DBSJ Journal, Vol.7, No.3
December 2008

構造型 P2P ネットワークにおけるキーワードを含む XPath による XML 文書検索

Searching XML Documents using XPath Full-Text in Structured P2P Networks

李 暁晨♥　天笠 俊之♦　北川 博之♠

Xiaochen LI　Toshiyuki AMAGASA
Hiroyuki KITAGAWA

近年，P2P 環境における XML 文書の分散管理が注目され，様々な研究や応用が進められている．我々の研究グループはこれまで分散ハッシュ表を利用した XML 文書の検索手法として，XPath を利用する手法と，転置リストを利用したキーワード検索手法を提案してきた．本論文では，これらの手法を結合し，XML 文書の構造情報とテキスト情報の双方を考慮し，キーワードを含む XPath による XML 文書検索手法を提案する．

Recently, Peer-to-Peer systems and XML are attracting increasing attention. There have been several attempts to deal with XML documents in P2P networks. Our research group has been proposed two methods for XML document processing in DHT-based P2P networks. One uses XPath and the other uses inverted list to realize keyword search. Here, we combine the two proposed methods to realize XPath full-text search on XML document which considers both the structural information and the text information of XML.

1. はじめに

Peer-to-Peer (P2P) 技術は，オーバレイネットワークの基盤技術として，近年活発に研究されている．P2Pシステムとは，システム内のピア同士が対等な関係を持ちながら処理を行う分散システムのことである．P2Pシステムの一手法として，分散ハッシュテーブル（DHT）がある．DHTを用いることにより，検索対象のデータを効率的に検索することが可能となる．

一方, XMLはタグによって文書やデータにおける意味や構造を記述するためのメタ言語である．1998年にW3C勧告となってから現在までの間に急速に普及し，標準的なフォーマットとして，様々な分野に広く応用されている．こういったXMLデータをP2P環境で利用することも一般的に行われるようになっているので, P2P環境でXMLデータの格納と検索を効率的に行うための技術基盤がますます重要になっている．

一般に，XML文書の問合せには，構造型問合せ，非構造型問合せ，半構造型問合せの三つのタイプがあると考えられる．構造型問合せでは，利用者がXML問合せ言語（XQery [7], XPath [10]）を用い検索要求を問合せ式に記述する．この方式は，利用者がXML問合せ言語の専門知識を有し，文書構造を事前に把握していることが前提になる．我々の研究グループは分散ハッシュ表に基づきXPathによるXML文書の効率的な格納・検索を可能にする手法[4]を提案している．

しかし，実際のP2P環境では異なる利用者がさまざまなXML スキーマを用いる可能性があり，全てのXML文書構造を把握することが現実的ではないため，非構造型問合せのニーズが高まっている．そこで，我々は転置リストを利用したXML文書のキーワード格納・検索手法を提案してきた[5].

一方，両者の中間的な問合せ方法として，半構造型問合せも考えることができる．この場合，利用者は検索対象のXML文書の部分的な構造とキーワードの双方を問合せ式に記述する．このような問合せは，利用者が部分的な構造しか知らない時に使用される．最近新たなXML問合せ言語として公表されたXPath Full-Text [9]では，ftcontainsなる演算子を用いてXPathによるXML文書のキーワード検索を表現することができる．

そこで本論文では，XPath Full-Textが想定するような，パスとキーワードを組み合わせたXMLキーワード検索をP2Pネットワーク上で実現するため，これまでに提案してきた二つの提案した手法を統合し，キーワードを含むXPathによるXML 文書検索手法を提案する．具体的には，XMLの構造情報とテキスト値の双方を検索するため，問合せ式を二つの部分（パス部分とキーワード部分）に分けて，拡張したKW-DHTによって処理を行う．

本論文の構成は以下の通りである．まず第2章で関連研究について述べ，第3章で本研究に関する基本的事項を説明する．次に第4章で提案手法の概要を述べ，最後に第５章でまとめと今後の課題について述べる．

2. 関連研究

P2P環境におけるXML文書の格納，検索には，いくつかの手法が提案されている. Reynolds 等[2]はDHTを用いて転置リストを実現し，（XML ではない）文書の全文検索を行う手法を提案している．そこでは，AND 検索におけるコンテンツIDの転送時にブルームフィルタを用いてトラフィックを低減する手法を用いている．本研究は，この手法をXMLに拡張したものとして位置付けられる．

Abiteboul等は，DHTに基くXML検索手法として，KadoP [13] を提案している．KadoPにおいて，XMLは要素あるいはテキスト中のキーワードに基いて分解され, DHT上に格納される．検索処理としてはXPathがサポートされ，XPath問合せを構成する各要素名あるいはキーワードのリストがDHTによって検索され，それらはHolistic Twig Joinによって問合せ結果へと変換される．さらに，[14]では，DHT を構成するピア間のストレージにおける負荷分散のために，DHT上にB-treeに類似した構造を構築する手法を提案している. また, 問合せ結果の候補の効率的な絞り込みのために，ブルームフィルタに基づくフィルタリング手法も提案している．

しかし，これらの手法は，キーワードを用いたXML文書の半構造問合せを実現していない．また，転置リストの実現に

♥ 学生会員　筑波大学システム情報工学研究科博士前期課程　babala0851@kde.cs.tsukuba.ac.jp
♦ 正会員　筑波大学システム情報工学研究科／計算科学研究センター　amagasa@cs.tsukuba.ac.jp
♠ 正会員　筑波大学システム情報工学研究科／計算科学研究センター　kitagawa@cs.tsukuba.ac.jp

ピアID を使っているため，万が一ピアがオフラインの場合は，検索処理を行うことができないといった点で本研究と異なる．

3. 基本的事項

本論に入る前に，提案手法の前提となる基本的事項について伸べる．

3. 1 分散ハッシュ表(DHT)

分散ハッシュ表(DHT; Distributed Hash Table)は，構造型P2Pネットワークの一種である．DHTでは，システムで共通の一つのハッシュ関数を用意し，このハッシュ関数から得られる値を元に，オブジェクト（ピア，データ）をオーバレイネットワーク上に配置する．ピアがDHTシステムに参加する際，担当すべきハッシュ値の範囲が与えられ，ピアで分割して担当することになる．ある検索キーで登録されたオブジェクトを検索する場合は，その検索キーのハッシュ値を計算し，このハッシュ値を担当するピアにオブジェクトの所在を尋ねる．DHTには，代表的な方式として，Chord[3], Pastry[6]とCAN[8]などがある．以下では本研究で用いるChordについて紹介する．

Chordでは，2^N 個のオブジェクトからなる円構造のハッシュ空間（ID サークル）を用いる． NをIDのビット数とする．オブジェクトはハッシュ関数でIDサークル上にマップされる．IDがiのピアは，担当するデータのID，フィンガーテーブル，前後に位置するピアのID情報(predecessor, successor) を保持している．またピアiが担当するデータは，前ピア(predecessor)からiまでのハッシュ値を持つオブジェクトである．フィンガーテーブルには，N個の他ピアへのリンクがあり，リンク先は$i+2^k$ (k = 0,1,2,...,N-1)位置を担当するピアである．このため，オブジェクトの検索に要するホップ数は$O(\log_2 N)$である．

図1にChordの構造とオブジェクト検索の例を示す．ピア12がキー6のデータを検索したいとする．ピア12のフィンガーテーブル内で最もキー6に近いのはピア5であるため，まずピア5にアクセスする．これを繰り返し行うことで，キー6の情報を保持するピアへアクセスする．この例では，キー6を保持しているのがピア8であるため，最終的にピア12はピア8にアクセスし，キー6に対応するデータを取得する．

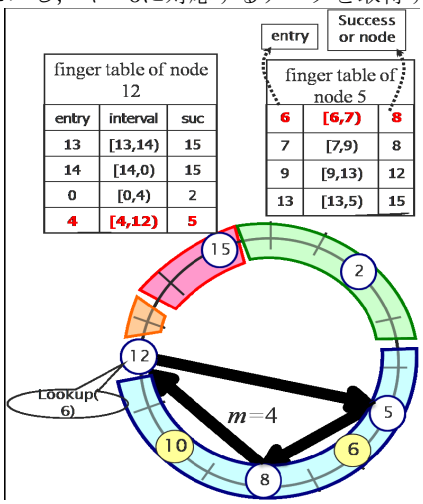


図 1　Chord の ID サークル

Fig.1　An example of Chord.

3.2 XPath Full-Text

XPath Full-Textは，XML問合せ言語XPath[10](XML Path Language)に全文検索機能を追加した問合せ言語である．XPath Full-Textは2008年の5月にW3C勧告として公表された, XML文書の構造を考慮した全文検索をするための標準的な方法を提供している．例えば，従来のXPathで提供されていた述語であるcontainsに対応する述語として，全文検索のマッチングを示すftcontainsを利用することができる．

//book[@no = "1"]/title ftcontains (”usability” ftand ”testing”)

は，号数が1の書籍において，タイトルが“usability”と“testing”に関連するかどうかを真偽値によって返す．本研究では，このftcontainsに着目し，DHT上での実現方法を検討する．

4. 提案手法

ここで, キーワードを含むXPathによるXML文書検索を実現するため，我々が文献[5]で提案してきたKW-DHTを改善したXML データの格納・検索手法を説明する．

4. 1 提案手法の概要

提案手法では，DHTを利用したXML問合せを実現するために，まずXMLデータを分解しDHT に格納する．XML データの構造に関する部分は[4]に従い, キーワード検索に関わるXMLテキストの部分の格納方法は[5]に従う．

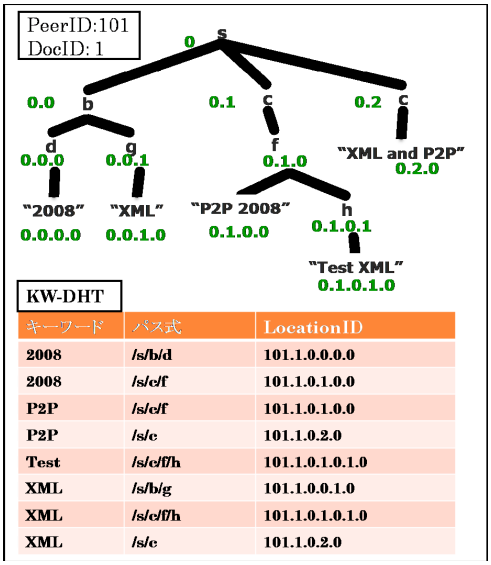
問合せ処理は以下のように行う: 1) 問合せのパス部分が述語を含まない場合，例えば，s/c ftcontains ("2008" ftand "XML" ftand "P2P")のような問合せの場合は，従来手法[5]を若干拡張することによって処理可能である．2) 問合せのパス部分が述語を含む場合, 例えば, s/c[f] ftcontains ("2008" ftand "XML" ftand "P2P")のような問合せの場合，問合せをパス部分s/c[f]とキーワード部分s/c ftcontains ("2008" ftand "XML" ftand "P2P")に分けて処理を行う．基本的な戦略としては，キーワード部分は1) と同様の処理を行う．パス部分については，[4]によって該当するノードの候補を取得し，最後に両者をマッチングすることによって最終的な解を得る．

以下，まずはDHT を利用した転置インデックスの実現方法から説明する．

4. 2 パス式を導入した KW-DHT

KW-DHTとは，[5]で我々が提案した，DHTを利用したXMLのための転置インデックスの構築方法である. 本研究では，パス式を含むキーワード検索を扱うため，KW-DHTを拡張する必要がある．具体的には，従来はある単語に対応するエントリの値が，その単語を含む要素名とそのノードのLocationID（ピアID + 文書ID + XMLノードID）だったのに対し，パス式によるフィルタリングを可能にするために，単語の所在パスをKW-DHTの転置リストに格納するようにした点である．XMLデータの各葉ノード（テキストノード）と属性ノードについて，テキスト値または属性値の中の各単語をハッシュキー，（単語所在パス, LocationID）の転置リストを値とする．単語所在パスはXPath Full-Textクエリのパス部分を処理するために利用する. LocationIDは単語が出現する場所の位置情報を保持するIDであり，ピアID, 文書ID, Dewey Numberの連結で構成される．

図2 に実際のXMLデータをKW-DHTに格納する例を示す. 例えば，ピアIDが101であるノードは文書番号が 1 である文書を保持しているとする．



KW-DHT

キーワード	パス式	LocationID
2008	/s/b/d	101.1.0.0.0.0
2008	/s/c/f	101.1.0.1.0.0
P2P	/s/c/f	101.1.0.1.0.0
P2P	/s/c	101.1.0.2.0
Test	/s/c/f/h	101.1.0.1.0.1.0
XML	/s/b/g	101.1.0.0.1.0
XML	/s/c/f/h	101.1.0.1.0.1.0
XML	/s/c	101.1.0.2.0

図 2 KW-DHT の例

Fig. 2 An example of KW-DHT.

4. 3 パス部分が単純パスである問合せ

次に，問合せ処理について述べる．上で説明したように，問合せのパス部分が述語を含まない単純パスかそうでないかによって処理手順が異なる．パス式のマッチングによるフィルタリングを行うかどうかの違い以外は，[5]と同一であるので，ここでは概要のみ説明する．

4. 3. 1 問合せ式とその答え

与えられたクエリは$q=|e_1|e_2|\ldots|e_n$ ftcontain k_1 ftand … k_mであるとする．ここで，e_i は要素名，k_j はキーワード，|はロケーションステップの分離記号で，/または//であるとする．

XMLは木構造を有するため，キーワード検索の際には，キーワードにマッチする部分文書を決定する必要がある．与えられたキーワードにマッチする部分文書を決定するために，いくつかの概念を導入する．最小共通祖先LCA (Lowest Common Ancestors) とは，木においていくつかのノードの共通の祖先の内，最も葉に近いノードである．LCAの中で与えられたキーワードを全て含む最小のLCAに対応するSLCA (Smallest LCA) を紹介する．

- LCAの内，与えられたノードを全てその子孫ノードに含み
- その子孫に同様に全てのノードを含むノードを持たない最小のノードである

すなわち，与えられたパス式で配下にあるSLCAを検索すれば良い．

4. 3. 2 キーワードによる DHT の検索

まず，クエリに含まれているキーワードk_j をKW-DHTによって検索する．原理的には，各キーワードに対応するピアにおいて，そのキーワードが出現する場所の情報が得られるので，それらをまとめた上でSLCAを計算すれば良い．得られたキーワードの転置リストにおいて，クエリのパス式と単語所在パス式がマッチするかどうかを確認し，マッチした転置リストのLocationIDを全て抽出する．

例えば，クエリはs/c ftcontains (“2008” ftand “XML” ftand “P2P”) であるとする．図2から，キーワード2008 の転置リスト: (/s/b/d, 101.1.0.0.0.0), (/s/c/f, 101.1.0.1.0.0); XMLの転置リスト: (/s/b/g, 101.1.0.0.1.0), (/s/c/f/h, 101.1.0.1.0.1.0), (/s/c, 101.1.0.2.0); P2Pの転置リスト: (/s/c/f, 101.1.0.1.0.0), (/s/c, 101.1.0.2.0)が得られる．その中で，クエリのパス部分s/c とマッチする転置リストのLocationIDを抽出し，2008のLocationIDリスト(101.1.0.1.0.0)，XMLのLocationIDリスト(101.1.0.1.0.1.0, 101.1.0.2.0) とP2PのLocationIDリスト(101.1.0.1.0.0, 101.1.0.2.0) が得られる．得られたLocationIDを用い，P2Pネットワーク上でキーワードのSLCAを計算する．

4. 3. 3 LocationID による SLCA の計算方法

P2Pネットワーク上では，キーワードの転置リストが分散配置されている．与えられたキーワード集合を含む部分文書からSLCAを計算するには，LocationIDをピア間でやり取りする必要がある．図3に例を示す．検索手順について，ピア間での通信量を最小限にするため，問合せが各検索キーワードのLocationIDリストのサイズの小さいものから順に処理を行う．まず，最初のピアにおいて，KW-DHTによる検索キーワードの転置リスト中のLocationIDリストを抽出し，次のピアに送信する．次のピアは，送られたLocationIDリストと自分が保持するキーワード集合にマッチして，LCAを探す．得られたLCAから親子もしくは祖先子孫関係にない最小部分木（SLCA）を計算する．これを全てのキーワードについて行うことで，全キーワードを含むSLCAを計算することができる．

二つLocationIDリストのマッチについて，例えば，検索キーワードがXMLとP2Pであるとする．XMLのLocationIDリスト: (101.1.0.0.0, 101.1.0.1.2.0.0, 101.1.0.1.1.1.0) とP2PのLocationIDリスト: (101.1.0.1.1.2.0, 101.1.0.1.2.1.0) により，三つの最大共通部分（LCA）（101.1.0, 101.1.0.1.2, 101.1.0.1.1）が得られる．この中で，100.1.0が他の二つLocationIDに含まれているため，最小部分木とならない．101.1.0.1.2と101.1.0.1.1が互いを含んでいないため，最小部分木となる．SLCAの定義により，この二つLocationIDはキーワードXMLとP2PのSLCAであることが分かる．計算されたSLCAをまた次のピアに伝送する．最終的に計算されたSLCAが検索を行ったピアに返される．

しかし，複数のキーワードによるSLCAの算出を行う際，転置リストの数が多くなるとLocationIDを送信するためのトラフィックが大量に発生してしまうという問題があるため，大規模なデータに対応できない恐れがある．

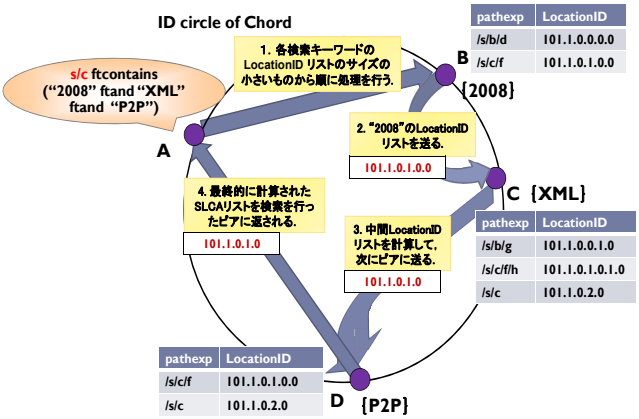


図 3 キーワードを利用した検索処理の例

Fig.3 An example of keyword search based on KW-DHT.

4. 3. 4 ブルームフィルタによる通信速度の改善

上で指摘した問題を解決するため，ブルームフィルタを利用する通信速度の改善手法について述べる．まず，ブルームフィルタの概要を紹介し，続いて適用手法にXMLデータへどのように適用するかについて述べる．

ブルームフィルタ

ブルームフィルタとは，複数の要素から構成される集合の中に，ある要素が含まれているかどうかを判定するアルゴリズムである．巨大なビット列に対し，コンテンツを表すキーワードを複数のハッシュ関数に通し，得られた値と等しい位置のビットをそれぞれ立てることでキーワードを表現する．ブルームフィルタを用いる利点は：

（１） キーワードはビットデータとして格納されているため，ノード間で情報を交換する際は少ない通信負荷で済む．

（２） コンテンツ有無の確認がビット演算で済むなど，処理に関する負荷が全体的に低いことが挙げられる．

例えば，二つのハッシュ関数h1, h2が定義されており，m=11ビットのビット配列を用いる．P2Pネットワーク上で，ピアAが持っているデータ集合は｛a,b,c｝，ピアBが持っているデータ集合は｛b,c,e,f｝とすると，AとBのデータ集合の論理積を求める．まず，ピアAは自分が持っているデータをハッシュ関数に通して，得られたハッシュ値のブルームフィルタビット列を作成して，ピアBに自分のデータ集合の代わりに，このビット列を伝送する．ピアBは受け取ったビット列と自分が持っているデータのハッシュ値に共通で含まれているデータ｛b,c｝を抽出する．抽出されたデータはAとBのデータ集合の論理積である．

ブルームフィルタによる転置リストのマッチング

ピア間での通信速度の改善についてブルームフィルタをXMLデータへどのように適用するかについて述べる．基本的には，二つ問題点がある．

（１） SLCAを計算するため，ブルームフィルタのどのような情報を登録すれば良いか．

（２）ブルームフィルタでは，ハッシュ関数を用いるため，ハッシュ関数による誤検出（False Positive）がある

ブルームフィルタの構成

まず，問題点1についての解決手法を説明する．XMLデータは木構造を有するため，ノード間の先祖・子孫関係を考慮してSLCAを算出しなければならない．このため，各LocationIDだけではなく，それらの全ての先祖ノードをブルームフィルタに格納する．例えば，LocationIDが101.2.0.1.1.2 とすると，0.1.1.2 による親を抽出して，ブルームフィルタに集合（101.2.0, 101.2.0.1, 101.2.0.1.1, 101.2.0.1.1.2）を入れる． ブルームフィルタを用いるSLCAを計算する際，ピアAは自分が持っている検索キーワードのLocationIDリストのハッシュ値を計算する．続いては，受け取ったブルームフィルタビット列によって，計算されたハッシュ値についての該当部分がすべて１かどうかをチェックして，ビット列に共通で含まれているLocationIDを得る．得られたLocationIDから互いに含まれていないLocationIDを抽出する．SLCAの定義により，これらのLocationIDはピアA が担当しているキーワードと受け取ったブルームフィルタが担当しているキーワードのSLCAである．最後に，ピアAが計算したSLCAをまた次のピアに伝送する．

誤検出の解消

続いて，ブルームフィルタの誤検出の解消について述べる．基本的な考え方は，以下の式が示す通りである．

A∩B (BUF(A)) = AUB

ここで，AとBは集合，F(A)は誤りを含む集合Aの部分集合である．仮にF(A)に偽陽性の要素が含まれていたとしても，再度Aとの論理積を取ってやれば，正しい答えが得られる．実際のキーワードを検索する場合は，まず，ブルームフィルタのみでSLCAを計算し，(誤りを含む)SLCAを求める．いったん求まったSLCAを今度は全てのピアに再度送信し，各ピアで論理積を計算することで，誤検出したLocationIDを排除する．

詳しく例で説明する．ピアAからピアZまでブルームフィルタを転送しながらSLCAの計算を行う．末端まで行き着いたら，次はピアZ から逆に，今度はブルームフィルタではなく候補解を直前のピアに手渡す．受け取ったピアは，自分が持っているデータと受け取ったLocationIDリストの論理積を計算する．これによって，偽陽性のエントリが取り除かれ，最終的に正しいSLCAを得ることができる．SLCAを算出した後は一般的にLocationIDの数が前より少なくなるため，例え候補解を伝送しても，LocationIDのサイズがかなり小さいことが期待できる．よって，全体の検索効率を上げることができる．

4. 4 述語を含むパス部分の問合せ処理

述語を含むパス部分の場合では，KW-DHTのみで処理できない．そこで，本研究では，[4]の手法を用い，述語を含むパス部分の問合せ処理を行う．

4. 4. 1 C-DHT と S-DHT による XML データの格納

まず，[4]を説明する．基本的には，XMLデータからテキスト情報を保持するノードと葉ノードに分解し，その要素名とテキスト値の組合せを，DHTに格納する．これをC-DHTと呼ぶ．具体的には，XMLデータのノードのうち，(1)テキストノードだけを含む要素ノード，(2)属性ノード，(3)テキストノードを含まない葉ノードなどをDHTに格納する．この際，要素名をハッシュキー，（ピアID，文書ID，パス式，DeweyOrder，テキスト）の組を値とする．

図4にXML データをC-DHTに格納した例を示す．例えば，cノードはテキスト値“XML and P2P”を持ち，それについてのパス式や順序ラベルを取得することができる．さらに，任意の部分XMLデータを検索，再構築するための手がかりとして，P2Pネットワーク上の全XMLデータからStrong DataGuide [11] を構築し，別途DHTに格納する．これをS-DHTと呼ぶ．具体的には，DGの各要素名をハッシュキー，（パス式，子ノード集合）の組を値として格納する．子ノード集合には，子ノードとして現れるノードを列挙するのに加えて，ノードの出現回数も記録しておく．図4に実際のDGをC-DHTとS-DHTに格納した例を示す．例えば，cノードは子要素とするテキストノードとfを持ち，それぞれ一つずつ存在することが分かる．

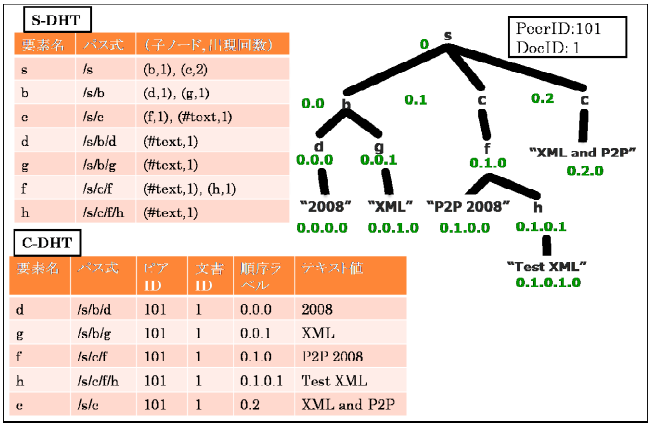


図 4 XML データから C-DHT，S-DHT を生成する
Fig. 4 Storing XML in C-DHT and S-DHT

4. 4. 2　C-DHT と S-DHT による XPath 検索

C-DHTとS-DHTによる検索処理の概要を述べる．まず，/,//だけを含む単純パス式の場合を説明する．与えられた検索式はq=|e1|e2|…|en であるとする．まず，qの末端要素名enを手がかりに，テキスト値を直接含む情報をC-DHTに探索する．続いてenを根とする部分XMLデータを辿るために，S-DHTの情報を手がかりにパス式を拡張し，再度C-DHTを参照する．具体的には，S-DHTからenの子要素である要素名を取得し，それをキーとして再度C-DHTを参照することで，テキストを得る．これを部分XMLデータの再構築が完了するまで繰り返す．

C-DHTとS-DHTを用い，述語を含むパス式の検索処理を説明する．ここでは簡単な例を用いて処理の概要を示す．例として，q=/a/b[c] なる問合せを考える．まずqから全てのパス式を抽出する（この例の場合は/a/b と/a/b/c になる）．得られたそれぞれのパス式に対して，上記で説明したC-DHTとS-DHTを用いる単純パスの問合せ処理を行い，解の候補集合を得る（例では，それぞれR1, R2とする）．次に，R1とR2がXMLデータ上で関連をもっているかどうか（先祖子孫関係にあるかどうか）を構造結合 (Structural Join) [12] によって調べる．具体的には，DeweyOrderの性質から，（パス式，ピアID，文書ID，DeweyOrder）について，ピアID，文書IDが等しく，DeweyOrderがプレフィックスを共有しているかどうかを調べる．構造結合の結果残ったデータが検索結果となる．

4. 4. 3　C-DHT と S-DHT による述語を含むキーワード検索

C-DHTとS-DHTによって述語を含むXPathを処理することができる．そこで，問合せののパス部分が述語を含む場合，例えば，s/c[f] ftcontains ("2008" ftand "XML" ftand "P2P") のような問合せの場合，問合せをパス部分s/c[f]とキーワード部分s/c ftcontains ("2008" ftand "XML" ftand "P2P")に分けて処理を行なうことにする．キーワード部分は上で述べたのと同様の処理を行えばよい．パス部分については，C-DHTとS-DHTにより平行して処理を行ない，両者の結果が得られた後，両者の間で実際にXMLデータ上で関連があるかどうかを構造結合によって判定する．図5に実際のP2PネットワークでのC-DHTとS-DHTによる述語を含むキーワード検索の例を示す．図5のように検索ピアAがパス部分とキーワード部分の二つの結果を持ち，構造結合を行う．

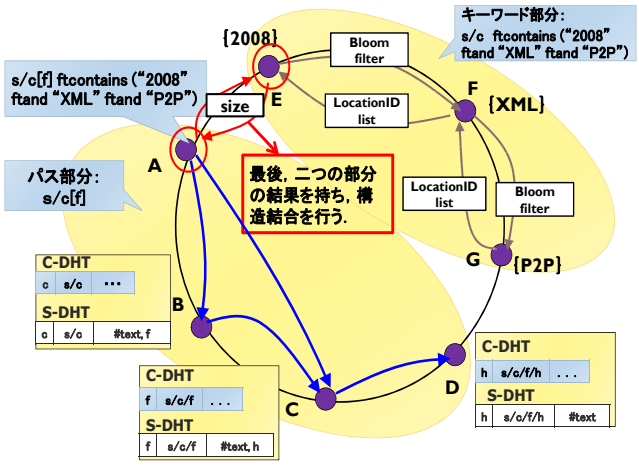


図 5 C-DHT と S-DHT による述語を含むキーワード検索処理の例
Fig. 5 Keyword search by C-DHT and S-DHT

5. まとめと今後の課題

本論文では，著者等がこれまでに提案してきたDHTにおけるXPathを利用するXML文書検索手法と転置リストを利用したキーワード検索手法を統合し，DHTにおいてキーワードを含むXPathによるXML文書検索手法を実現する手法を提案した．これにより，利用者はキーワードおよびXML文書の部分的な構造を利用した問合せをDHT上で処理することが可能となる．今後は，提案した手法を実装し，性能評価をおこなうよていである．また，XPath Full-Textにおけるftcontainsやftand以外の述語を含むような問合せにも対応したいと考えている．

[謝辞]

本研究の一部は科学研究費補助金若手研究（＃19700083），科学技術振興機構CREST「自律連合型基盤システムの構築」による．

[文献]

[1] Y. Xu and Y. Papakonstantinou. Efficient keyword search for smallest LCAs in XML databases. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pages 527-538, 2005.

[2] Reynolds, P., Vahdat, A. Efficient peer-to-peer keyword searching. Technical Report 2002, Duke University, CS Department, Feb. 2002.

[3] Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H. Chord: A scalable peer-to-peer lookup service for Internet applications. In Proc. ACM SIGCOMM'01, San Diego, CA, Aug. 2001.

[4] T. Amagasa, C. Wu, and H. Kitagawa. Retrieving arbitrary XML fragments from structured peer-to-peer networks. In Joint Conference of the 9th Asia-PacificWeb Conference and the 8th International Conference on Web-Age Information Management (APWeb/WAIM 2007), pages 317-328, 2007.

[5] Xiaochen Li, Toshiyuki Amagasa, and Hiroyuki Kitagawa, "Searching XML Documents by Keywords in Structured P2P Networks", 3rd International Workshop on XML Data Management Tools and Techniques (XANTEC 2008), Turin, Italy, Sept 1 - 5, 2008.
[6] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In In the 18th IFIP/ACM International Conference on Distributed Systems Platforms(Middleware), pages 329- 350, 2001.
[7] W3C. XQuery 1.0: An XML Query Language. http://www.w3.org/January 2007. Recommendation.
[8] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In ACM SIGCOMM 2001, pages 161-172, 2001.
[9] W3C. XQery and XPath Full-Text Language Version 1.0. http://www.w3.org/TR/xpath-full-text-10, May 2008. Recommendation.
[10] W3C. XML Path Language (XPath) Version 1.0. http://www.w3.org/TR/xpath.html, November 1999. Recommendation.
[11] R. Goldman and J. Widom, DataGuides:Enabling Query Formulation and Optimization in Semistructured Databases. Proc. V LDB, pp.436-445(1997).
[12] S. Al-Khalifa, H. Jagadish, N. Koudas, J. Patel, D. Srivastava and Y. Wu, Structural Joins: A Primitive for Efficient XML Query Pattern Matching. Proc. ICDE, 2002.
[13] S. Abiteboul, I. Manolescu, and N. Preda, Constructing and querying peer-to-peer warehouses of XML resources. in SWDB, 2004.
[14] Abiteboul, S. Manolescu, I. Polyzotis, N. Preda, N. Chong Sun. XML processing in DHT networks. Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference. pp.606-615, 7-12 April 2008

李 暁晨　Xiaochen LI
筑波大学大学院システム情報工学研究科博士前期課程修士 2 年在学中. 現在は P2P 環境における XML データの検索に関する研究に従事. 日本データベース学会学生会員.

天笠 俊之　Toshiyuki AMAGASA
筑波大学大学院システム情報工学研究科, 計算科学研究センター講師. データベースシステムの研究に従事. 情報処理学会, 電子情報通信学会, 日本データベース学会各会員.

北川 博之　Hiroyuki KITAGAWA
筑波大学大学院システム情報工学研究科, 計算科学研究センター教授. 理学博士（東京大学）. 異種情報源統合, データマイニング, 文書データベース, 情報検索などの研究に従事. 情報処理学会および電子情報通信学会フェロー. 日本データベース学会理事. 日本ソフトウェア科学会, ACM, IEEE CS 各会員. 著書に「データベースシステム」（昭晃堂）など.