32	20	/bookstore/proceedings[2]/paper[2]

定義 2 (ノード集合の文脈) D 中のテキストノード以外のノード集合 $\{n_1, \ldots, n_k\}$ に対して関数 **context** を次のように定義する：

[3]dc は名前空間の接頭語である．紙面の制約から URI に展開していない．

[4]経路式については，経路の出現順は考慮しないものとする．

[5]カレントノードは，XPath を利用して容易に求めることができる．また，名前空間を利用した要素名を指定した場合，その要素が文書中での他の要素との関係を求めるため，カレントノードは，文脈ノードにしないこととした．

D 中のノード集合 $S = \{n_1, \ldots, n_k\}$ に対して，**context**$(S) = \{\ c(n_1),\ \ldots,\ c(n_k)\ \}$

XPath 式はこの **context** 関数の引数として指定できることに注意されたい．

例 3 表 1 より次の等式が成立することがわかる：

```
context(//dc:title)
= context({11,14,16,18,23,28,30})
= {5,6,7,8,19,20}
context(//dc:creator)
= context({15,21,22,25,26,31,32})
= {5,6,7,8,19,20}
```

例 4 (単語による問合せの文脈) 単純問合せ "$(\theta\ v)$" の文脈は，次のように定義できる：**context**$(//*[\theta\ v])$ "*" は，XML 文書 D 中の根ノードのすべての子孫ノードを意味する．この問合せは，従来の HTML 検索の述語に相当する．

例 5 図 3 から，述語 (dc:creator contains 'Masa'), (dc:title contains 'DB') に対する部分文書の文脈ノードが次のように取り出せる．

```
context(//dc:creator[contains(text(),'Masa')]
)
= context({26,32})
= {8,20}
context(//dc:title[contains(text(),'DB')])
= context({14,16,18,28,30}
= {6,7,8,19,20}
```

4 文脈ノード特定実験

文脈ノードの定義に従い，いくつかの XML 文書について文脈ノードを求め，それらの特徴と集計を行った．各 XML ファイルごとに，(1) 実体の展開，文字コードの統一，名前空間の統一のため，正規化 (canonicalization) を行い，(2) パーズしながら DOM 木を走査して，各ノードについて，対応する文脈ノードを求める．(3) 各文脈ノードをさしているテキスト以外のノードの数を集計した．文脈ノードの経路式について，出現の順番を考慮しないものと順番を考慮する二通りの集計を行った．対象としたファイルは，図 2 の XML ファイル，W3C XML 関係の仕様書，Tipstar から作成した XML ファイルなどである．

ここでは，W3C の仕様書の XML ファイルについての結果を示す (表 2)．ここで使用した XML の仕様書類は，入れ子構造が深く文脈ノードもレベルが深くなっている．テキスト以外のノード 3153 について対応する文脈ノードを求めた．文脈ノードの種類は，ノードの出現順を考慮しない場合 43 種類となりテキスト以外の全ノード数のほぼ数%でおさえられていることから抽出する部分文書の上位構造の種類が限定されることがわかる．これは，文脈ノードの定義が同じ要素型の兄弟ノードの存在に依存していることによる．

実験 1 (XSLT の仕様書 XML 文書に対する問合せ) 図 4 のように，XSLT の仕様書の XML 文書[6]に対しての問合せについての文脈ノードを示す．カッコ内の要素型が述語を満たすノードである．

実験 2 (名前空間の要素と文字列を指定した問合せ) 図 4 のような EDI の XML 文書[7]に対しての問合せについての文脈ノードを示す．カッコ内の要素型が述語を満たすノードで，抽出文書の例が四角形内に表す．抽出部分文書は，この問合せに対して文脈上の最小単位となっている．

4.1 考察

スキーマのない多様な XML 文書を効率的に処理するためには，XML プロセッサで各文書の構造を解析すると同時に各文書に関する統計を保持する必要がある．本稿では，検索結果として取り出す部分を特定するための文脈ノードを定義した．この文脈ノードは文書を走査して構造上の最小の単位として有効であることが実験によって検証できた．また，文書の根ノードから文脈ノードまでの経路式が文脈ノード以下の部分文書の構造上の役割を示していることが確認できた．しかし，文書構造の下位部分に同じ要素型が兄弟ノードとして多数出現している場合は，文脈ノード c(n) の総数が多くなり，抽出文書の粒度が小さくなる．逆に文書の上位部分に同じ要素型が兄弟ノードとして多数出現している場合は，c(n) の総数も少なく抽出文書の粒度が大きくなるというトレードオフの関係が確認された．さらに，構造の似通っている XML 文書への問合せ結果として抽出する部分文書が文書ごとに異なるという限界も認められた．そのため，文脈ノードの決め方にある程度のヒューリスティクスを導入する必要がある．

5 まとめと今後の課題

本稿ではスキーマのない多様な XML 文書に対して利用者がキーワードや標準の名前空間に属する要素型や属性を用いた問合せと問合せ結果の部分文書の特定方法について述べた．問題点と課題は次の通りである．

[6] http://www.w3.org/TR/1999/REC-xslt-19991116.xml
[7] http://www.eccnet.com/xmledi/guidelines-styled.xml

実験 1 文字列 "siblings" を含む部分文書の文脈ノード （→の右のノードが述語をみたしている）:

```
context(//*[contains(text(),'siblings')])
 /{}spec[1]/{}body[1]/{}div1[5]/{}div2[2] → (/{}p[6])
/{}spec[1]/{}body[1]/{}div1[11]/{}div2[5] → (/{}p[1])
/{}spec[1]/{}body[1]/{}div1[4]/{}div2[2]/{}ulist[2]/{}item[2] → (/{}p[1])
/{}spec[1]/{}body[1]/{}div1[7]/{}div2[7]/{}ulist[2]/{}item[1] → (/{}p[1])
```

実験 2 ecc 名前空間に属する要素型 definition に文字列 "element" を含む部分文書：

```
context(//{http://www.eccnet.com/namespaces/ecc-namespace}definition [contains(text(),'element')])
 {http://www.eccnet.com/namespaces/ecc-namespace} report[1]/body[1]/section[2]/p[2] → (
/definition[1] ''data element'')
{http://www.eccnet.com/namespaces/ecc-namespace} report[1]/body[1]/section[2]/p[2] → ( /definition[2]
''compound data elements'')
```

部分文書

```
<p> The basic unit of information in an EDI message is the <definition>data element</definition>. For
an EDI invoice, each item being invoiced would be represented by a data element. Data elements can
be grouped into <definition>compound data elements</definition>, and data elements and/or compound
data elements may be grouped into <definition>data segments</definition>. Data segments can be
grouped into <definition>loops</definition>; and loops and/or data segments <emphasis>form business
documents</emphasis>.</p>
```

```
{http://www.eccnet.com/namespaces/ecc-namespace} report[1]/body[1]/section[2]/p[4]/ → (
/definition[3] ''data element dictionary'')
```

図 4: 実験結果

1. スキーマのある XML 文書は，その論理構造の最上位から順にアクセスしてきたのに対し，スキーマのない XML 文書では，論理構造の最下位にあるテキストノードにある文字列から構造を上に向かってアクセスする必要がある．さらに良く知られた名前空間に属する要素型や属性があればそれを手がかりとできるが，これらは，XML 文書中の中間ノードとして存在する．そのため，従来の構造化文書で用いられていた構造索引だけでは不十分で，中間ノードからのアクセスやテキストノードから構造を上にたどる逆経路索引が必要となる．我々は，従来の経路と逆経路を合わせ持つ NRP(Normal and Reverse Path) 経路索引を提案している [17, 18, 19]．効率的なアクセス法のために索引を利用する必要がある．
2. 本稿で想定した単純 XML 問合せ $(e/\theta/v)$ では，述語を満たすか否かの真偽値で評価するため，検索文字列の出現回数を考慮していない．また，文字列の比較は単なるパターンマッチのため，日付のような多様な書式表現のあるものにはうまく適用できない．これは，XML 文書についてデータ型の導入や問合せと問合せ結果の関連度を算出する枠組を必要としている．
3. 本稿では，構造に関する問合せを想定していないが，スキーマのない XML 文書リポジトリに対しては，内容の検索同様，構造検索も重要となる．そのためにも，各 XML 文書の構造に関する統計を管理し，構造の問合せを効率的に行なう方法が必要である．
4. 本稿では，問合せ条件がひとつの場合を考察した．問合せ条件が複数の述語の連言の場合の部分文書の特定へ拡張する必要がある．
5. 本稿では，文書内の参照関係や文書内外のリンクについての想定をしていない．これらを考慮した部分文書の特定法に拡張する必要がある．
6. 抽出した部分文書の再構成の方法
   問合せに利用した要素型は共通に持つ構造であることを生かした仮想文書の作成の方法も今後の課題である．

参考文献

[1] Ben Y. Zhao, and Anthony D. Joseph. XSet: A High Performance XML Search Engine. http://www.cs.berkeley.edu/~ravenben/xset, 2000.

[2] Dublin Core. Dublin core metadata. http://purl.oclc.org/metadata/dublin_core/.

[3] ISO. ISO 8879: 1986. *Information Processing - Text and Office System - Standard Generalized Markup Language (SGML)*, Oct. 15 1986.

[4] JIS X 4151:1992 文書記述言語 SGML (Standard Generalized Markup Language)，日本規格協会，1992.

[5] JIS X 4151:1998 文書記述言語 SGML (Standard Generalized Markup Language)(追補 1)，日本規格協会，1998.

[6] TR X 0008:1998 拡張可能なマーク付け言語 XML (eXtensible Markup Language)，日本規格協会，1998.

[7] Dongwook Shin, Hyuncheol Chang, and Honglan Jin. Bus: An effective indexing and retrieval

表 2: テキスト以外の各ノードが対応する文脈ノードの出現回数:XSLT

	出現ノード回数	文脈ノードまでの経路式{名前空間 URI}
0	100 回 (3.17%)	/{}spec
1	16 回 (0.50%)	/{}spec/{}header/{}status/{}p
2	254 回 (8.05%)	/{}spec/{}body/{}div1
3	148 回 (4.69%)	/{}spec/{}body/{}div1/{}p
4	2 回 (0.06%)	/{}spec/{}body/{}div1/{}p/{}termdef
5	657 回 (20.83%)	/{}spec/{}body/{}div1/{}div2
6	652 回 (20.67%)	/{}spec/{}body/{}div1/{}div2/{}p
7	27 回 (0.85%)	/{}spec/{}body/{}div1/{}div2/{}note
8	44 回 (1.39%)	/{}spec/{}body/{}div1/{}div2/{http://www.w3.org/1999/XSL/Spec/ElementSyntax}element-syntax
9	226 回 (7.16%)	/{}spec/{}body/{}div1/{}div2/{}ulist/{}item
10	177 回 (5.61%)	/{}spec/{}body/{}div1/{}div2/{}div3
11	251 回 (7.96%)	/{}spec/{}body/{}div1/{}div2/{}div3/{}p
12	11 回 (0.34%)	/{}spec/{}body/{}div1/{}div2/{}div3/{}p/{}termdef
13	53 回 (1.68%)	/{}spec/{}body/{}div1/{}div2/{}div3/{}ulist/{}item
14	14 回 (0.44%)	/{}spec/{}body/{}div1/{}div2/{}div3/{}note
15	4 回 (0.12%)	/{}spec/{}body/{}div1/{}div2/{}slist/{}sitem
16	80 回 (2.53%)	/{}spec/{}body/{}div1/{}ulist/{}item
17	18 回 (0.57%)	/{}spec/{}body/{}div1/{}div2/{}scrap/{}prodgroup/{}prod
18	32 回 (1.01%)	/{}spec/{}body/{}div1/{}div2/{}scrap/{}prodgroup/{}prod/{}rhs
19	7 回 (0.22%)	/{}spec/{}body/{}div1/{}div2/{}note/{}p
20	10 回 (0.31%)	/{}spec/{}body/{}div1/{}div2/{}olist/{}item
21	8 回 (0.25%)	/{}spec/{}body/{}div1/{}div2/{}olist/{}item/{}p
22	25 回 (0.79%)	/{}spec/{}body/{}div1/{}div2/{}olist/{}item/{}ulist/{}item
23	3 回 (0.09%)	/{}spec/{}body/{}div1/{}div2/{}eg
24	8 回 (0.25%)	/{}spec/{}body/{}div1/{}div2/{}div3 /{http://www.w3.org/1999/XSL/Spec/ElementSyntax}element-syntax /{http://www.w3.org/1999/XSL/Spec/ElementSyntax}attribute
25	1 回 (0.03%)	/{}spec/{}body/{}div1/{}div2/{}div3/{}note/{}p
26	8 回 (0.25%)	/{}spec/{}body/{}div1/{}div2/{}note/{}ulist/{}item
27	10 回 (0.31%)	/{}spec/{}body/{}div1/{}div2 /{http://www.w3.org/1999/XSL/Spec/ElementSyntax}element-syntax/ {http://www.w3.org/1999/XSL/Spec/ElementSyntax}attribute
28	8 回 (0.25%)	/{}spec/{}body/{}div1 /{http://www.w3.org/1999/XSL/Spec/ElementSyntax}element-syntax /{http://www.w3.org/1999/XSL/Spec/ElementSyntax}attribute
29	2 回 (0.06%)	/{}spec/{}body/{}div1/{}ulist/{}item/{}p
30	12 回 (0.38%)	/{}spec/{}body/{}div1/{}ulist/{}item/{}ulist/{}item
31	4 回 (0.12%)	/{}spec/{}body/{}div1/{}note
32	14 回 (0.44%)	/{}spec/{}body/{}div1/{http://www.w3.org/1999/XSL/Spec/ElementSyntax}element-syntax
33	2 回 (0.06%)	/{}spec/{}body/{}div1/{}div2/{}ulist/{}item/{}p
34	2 回 (0.06%)	/{}spec/{}body/{}div1/{}p/{}code
35	4 回 (0.12%)	/{}spec/{}back/{}div1
36	22 回 (0.69%)	/{}spec/{}back/{}div1/{}div2
37	43 回 (1.36%)	/{}spec/{}back/{}div1/{}div2/{}blist/{}bibl
38	60 回 (1.90%)	/{}spec/{}back/{}inform-div1
39	10 回 (0.31%)	/{}spec/{}back/{}inform-div1/{}p
40	25 回 (0.79%)	/{}spec/{}back/{}inform-div1/{}div2
41	65 回 (2.06%)	/{}spec/{}back/{}inform-div1/{}orglist/{}member
42	34 回 (1.07%)	/{}spec/{}back/{}inform-div1/{}ulist/{}item
	3153 回	

scheme in structured documents. In *Proc. of Digital Libraries '98*, pp. 235–243, Pittsburgh, 1998.

[8] Dongwook Shin. XRS: XML Retrieval System. http://www.dlb2.nlm.nih.gov/~dwshin/xrs.htnml, 2000.

[9] H. Kinutani, et al,. On Contextual Queries for XML Documents using Na mespaces. In *Proc. of International Symposium on Digital Libraries 1999 (ISDL99)*,pp.151–154, Sep. 1999.

[10] Roy Goldman and Jennifer Widom. DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. In *Proc. of VLDB*, pp.436–445, 1997.

[11] Roy Goldman and Jenifer Widom. Approximate DataGuides. Workshop on Query Processing for Semistructured Data and Non-Standard Data Formats, 1999. http://www-db.stanford.edu/pub/papers/adg.ps, 1999.

[12] World Wide Web Consortium. Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/1998/REC-xml-19980210, February 1998.

[13] World Wide Web Consortium. QL'98 - The Query Languages Workshop. http://www.w3.org/TandS/QL/QL98/, December 1998.

[14] World Wide Web Consortium. Web naming and addressing overview (URIs, URLs, ...), June 1998. http://www.w3.org/Addressing/Addressing.html.

[15] World Wide Web Consortium. Namespaces in XML. http://www.w3.org/TR/1999/REC-xml-names-19990114/, January 1999.

[16] World Wide Web Consortium. XML Path Language (XPath) version 1.0. http://www.w3.org/TR/1999/REC-xpath-19991116, November 1999.

[17] 山本陽平, 吉川正俊 , 植村俊亮. 名前空間を含む XML 文書に適した索引. 日本ソフトウェア科学会第 2 回インターネットテクノロジーワークショップ (WIT'99), pp.107–114, 1999.

[18] Y. Yamamoto, M. Yoshikawa and S. Uemura. On Indices for XML Documents with Namespaces. In *Markup Technologies'99*, pp.235–243, Dec. 1999.

[19] 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. 第 3 回 XML 開発者の日. http://www.xml.gr.jp/xmldevday3.html, 2000.