Vol.2009-DD-73 No.7
2009/9/25

異種 XML データに対するファセット検索手法の提案

駒水孝裕[†1] 天笠俊之[†1,†2] 北川博之[†1,†2]

近年の情報技術の発展に伴い増加した膨大なデータを探索する際にファセット検索が用いられる．一方，XML データはデータ表現の標準フォーマットとして広く利用されつつある．しかしながら，XML データに対するファセット検索に関する研究はあまり行われていない．そこで本研究では異種 XML データに対するファセット検索手法を提案する．ファセット検索に必要な情報である検索対象の構造を表すクラス，クラスにおける属性，実際のデータから得られるオブジェクトと検索に用いるファセットの四つの概念を定義する．さらにこれらの定義に基づき，XML データに対するファセット検索の形式的な定義を与える．

A Faceted Navigation Scheme for Heterogeneous XML Data

Takahiro Komamizu,[†1] Toshiyuki Amagasa[†1,†2] and Hiroyuki Kitagawa[†1,†2]

With the rapid progress of information technology, huge amount of information resources have been produced by mankind, and faceted navigation therefore has attracted a great attention recently as a method for retrieving information from huge amounts of data. XML has been widely used as a standardized data format data representation. However, faceted navigation over XML data has not been well-studied. In our research, we propose a faceted navigation scheme over heterogeneous collection of XML data. In this paper, we define important information (classes, attributes, objects and facets) of faceted navigation. In addition, based on these definitions, we formally define the process of faceted navigation over XML.

1. はじめに

XML[1] の出現以来我々の身の回りにおいて構造化文書やデータファイルなどの用途で XML が使われている．Web が発達し情報爆発時代に入り，アプリケーション間でデータのやり取りをする際のフォーマットとして XML が利用されるようになったことも XML の普及に大きく影響している．その普及に伴い XML データのサイズも膨大になり，XML データを効率的に検索することがますます重要となりつつある．

従来，XML データの検索には XPath[2] や XQuery[3] などのパス指定に基づいた問合せや，情報検索技術をベースにしたキーワード検索が用いられてきた．しかし，これらを利用するためには問合せ言語の構文や，検索対象 XML データの構造を把握する必要がある．しかしながら膨大な XML データの構造を把握することは困難である．このため，検索作業を単に繰り返すのではなく，システムにより利用者の検索行動をサポートすることが重要である．

そこで本研究では，問合せ言語による検索，キーワード検索とは異なる新たな検索手法として探索的検索手法 (Exploratory search)[4] の手法であるファセット検索を適用する．ファセット検索は膨大なデータに対する効果的な検索を行う手法として注目を集めている．検索対象データはあらかじめファセットと呼ばれる独立したカテゴリにグループ化されている．検索処理においては，表示されているファセットとその値（キー）を選択し検索対象データを絞り込む，という操作を繰り返し行うことで検索を行う手法である．ファセット検索では各ファセットが独立，直行していることから，ファセットを選ぶ順番に影響を受けることなく検索が行えるという特徴がある．ファセット検索の例としては DBLP Bibliography[⋆1] や Flamenco Search[⋆2] がある．

ファセット検索を XML データに適用する場合，以下のような問題点が考えられる．

- XML データの半構造性により，構造化されている部分と構造のない部分が混在する可能性がある．このような場合でも検索対象データとファセットを定義する必要がある

†1 筑波大学大学院システム情報工学研究科
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba
†2 計算科学研究センター
Center of Computational Sciences

⋆1 DBLP Bibliography, http://www.informatik.unitrier.de/ ley/db/
⋆2 The Flamenco Search, http://flamenco.berkeley.edu/

- ファセットのキーが単純値ではなく部分 XML データとなる場合がある

上記の問題点に対し,[5] では XML におけるファセット検索フレームワーク *FoX* を提案している．しかし,[5] では検索対象やファセット，キーなどを明確に定義していなかった．そこで本稿では XML データに対するファセット検索におけるオブジェクト，ファセット，キーを形式的に定義する．

さらに,[5] では考慮していなかった異種 XML データに対する検索についても考慮する．XML データは，同じデータを別のスキーマを使って記述することがよくある．たとえば，文献情報を記録した DBLP の文献情報 XML と，SIGMOD Record の XML データでは，記述する内容は同じ文献情報であっても，利用している DTD が異なるため，そのままでは同様に扱うことができない．このような場合に同じデータでデータ構造の異なるデータが検索できることは重要である．本研究では XML データに対するファセット検索におけるオブジェクト等の定義を用い，異種 XML データに対するファセット検索を目指す．

本稿の構成は次の通りである．まず第 2 節で基本概念の説明をする．続く第 3 節で XML におけるファセット検索における検索対象オブジェクトやファセットとキーを定義し，この定義を基に検索処理の処理手順を紹介をする．第 4 節で提案手法のシステム概要を紹介し，第 5 節で関連研究の紹介をする．最後に第 6 節でまとめと今後の課題について述べる．

2. 準　　備

本節では基本概念として，まずファセット検索を紹介し，次に XML の概要を説明する．最後に DTD と DTD に基づいたスキーマ木を紹介する．

2.1 ファセット検索

あるデータに対して検索する際にはキーワード検索などの情報検索技術を用いた手法が用いられてきたが，これを行うためには検索する意図を明確に持たなければならない．これに対し探索的検索手法では，利用者が明確な意図を持たない場合であっても，一連のブラウジング操作を通じて，興味のあるオブジェクトを提示することができる．ファセット検索はこのような探索的検索手法の一つである．

ファセット検索では，検索対象データはファセットと呼ばれるいくつかの直行したカテゴリに分類される．ファセットはデータにおける重要な「側面」を表している．各ファセットはキーと呼ばれる対象データから抽出された値を保持しており，利用者はあるファセットに対してキーを選択することで検索対象データを絞り込む．ファセット検索はこの動作を繰り返し行う検索手法である．

タイトル	著者	ジャンル	発売年
書籍 1	著者 1	小説	2009
書籍 2	著者 1	伝記	2008
書籍 3	著者 2	小説	2009
書籍 4	著者 2	雑誌	2007
書籍 5	著者 3	小説	2008
書籍 6	著者 3	小説	2007
書籍 7	著者 3	伝記	2007

表 1　書籍データのメタデータ

タイトル	著者	ジャンル	発売年
書籍 1(1)	著者 1(2)	小説 (4)	2009(2)
書籍 2(1)	著者 2(2)	伝記 (2)	2008(2)
書籍 3(1)	著者 3(3)	雑誌 (1)	2007(3)
書籍 4(1)			
書籍 5(1)			
書籍 6(1)			
書籍 7(1)			

表 2　書籍データのファセットその一

タイトル	著者	ジャンル	発売年
書籍 1(1)	著者 1(1)	小説 (4)	2009(2)
書籍 3(1)	著者 2(1)		2008(1)
書籍 5(1)	著者 3(2)		2007(1)
書籍 6(1)			

表 3　書籍データのファセットその二

表 1 は書籍のメタデータを示した例である．これらのメタデータを用いファセット検索する際には，メタデータに示された属性をファセット，属性値をキーとして抽出しておくで検索できる．表 2 に表 1 のファセットとキーを示した．この場合のファセットはタイトル，著者，ジャンル，発売年，となり，キーはその値となる．キーの直後にある括弧はそのキーで検索できるオブジェクト（書籍）の数を示している．この表示されたファセットとキーに対して利用者はただファセットとキーを選択するだけで検索ができる．例としてファセット「ジャンル」からキー「小説」を選ぶ．この選択をもとにシステムが検索対象オブジェクト（書籍）を絞り込む（表 3）．利用者は新たに表示されたファセットとキーを再度選択し，絞り込む．以下同様の動作を繰り返し行い検索対象オブジェクトを徐々に絞り込んでいく．これがファセット検索である．

2.2 XML (Extensible Markup Language)

XML (Extensible Markup Language) はデータの構造や意味を記述するためのマークアップ言語で，タグでくくられる文字列を用いて地の文に意味や構造を付加するものである．このタグの文字列を利用者が自由に記述できることから，ソフトウェア間の通信や様々なデータを保存するためのデータ形式として広く利用されている．XML データの特徴としては木構造を有すること，半構造データであることが挙げられる．図 1 に簡単な例を示した．図 1 の例はある本についての XML であり，その情報として著者とタイトルが記述され

Vol.2009-DD-73 No.7
2009/9/25

```
<book>
  <author age="22">
    <first>Albert</first>
    <last>Holstein</last>
  </author>
  <title>A tale of milch</title>
</book>
```

図 1 XML データの例：本
Fig. 1 Book : An example of XML data

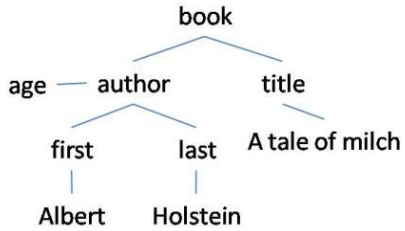

図 2 木構造
Fig. 2 Tree structure

ている．この XML データを木構造の形で表現したものが図 2 である．

定義 1 (XML) XML は (V, E, r) の三つ組で表現される木構造である．ここで，$V = \{v_1, v_2, ..., v_n\}$ をノード集合として表し，$E = \{(v_i, v_j) \mid v_i, v_j \in V \wedge i \neq j\}$ は枝集合を表す．要素 r は $r \in V$ なる根ノードである．

2.3 DTD とスキーマ木

本節では XML の構造を表現する DTD と DTD を基に作られるスキーマ木について説明する．

(1) DTD (Document Type Definition)

DTD (Document Type Definition) は SGML[⋆1] や XML の文書内で記述できる要素やその属性などを記述するスキーマ言語である．DTD を用いることでスキーマを厳密に定め，処理の正確性や安全性を高めることができる．図 3 に DTD の例を示した．DTD の各行ではある要素についての構造情報が記述される．「#PCDATA」は任意の文字列が入ること示す．例えば，図 3 の上から 2 行目では，book という要素について記述されており，ここでは book 要素の子要素は authors と title，data, section から構成される．また，同図の上から 5 行目は「⟨!ATTLIST」から始まり，ある要素の属性リストを示す．ここでは author 要素は age 属性を持ち，その値には任意の文字列が使用できることが示されている．本稿ではこの DTD を対象とする．

(2) スキーマ木

スキーマ木は DTD を木構造で表現したものである．図 4 に例を示す．子要素は要素の親要素からの枝と結ばれ，属性要素は要素の側部からの枝で結ばれる．スキーマ木において「#」はテキストを示しているが，属性はそれ自体に値を含んでいるために「#」が属性の子要素にはならない．

```
<!ELEMENT books (book*)>
<!ELEMENT book (authors, title, date, section*)>
<!ELEMENT authors (author+)>
<!ELEMENT author (first, middle?, last)>
<!ATTLIST author age CDATA #IMPLIED>
<!ELEMENT title (#PCDATA)>
<!ELEMENT date (year, month)>
<!ELEMENT section (title, number)>
<!ELEMENT first (#PCDATA)>
<!ELEMENT middle (#PCDATA)>
<!ELEMENT last (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT month (#PCDATA)>
<!ELEMENT number (#PCDATA)>
```

図 3 DTD の例：本
Fig. 3 Book : An example of DTD

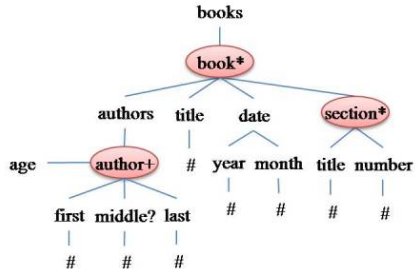

図 4 スキーマ木
Fig. 4 Schema tree

定義 2 (スキーマ木) スキーマ木 ST は $ST = (V_s, E_s, r_s)$ の三つ組で表現される木構造である．ここで $V_s = \{v_{s1}, v_{s2}, ..., v_{sn}\}$ はノード集合を表し，$E_s = \{(v_{si}, v_{sj}) \mid v_{si}, v_{sj} \in V_s \wedge i \neq j\}$ は枝集合を表す．要素 r_s を $r \in V_s$ なる根ノードである．なお，各ノード $v_s (\in V_s)$ は濃度 ($cardinality$) が与えられており，とりうる値は $\{*, +, ?, \emptyset\}$ のいずれかである．これを $v.card$ で参照する．

濃度において，「*」はその要素が 0 回か 1 回以上繰り返し出現すること，「+」はその要素が 1 回以上繰り返し出現することを意味し，「?」はその要素が 0 回か 1 回出現することを意味する．また，濃度が空の時はその要素が必ずただ 1 回だけ出現する．
DTD の各要素はスキーマ木におけるノードであり，ノード間の枝は親子関係を示す．また，DTD において任意の文字列を意味する「#PCDATA」はノード (#) となる．
DTD からスキーマ木を作成した例として図 4 を示した．図 4 のスキーマ木は図 3 を基に作成されている．ここでは便宜上，要素の濃度を要素名の直後に付加してある．

3. XML におけるファセット検索

前述したように，本研究でファセット検索の対象としているデータは XML データである．XML データにおけるファセット検索の問題点は，半構造性などの XML の特徴を考慮

⋆1 Standard Generalized Markup Language, http://www.w3.org/MarkUp/SGML/

したうえでファセットとキーの定義を与えなければならない点にある．そこで本節ではファセットとキーの定義を与える．そのために，まず XML データにおける検索対象クラスとその属性を定義する．次にクラスのオブジェクトにおけるファセットとキーを定義し，さらにこれらのファセットとキーを用いた検索処理過程を示す．

3.1 クラスと属性とオブジェクト

本節では XML データと DTD におけるクラス，クラス属性，オブジェクトを定義する．ここで言うクラスとは，DTD において検索対象となる要素を頂点とする DTD 木の部分木である．あるクラスから見て，子あるいは子孫に存在するテキストを含む要素を，そのクラスの属性と見なす．これにより，ファセット検索で対象とする XML 要素の範囲を明確化する．また，DTD 上で定義されるクラスとクラス属性に対して，実際に XML データ上で対応する部分 XML 木のことをオブジェクトと呼ぶ．以上のような定義により，ファセットは各クラスにおける属性，キーはオブジェクトから抽出される属性値として議論することが出来るようになる．

まずどのようにクラスを定義するかについて説明する．スキーマに注目すると，一つの XML データ中で複数回出現する要素，すなわち,「*」もしくは「+」が付いている要素は，複数回出現することが期待されているので，何らかの情報の単位を表現していると考えることができる．例えば，図 3,4 の「book」に着目すると,「book」が複数存在することを表しており，各々の「book」がデータ内で重要な情報の単位であることが推察できる．同様に,「author」,「section」もあるまとまった単位を表現していると期待される．この考え方はキーワード検索を行う際にも用いられている[6]．本稿ではこのような検索対象要素をクラスと呼ぶ．クラスの具体的な定義は以下のようになる．

定義 3 (クラス) スキーマ木 $ST = \{V_s, E_s, r_s\}$ において，クラス集合 C は以下のように定義される．

$$C = \{v \mid v \in V_s \wedge v.card \in \{'*','+'\}\}$$

次に，クラスの属性を定義する．あるクラス $c \in C$ について，基本的には，その子ないし子孫要素でテキスト値を直接含むノードをそのクラスの属性と考える．逆に，各テキスト値について，そこからたどることができる先祖ノードのうち，もっとも近いところにあるクラスが，その属性を保持すると考える．

定義 4 (属性) あるクラス $c \in C$ について，c のもつ属性 $c.A$ は以下のように定義される．

表 4 クラスリスト
Table 4 A list of classes

クラス	属性
book	title, year, month
author	first, middle, last, age
section	title, number

表 5 例：book オブジェクト集合
Table 5 A collection of book objects

ID	title	year	month
1	A tale of milch	2008	5
2	Milch stories	2009	1
3	Milch 物語	2009	2

$$c.A = \{v \mid v \text{ は子ノードに\#を持ち，} c \text{ までの経路に別のクラスを含まない }\}$$

ここで，$c.A$ のある属性を $c.a$，$c.a$ の要素名を要素名 $c.a.name$ として参照する．なお，c からの a までのパス式を相対パス $c.a.context$ として参照する．

表 4 に図 4 に示した例におけるクラスを示した．図 4 において「*」や「+」を持つ（赤丸で囲まれている）要素がクラスとなる．例として author 要素を見ると，濃度として「+」を持っているのでクラスである．また，その子要素 (first, middle, last) を見ると各々テキスト要素を保持しているので，author クラスの属性となる．さらに，author 要素は属性要素 age を持っている．この age 要素も author クラスの属性となる．

以上でクラスと属性の定義が与えられた．本研究では，あるクラスに対するオブジェクトとは，クラスに該当する XML データ中の部分木であると定義する．

定義 5 (オブジェクト) あるクラス $c \in C$ のオブジェクトとは，XML データ上でスキーマノード c で規定される部分 XML 木である．ここで，クラス C におけるオブジェクト集合を $c.O$ と定義する．

表 5 に表 4 に示した book クラスのオブジェクト集合を一例として示した．ここで,「ID」はオブジェクトを識別するための識別子である．

これまでに XML データにおけるクラス，属性，オブジェクトを定義した．これらの定義から一つの XML データからは一般に複数のクラスが抽出され得る．

3.2 ファセットとキー

次に，上で定義したクラス，属性，オブジェクトを基にファセットとキーを定義する．ある XML データに含まれるすべてのクラスが常にファセット検索の対象であるとは限らない．そこで，事前に検索対象となるクラス $C'(\subseteq C)$ を選んでおく．このとき，ファセットとは C' に含まれるすべての属性として定義される．

定義 6 (ファセット) 検索対象クラス C' のファセット集合 F は以下のように定義される．

$$F = \bigcup_{c \in C'} c.A$$

ファセットにおけるキーを考える際にまずクラス C を取得する．このクラス集合のオブジェクトを基にあるファセット f におけるキー集合を定義する．

定義 7 (キー) ファセット $f \in F$ についてキー集合 K_f は以下のように定義される．

$$K_f = \bigcup_{o \in c.Os.t. c \in C' \wedge f \in c.A} o.f.value$$

3.3 検索処理

本節では検索処理について説明する．ファセット検索を実現する上で必要な処理には以下が考えられる．

- 利用者がこれまでに選択したファセットとキーのペアを条件として，それに該当する全オブジェクトの検索．
- 現在選択されているオブジェクト集合に対して，あるファセットを持つオブジェクト数の計算．
- 現在選択されているオブジェクト集合に対して，あるファセットにおけるキーに該当するオブジェクト数の計算．

以下では，以上の各処理をどのように行うかについて順を追って説明する．

3.3.1 ファセットとキーを指定したオブジェクトの検索

ファセットとキーが複数指定された際のオブジェクトの検索処理について述べる．なお，ファセット検索においてファセットのないキーは選択できない，ここでは指定されているファセットとキーのペアの集合 $S = \{(f_1, k_1), (f_2, k_2), \ldots\}$ として扱う．ファセットとキーのペアの集合 S により絞り込まれるオブジェクト集合 O_S を以下のように計算できる．

$$O_S = \bigcup_{(f,k) \in S} O_{f,k}$$

3.3.2 キーに該当するオブジェクト数の計算

オブジェクト集合 $O_{f,k}$ をあるファセット f とそのキー k を保持するオブジェクト集合と定義する．その大きさを計算することで各キー毎に検索可能なオブジェクトの数を算出する．この計算の後にファセット毎に集約し，ファセットにおけるキー集合を取得する．また，各ファセット毎に検索可能オブジェクト数を算出するためには，各キーごとに計算したオブジェクト集合の和を計算すればよい．ファセット $f \in F$ のキー $k \in K_f$ を持つオブジェクト集合 $O_{f,k}$ を以下のように計算できる．

$$O_{f,k} = \{o \mid c \in C' \wedge f \in F \wedge o \in O_S \wedge o.f.value = k\}$$

3.3.3 ファセットを持つオブジェクト数の計算

次にこのキーごとのオブジェクト集合を用いて，あるファセット $f \in F$ の検索可能オブジェクト数付きキー集合 KC_f を計算する．ファセット $f \in F$ における検索可能オブジェクト数付きキー集合は以下のように計算できる．

$$KC_f = \bigcup_{k \in K_f} \{(k, \mid O_{f,k} \cap O_S \mid)\}$$

クラス集合 C' におけるオブジェクト検索可能数付きファセット集合は以下のように定義される．

$$FC_{C'} = \bigcup_{f \in F} \{(f, \mid \bigcup_{k \in K_f} O_{f,k} \cap O_S \mid)\}$$

これらを利用者に表示することで，利用者は各ファセットあるいは各キーがどの程度の絞込み能力を持っているかを判断できる．

4. ファセット検索システム概要

これまでに XML データを対象としたファセット検索を行うために，検索対象データに関する形式的な定義を与えた．本節ではそれらの定義に基づき得られた情報を基に検索するシステムの概要と実際に用いるデータの格納方法の説明をする．図 5 は XML データにおけるファセット検索システムの概要である．

4.1 システム構成

本節では図 5 の各機構 (Object Database, Facet Database, User Interface, Object Retriever, Result Generator, Facet Viewer, Result Viewer) の説明をする．

(1) XML Database は検索結果に必要なデータが格納されたデータベースで XML データベースを用いる．XML Database には検索対象となるすべての XML データが格納されているものとする．XML Database に対して検索処理を行う際には XQuery を用いる．

(2) Facet Database はファセットとキーを格納するデータベースで関係データベースを用いる．関係データベースを用いることでキー毎の検索可能オブジェクト数の計算等の集約演算を効率的に行うことができる．Facet Database として XML データベースを用いない理由としては，XML データベースには集約演算が存在せず，集約演算

のために複雑な問合せをしなければならないためである．また，XML データベースでは複雑な問合せには膨大な時間がかかってしまうことも関係データベースを用いる理由である．

(3) User Interface は利用者とのインタラクションを行う．利用者に対し，ファセットとキーを提示し，選択されたファセット，キーの内容を Object Retriever に伝える．また，これまでに選択してきたファセットとキーを表示し，利用者の検索をサポートする．

(4) Object Retriever は選択されたファセットとキーを基にオブジェクトの絞込み，キー毎，ファセット毎の検索可能オブジェクト数の計算を行う．Object Retriever の入力 S はファセットとキーのペアの集合である．入力 S について SQL を生成し，ファセットとキーの検索可能オブジェクト数対リストと結果オブジェクトを取得するためのパス式集合を獲得する．その各々を Facet Viewer と Result Generator に送信する．

(5) Result Generator は送信されてきたパス式集合を基に XQuery を生成し結果を取得する．その後その結果を Result Viewer に送る．

(6) Facet Viewer は送信されてきたファセットとキーの検索可能オブジェクト数つきリストを利用者の見やすい形に変換する．単純な例としては HTML のリストを用いてファセットの下にキーを表示することが考えられる．ここで変換されたファセットとキーのリストを User Interface に渡して，利用者に再度表示する．

(7) Result Viewer は基本的な機能は Facet Viewer と同じである．Facet Viewer との違いは入力が XML データで与えられるため，XSLT[7] を用いての変換ができることである．これはすなわち，ファセット検索システムを構築する人の表示したい形に変換できるということである．ここで変換された検索結果を User Interface に渡し，利用者に表示する．

4.2 データベーススキーマ

本節では図 5 のデータベースに格納するためのスキーマについて説明する．

まず，Facet Database におけるスキーマについて説明する．Facet Database への問合せには二種類ある．一つは検索結果 XML データ取得用のパス式を取得する問合せ，もう一つは絞り込まれたオブジェクトのファセットとキーの算出の問合せである．後者にはこれに加えて，各々が検索できるオブジェクトの数を算出しなければならない．これらを実現可能なデータベーススキーマを考察する．

第 3 節で述べたように XML データにおけるファセット検索で用いられる情報は，クラス，クラス属性，オブジェクト，ファセットの四つである．この四つの概念のスキーマを表 6 のように定義した．

図 5 システム概要
Fig. 5 A system architecture

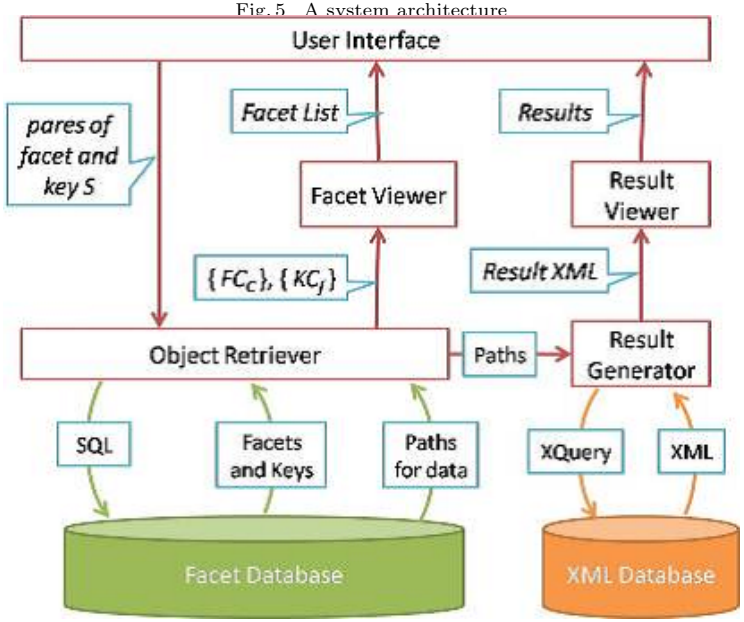

表 6 データベーススキーマ
Table 6 Data Schema

テーブル名	スキーマ
クラス	クラス（クラス ID，クラス名，パス式）
クラス属性	属性（クラス ID，属性名，パス式）
オブジェクト	オブジェクト（オブジェクト ID，クラス ID，パス式）
ファセット	ファセット名（キー，クラス ID，オブジェクト ID）

まずクラスのテーブル名は「クラス」で，各々のクラスに ID を付加する．クラス名は主キーには得ない．これは異種 XML データを扱う際に，複数の DTD において同一の要素名を持つものの異なる内容モデルを持つ要素が複数存在し得るからである．このような観点から同じ名前を持ったクラスでも識別できるように「クラス ID」を用いる．また，同スキー

マ内の「パス式」は検索結果 XML データを得るために使用するものである．

クラス属性テーブルはその名前を「属性」とし，所属しているクラスのクラス ID と属性名を主キーとする．これは同一クラス内には同じ名前の属性は存在せず，他のクラスに同一名の属性が存在するためである．また，このスキーマの「パス式」はクラスの要素からの相対パス式で表現される．

次にオブジェクトテーブルはテーブル名「オブジェクト」とし，すべてのオブジェクトを管理する．これまでのテーブルはデータ構造に関する情報を保持してきたが，このオブジェクトテーブルとこの後に紹介するファセットは実データが入る．このスキーマ中の「オブジェクト ID」はオブジェクトを一意に識別する主キーである．また,「クラス ID」を保持することでクラス毎にオブジェクトを見分けることができる．ここでの「パス式は」実データへのアクセスのためのパス式で，基本的にはクラスが持つパス式と同じだが，以下のようにデータの場所がより詳細に記されている．

$document(\text{"}mybook\text{"})//book/journal$[25]

このパス式は「mybook」という名前で保存された XML 文書に対して，その中の「book」要素の子要素の「journal」要素の 25 番目の要素を指示している．

最後にファセットテーブルを説明する．このスキーマはテーブル名をファセット名としてあり，ファセット毎にテーブルを構築する．このスキーマを構成する，キー，クラス ID，オブジェクト ID の組合せが主キーとなる．これは，あるオブジェクトのファセットを見たときにそのオブジェクトが複数のキーを持つこと可能性があることによる．例えば，買い物のカテゴリを考えると商品がオブジェクトでジャンルがファセットであると考えられる．このジャンルにおいて，複数のジャンルにまたがる商品が考えられる．具体的には，ダイエットに関する本を考えると，この本は「ダイエット・健康」というジャンルと「本」というジャンルの両方に属する．この場合ジャンルというファセットに二つのキー（ダイエット・健康，本）があることになる．このような場合が想定されるため，表 6 のようなスキーマになっている．

5. 関 連 研 究

本節では，ファセット検索を適用した関連研究を紹介する．

RDF を対象としたファセット検索[8)]

Oren らは[8)] においてファセット検索を RDF データに適用しており，本稿とは扱うデータが異なる．また，Oren らはファセット検索におけるファセットの順番の重要性について述べ，ファセットの順序付けに対する指標を提案し，その有用性を検証した．ファセットの順序付けに関しては[5)] で[8)] の手法を拡張したものを用いていたが，計算にかかるコストが高く速度面に問題があると結論付けていた．

大規模ファイルシステムにおけるファセット検索[9)]

Koren らは[9)] にて，ペタバイト級の大規模ファイルシステムの検索にファセット検索を用いることを提案していた．近年ペタバイト級のファイルストレージを用いることが増つつあることに着目し,[9)] ではファイルシステムで検索する際に利用者がはっきりとした意図で適切な問合せを表現することはできないと述べている．この点への改善策としてファセット検索を適用していた．このことから，大規模データの検索に対する手法としてファセット検索が有用であることがうかがえる．ファイルシステムにおける検索対象はすべてのファイルであるが，本稿では XML データを対象にしているために検索対象が異種のものになるという点で扱うデータが異なっている．

6. まとめと今後の課題

本稿では XML データにおけるファセット検索を行うために検索対象クラス，クラス属性，オブジェクト，ファセットの定義を行った．これらの定義に基づき，ファセット検索の過程で必要な処理を定式化した．また，システムの概要を示し処理の流れとその処理に適切なデータベーススキーマを提示した．

今後の課題としては，まず検証用システムを構築し，処理の流れやデータベーススキーマの妥当性を検証する．ここで妥当性が検証できれば，実際にファセット検索システムを構築し，本稿で述べた手法を実現する．そうでなければ，処理の流れに適したデータベーススキーマを定義する．また,[8)] でも述べられていたが，ファセットの並び順はファセットを選ぶ上で重要と考えられる．しかし,[5)] によると計算時間に問題があった．そこで本研究としては[5)] で利用されていた順序付け手法の定量的な評価を行い，その上で新たなファセットの順序付け手法を検討する．その後に，ファセット検索システムを構築する人が検索対象 XML データに対する詳しくなくともこのシステムを構築できるように，システム構築のステップと処理過程をフレームワーク化する．将来的には，このフレームワークを Web 上で公開し，XML 検索の更なる発展を目指す．

謝辞 本研究の一部は特定領域研究 (#21013004)，挑戦的萌芽研究 (#21650017)，クラウド環境におけるセキュアでトレーサブルな XML データ流通機構 (#21700093) による．

Vol.2009-DD-73 No.7
2009/9/25

参 考 文 献

1) W3C: XML1.0. http://www.w3.org/TR/REC-xml/.
2) W3C: XML Path Language. http://www.w3.org/TR/xpath/.
3) W3C: XML Query Language. http://www.w3.org/TR/xquery/.
4) W., R., White, Kules, B., M., S., Drucker and Schraefel, M.: Supporting Exploratory Search, *Introduction, CommuSupporting Exploratory Senications of the ACM*, Vol.49, No.4, pp.36–39 (2006).
5) 駒水孝裕，天笠俊之，北川博之：XML データに対するファセットナビゲーションのためのフレームワーク FoX の提案，*DEIM* (2009).
6) Liu, Z. and Chen, Y.: Identifying meaningful return information for XML keyword search, *SIGMOD Conference* (Chan, C.Y., Ooi, B.C. and Zhou, A., eds.), ACM, pp.329–340 (2007).
7) W3C: XSL Transformations. http://www.w3.org/TR/xslt.
8) Oren, E., Delbru, R. and Decker, S.: Extending Faceted Navigation for RDF Data, *International Semantic Web Conference* (Cruz, I. F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M. and Aroyo, L., eds.), Lecture Notes in Computer Science, Vol.4273, Springer, pp.559–572 (2006).
9) Koren, J., Leung, A., Zhang, Y., Maltzahn, C., Ames, S. and Miller, E.L.: Searching and navigating petabyte-scale file systems based on facets, *PDSW* (Gibson, G.A., ed.), ACM Press, pp.21–25 (2007).