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

駒水　孝裕† 　天笠　俊之†† 　北川　博之††

† 筑波大学第三学群情報学類
†† 筑波大学大学院システム情報工学研究科
†† 筑波大学計算科学研究センター
〒 305-8573 茨城県つくば市天王台 1-1-1
E-mail: †taka-coma@kde.cs.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.ac.jp

あらまし　大規模なデータ群に対する探索的検索手法としてファセット検索がある．通常，ファセット検索インターフェースはデータの特性を熟知した人により作成される．しかし XML (*Extensible Markup Language*) [1] においてはデータ構造が複雑であることや，考慮すべき属性の組合わせが膨大になるという問題がある．そこで本研究では，XML におけるファセット検索インターフェースのためのフレームワーク *FoX* を提案する．*FoX* では，まず検索対象となる XML データからカテゴリ（ファセット）となり得るノード名をリストアップし，インターフェース提供者に選択させ，そのファセットが持つすべての値（キー）を抽出する．これらのファセットとキーを基にファセット検索インターフェース [2] を構築する．その際にファセットの順序を評価するランキングも考慮し，より使いやすいインターフェースの構築を目指す．
キーワード　XML，ファセット検索，フレームワーク，DataGuide

A framework for a faceted-navigation on XML data

Takahiro KOMAMIZU†, Toshiyuki AMAGASA††, and Hiroyuki KITAGAWA††

† College of Information Sciences
†† Graduate School of Systems and Information Engineering
†† Center for Conputational Sciences
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan
E-mail: †taka-coma@kde.cs.tsukuba.ac.jp, ††{amagasa,kitagawa}@cs.tsukuba.ac.jp

Abstract　For navigating over a cloud of data, a faceted navigation as an exploratory searching method is very useful. In general, a faceted-navigation interface is designed by a veteran who knows the data very well. However, in the case of XML (*Extensible Markup Language*) data, there are some problems such as complicated data structure and a lot of combinations of attributes. In our research, we propose a framework *FoX* for a faceted-navigation on XML data. Specifically, at first, we list up all the node name and generate categories (called facets). Then, we show the list to the who wants to construct a faceted-navigation interface to let he/she to choose necessary ones. Having obtained input from the user, the system extracts values(called keys) of the chosen facets from original XML data. Based on these facets and keys, the framework *FoX* constructs the faceted-navigation interface. Additionally, we consider the order of facets for effective exploration by ranking them.
Key words　XML, faceted navigation, framework, DataGuide

1. は じ め に

近年 XML データ [3] はデータフォーマットの標準として，構造化文書やデータファイルなど多くの用途で利用される機会が増加している．このようにさまざまな用途に応用される XML データがネットワークを通じて流通している．そのため XML

図 1　Flamenco Search

表 1　本のメタデータ

No	タイトル	著者	ジャンル	発売年
1	Book1	Author1	漫画	2007
2	Book2	Author1	漫画	2008
3	Book3	Author2	語学	2007
4	Book4	Author3	語学	1996
5	Book5	Author3	雑誌	2007
6	Book6	Author2	雑誌	2009
7	Book7	Author1	語学	2007

表 2　ファセットとキー 1

タイトル	著者	ジャンル	発売年
Book1	Author1(3)	漫画 (2)	2007(4)
Book2	Author2(2)	語学 (3)	2008
Book3	Author3(2)	雑誌 (2)	1996
Book4			2009
Book5			
Book6			
Book7			

表 3　ファセットとキー 2

タイトル	著者	ジャンル	発売年
Book1	Author1(2)	漫画 (2)	2007
Book2			2008

形式の情報源から必要なデータを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 XQuery [4] や XPath [5] などの XML のパスに基づいた問合せや情報検索技術を用いたキーワード検索がなされてきた．しかし，これらを利用するためには問合せ言語の構文や XML データを解析する際に用いられる DOM または SAX についての知識や検索対象 XML データの構造を把握する必要がある．しかしながら，XML データの増大に伴い検索対象 XML データの構造のすべてを把握することは困難である．このため，個別の検索を単に繰り返すだけでなく，利用者の探索行動をサポートを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 (*Exploratory search*) [6] をサポートする手法である，ファセット検索を適用する．

近年, ファセット検索は膨大なデータを対象に効率的な検索を行う手法として注目を集めている．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ごとにグルーピングされている．ファセット検索とは，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，という動作を繰り返し行いデータの検索を行う手法である．ファセット検索は以下のような特徴を持つ．

- 現在の検索結果とそれに関連したファセットとキーを表示する．
- ファセットとその値を選択したときに結果が何もないようなファセットは除外される．
- ファセットの値にはその値によって検索できるデータ数も同時に表示する．

ファセット検索の例としては DBLP Bibliography [7] や Flamenco Search（図 1）[8]，mSpace などがある．

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

- XML データが半構造データであるために構造化されている部分とされていない部分が混在する可能性がある．このような場合に検索対象データとファセットを定義する必要がある．
- ファセットのキーが単純値だけでなく，部分 XML データとなる場合がある．

本研究では科学データや文書データなどの様々な XML データに対するファセット検索インターフェースを生成する際に，上記の問題点に対処するための処理を取り入れたフレームワーク *FoX* を提案する．

本稿の構成は以下のようである．まず次の第 2 節にて基本技術としてファセット検索を説明する．続く第 3 節では提案するフレームワーク *FoX* の動作内容である，ファセットの抽出，キーの抽出，ファセットのランキングについて紹介する．第 4 節では *FoX* を DBLP の論文データに適用した例を紹介する．第 5 節では第 4 節で用いた例を基にランキングの評価をする．第 6 節で既存の関連研究を紹介し，最後に第 7 節でまとめと今後の課題を述べる．

2. 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従来型手法が用いられてきたが，これにはデータに対する知識が必要である．これに対し探索的検索手法では前提知識なしに検索を行うことができる．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交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．各ファセットにはそれぞれを象徴する値（キー）を保持しており，利用者はそ

れを選択することで検索対象データを絞り込んでいく．この動作を繰り返し行うことで検索を進めていく．

より具体的に示すと次のようになる．表 1 には 7 つの本のメタデータが記されている．ただし表 1 の No の項目は便宜上付けたものなのでファセットとしては扱わない．このデータに対するファセットとキーを表 2 に示した．ファセットは「タイトル」,「著者」,「ジャンル」,「発売年」となっており，キーはそれ以下に並ぶ値となっている．尚，括弧の中に記載されている値はデータ中に出てきた個数である．ただし記載のないものは個数が 1 のものである．検索をする際にはまずファセットを選択する．ここでは「ジャンル」を選択する．次にその中からキーとして「漫画」を選択する．すると表 1 において「ジャンル」が「漫画」となっているのは No 1 と No 2 の本なのでそれらが検索結果として得られる．そしてそれに伴って検索対象におけるファセットとキーは表 3 のように変化する．ここからさらにファセット「発売年」，キー「2008」などと選択するとひとつのデータに絞られる．このようにして徐々に見つけたいものに絞り込んでいくのがファセット検索である．

3. XML のためのファセット検索フレームワーク *FoX*

3.1 フレームワークの概要

前章で述べたファセット検索を XML データで実現するするためにフレームワーク *FoX* の概要について述べる．*FoX* は「*A Framework For Faceted navigation on XML*」の略で，その動作は以下の五つに分けることができる．

（1） ファセットの抽出
（2） キーの抽出
（3） ファセットのランキング
（4） インターフェースの生成

各々の説明に入る前に全体の流れを説明する．まず，ファセット検索インターフェース提供者はユーザに検索させたいデータを XML データベースに格納し，その情報を *FoX* に登録する．*FoX* は与えられた情報を基に XML データの DataGuide [9] を作成する (1)．次に DataGuide をファセット検索インターフェース提供者に提示し，どのファセットをインターフェースで使用するかを選択する．この時点で使用するファセットが決定するので，それを基にキーの抽出を行う (2)．キーの抽出が終わると今度は使用するファセットを評価した指標のパラメータを設定し，有用な順番にランキングする (3)．これによって検索を効率的に行えるようにする．それらが終了するとファセット検索インターフェースを生成 (4) し終了となる．

3.2 ファセットの抽出

ファセットを抽出する際には DataGuide が使われる．図 2 に示したのは三つの XML データとその DataGuide である．一番目のデータのルートは「BOOK」なので DataGuide のルートを「BOOK」とし，その子要素を「BOOK」の下に追加する．テキストノードの値は見ないので「text」としてテキストノードを表現する．二番目のデータを見ると一番目にはなかった要素「MONTH」,「DAY」,「URL」があるのでそれらを追加する．三番目のデータについても同様に行い図 2 の右側の DataGuide が得られる．

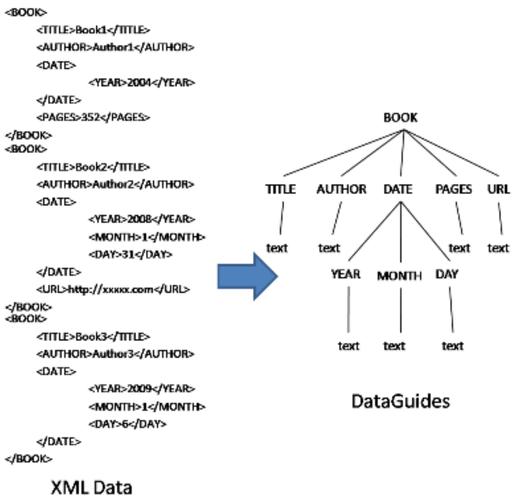

図 2　DataGuide

この得られた DataGuide から「BOOK」データのファセットの候補が取得できる．この場合では，ファセットの候補が「TITLE」,「AUTHOR」,「DATE」,「PAGES」,「URL」となり，その子要素がキーとなる．この中からインターフェース提供者が必要とするファセットを選択する．

ここでファセット「DATE」の子要素を見ると，三つの要素からなる部分 XML データとなっていることがわかる．このような場合に *FoX* ではインターフェース提供者にどのように表現するかを定義させる．例としては,「YEAR:MONTH:DAY」のようにコロン (:) で区切ることで一つのテキスト値として扱う．

このようにあらかじめ階層構造が存在するものを任意の形に成形しておくことで，階層構造の意味を損失することなく表現することができる．

3.3 キーの抽出

前節の方法で得られたファセットを基に検索対象 XML データからキーを抽出する．抽出方法は下記の様な XQuery をファセットごとに書くことで抽出を行う．

問合せ処理の流れとしては，まず FOR 節でファセットとなる要素をバインドする．続く LET 節で子要素をバインドし，RETURN 節で前節で定義された形式に基づいてデータの成形を行い，キーを抽出する．この作業を選択されたすべてのファセットに対して行うことで，必要なすべてのキーの抽出を行う．

```
FOR $facet at $pos in document(”book”)/BOOK/DATE
LET $node1 := $facet/YEAR
LET $node2 := $facet/MONTH
LET $node3 := $facet/DAY
RETURN
<data id=”$pos”>
{ data($node1)} :{ data($node2) } :{ data($node3)}
</data>
```

そして，抽出されたキーとファセットをデータベースに格納

ファセット	頻度
year	1
booktitle	1
mdate	1
title	1
key	1
url	0.9995
crossref	0.8403
pages	0.8297
author	0.7916
ee	0.6653
publisher	0.0124
cdrom	0.0072
isbn	0.0038
cite	0.0024
chapter	0.0002

表 4　ファセット頻度

ファセット	均衡度
ee	1
key	1
cdrom	1
chapter	1
url	0.9998
isbn	0.9967
title	0.991
publisher	0.9875
cite	0.9372
pages	0.7976
author	0.7226
booktitle	0.4648
crossref	0.4202
year	0.3732
mdate	0.2017

表 5　ファセット均衡度

ファセット	濃度
pages	0.8656
ee	0.8144
cite	0.8028
mdate	0.7644
booktitle	0.7599
crossref	0.7482
cdrom	0.7379
year	0.7337
publisher	0.7301
isbn	0.7296
chapter	0.7292
title	0.4054
url	0.3955
key	0.3949
author	0.2782

表 6　キー濃度

ファセット	単調性
year	0.99959
crossref	0.8542
booktitle	0.7583
mdate	0.7101
publisher	0.3712
isbn	0.221
cite	0.0559
author	0.0485
pages	0.0468
ee	0
title	0
key	0
url	0
cdrom	0
chapter	0

表 7　キー単調性

する．

3.4　ファセットのランキングのための指標

ファセット検索において利用者が検索時に見ることのできる情報はファセットとキーだけである．多数のファセットが定義されていても，スクリーンのサイズから，表示可能なファセットの数は限られており，利用者のユーザビリティを考えると，より使いやすく，利用されやすいファセットを画面内に表示させることが重要である．このことからファセット及びキーの順番は検索を効率化する上で重要な役割を果たすと言える．特に各ファセットは検索対象を絞り込む能力に差がある．例えば本のデータ群について二つのファセットを考える．まず一つ目は「ISBN」である.「ISBN」は国際標準図書番号のことで書籍を特定するための番号である．つまり，書籍と「ISBN」は一対一の関係にある．このようなファセットが検索対象を絞り込める力は非常に強いが，オブジェクト毎にとる値が異なるため，ファセットには向いていない．もう一つのファセット「言語」のキーは「和書」「洋書」の二つである．このようなファセットは検索対象を絞り込む力は弱いが，大きく二分できるので感覚的には使いやすい．しかし，二分割していく検索方法では検索したいものにたどり着くまでに何回も絞り込みを行わなければならないので，ISBN とは逆の意味でファセットには向かない．複数の書籍を著す著者のように，これら二つの中間の絞り込み能力を持つものを良いファセットであると考えられる．

良いファセットを判断する指標として以下の四つが考えられる [10]．この四つの指標はファセットの特徴をあらわしたものである．本節ではそれらの指標について説明する．次節にて四つの指標を用いたファセットの優劣を決めるランキングについて述べる．

- ファセット頻度
- ファセット均衡度
- キー濃度
- キー単調性

なお、ここではキーの順番については触れなかったが，キーはそれ自体が検索語なので単純に検索できるデータ数が多い順に並ぶことが望ましい．

3.4.1　ファセット頻度

ファセット頻度は検索対象データ全体でのファセットの出現頻度である．検索対象データで局所的に出現するファセットはファセットしては不適切である．なぜなら，局所的に出現するということはそれが含まれているごく一部のデータしか検索できないからである．ファセット頻度の式は以下のようになる．N は検索対象データ全体の数で n(facet) はファセットあたりの検索対象データの数である．

$$freq(f) = \frac{n(facet)}{N} \quad (1)$$

表 4 を見ると「year」,「booktitle」,「mdate」,「title」,「key」がすべてのデータに出現することがわかる．逆に「publisher」,「cdrom」,「isbn」,「cite」,「chapter」は低頻度でデータに出現しているためファセットとしてあまり好ましくないと考えられる．

3.4.2　ファセット均衡度

ファセット均衡度はファセット毎の検索できるデータ数の分布である．バランスの良いファセットほど検索できるデータの範囲が広く，適切である．ファセット均衡度の式は以下のようになる．$n(key_i)$ はキー key_i において検索できる検索対象データ数である．n_{key} はファセット f におけるキーの数である．μ は $n(key_i)$ の平均である．

$$bala(f) = 1 - \frac{\sum_{i=1}^{n} \mid n(key_i) - \mu \mid}{(n_{key} - 1)\mu + (N - \mu)} \quad (2)$$

表 5 を見るとキーの偏りの大きい「mdate」が最下位に来ている．

3.4.3　キー濃度

キー濃度は各ファセットにおけるキーの数の分布である．つまり，n_{key} の正規分布に基づいて評価することでキーがある程度限られた数のファセットを重視する，ということである．n_{key} はファセット f におけるキーの数である．μ は n_{key} の平

図 3　Choosing Facet

図 4　Setting Parameter

均で，σ^2 は分散である．

$$card(f) = \begin{cases} 0 & if n_{key} \leqq 1 \\ exp^{-\frac{(n_{key}-\mu)^2}{2\sigma^2}} & otherwise \end{cases} \quad (3)$$

表 6 を見ると「pages」が一番平均的なキーの数であったことがわかる.「author」が最低の値となっている原因としては各検索対象データに対して複数キーが存在するためにキーの絶対数が検索対象データの数よりも大きくなってしまい，悪い値となっている．

3.4.4　キー単調性

キー単調性はキーの単調性の指標である．あるファセットにおけるキー毎に検索できるデータ数の平均を評価する．このとき一定以上のデータを検索できるファセットが良いファセットであると言える．下記式の avg は $n(key_i)$ の平均で，μ，σ^2 はその平均と分散である．

$$mono(f) = \begin{cases} 0 & if avg < \frac{1}{\mu} + \frac{1}{\sigma^2} \\ 1 - exp^{-\frac{avg}{\mu}} & otherwise \end{cases} \quad (4)$$

表 7 を見ると下位六つのファセットが 0 となっている．この中の「key」を見ると，各キーが検索できるデータ数はすべて 1 である．これは「key」がデータを一意に識別する識別子のようなものであるためである．他の五つも検索対象データを一意に識別するようなキーを大量に保持しているために 0 となっている．

3.5　ファセットのランキング

本節ではファセットのランキングの手法について説明する．前節の各指標の値の線形和をとることでファセットのランキングをする．その際に重みづけをおこなう．各重みはインターフェース提供者が設定する．設定された重みを基に検索が行われるたびにランキングを行いファセットの順序の最適化を行う．ファセット毎のランキングに用いる値は次の式によって与えられる．

$$calc(f) = \begin{cases} \alpha * freq(f) + \beta * bala(f) \\ \quad + \gamma * card(f) + \delta * mono(f) \\ \qquad if \quad mono(f) > 0 \\ 0 \qquad otherwise \end{cases} \quad (5)$$

但し，$\alpha + \beta + \gamma + \delta = 1$ とする．

キー単調性が正の値のときに値をとる理由はキー単調性によって冗長なファセットを排除するためである．

3.6　インターフェースの生成

前章でのランキングの値を基にファセットの順番を決定し，インターフェースを生成する．まずファセットのデータベースから同じキーをグルーピングし，各キーの数をカウントしたものの上位十五件を取得する．それらをランキングに基づく順序に並べた HTML ファイルをトップページ用に作成する．次章で実際に作成したインターフェースの例を示す．

4.　*FoX* の適用例 (DBLP)

本章では *FoX* を DBLP の論文データに対して適用した例を示す．

最初の処理としてデータベースの情報を登録する．これによって *FoX* が各種データベースへアクセスできるようになる．次に XPath を用いて検索対象データを指定する．以下のような XPath で指定する．

$/dblp/*$

このパスは DBLP の論文データのすべてを検索対象にする，という意味である．ここで *FoX* は DataGuide を用いて指定されたデータについてすべてのファセットの候補を抽出し表示する．インターフェース提供者は表示された候補の中から必要なファ

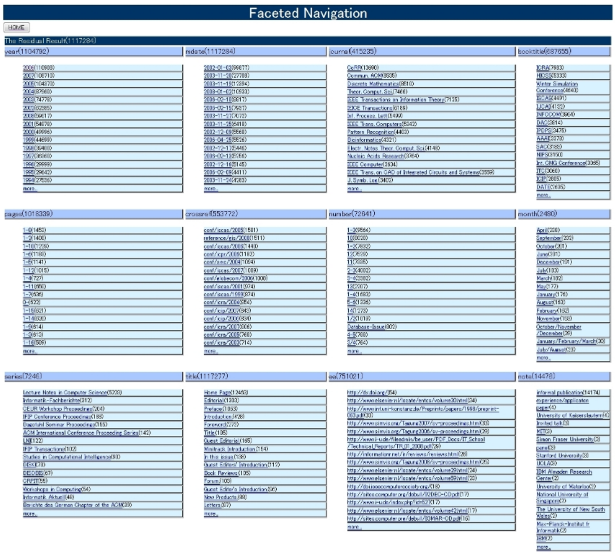

図 5　Top Page

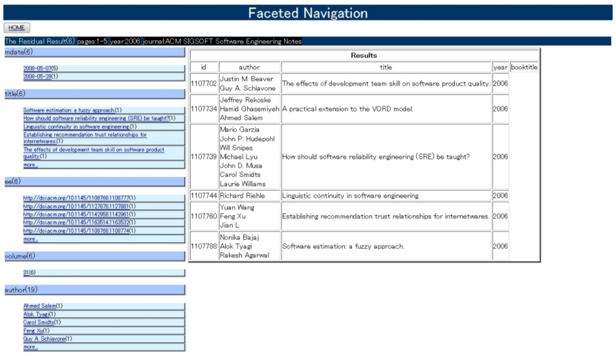

図 6　Results of navigation

セットを選択し FoX に登録する．図 3 は表示されたファセットを選択する画面である．図の左側部分がすべてのファセットでチェックボックスにチェックを入れることでファセットを選択する．同時に右側のラジオボックスから階層構造をどのように表示するかを選択する．

その後，FoX は選択されたファセットのすべてのキーを抽出し関係データベースへ格納する．関係データベースに格納されたキーを基にファセット毎に四つのランキングの指標を評価する．インターフェース提供者は四つのパラメータを設定し指標に重みづけをする．図 4 は四つのランキングのための指標の重みを決める画面である．図のスライドバーを動かすことで重みを 0 から 100 の間で決定する．内部的には四つのパラメータの合計で割ることで重み付き平均となる．右側のファセットのリストはスライドバーと連動して順位が入れ替わる．ここでデータの性質に合わせてパラメータを設定することでより効率的に検索ができるようになる．ここで設定したパラメータは実際の検索時の再ランキングにも用いられる．これらを基にファセット検索の初期コンテンツを作成してフォーマットを完了する．

図 5 は実際に FoX のプロトタイプで作成したファセット検索インターフェースである．トップページはファセットとキーが並んでいるだけで一切の結果は表示していない．その代りにキーの表示数を通常時よりも多くすることで，ユーザがほしいと思うデータに対するイメージに近い値が表示されている可能性を高めている．

最後に図 6 はファセット検索の一つの終わりの形である．表示できる結果の数が四つだけなのでこれ以上絞り込まずに目視によってデータを探すことができる．この結果に至るまでにファセットは三つしか使っていないことにも注目してほしい．このことからもファセット検索が効率的な探索手法であると言える．

5. ランキングの評価

本章では前章で適用したフレームワークで生成されたインターフェースのランキングの評価を行う．なお，前章の例同様に対象データは DBLP の論文データで検索対象はすべての種類の論文とした．ファセットの抽出などの手順は前述した通りに行い，ランキングのパラメータに関しては動作の評価時はすべて 50 に固定した．結果は以下のようになった．図 7 から図 10 まで検索の一連の流れに沿った指標の値とランキングの値を示している．棒グラフの内訳は，青がファセット頻度，赤がファセット均衡度，緑がキー濃度，紫がキー単調性である．水色の折れ線グラフはランキングに使用する値で今回はパラメータがすべて同じなのでそれぞれの値の和となっている．但し，キー単調性の値が 0 のものはランキングの値は 0 となっている．

まず初期のファセットのランキングを表している図 7 を見ると，ファセット頻度やキー単調性の二つが高いものが上位に来ることがわかる．逆にファセット均衡度とキー濃度のみが高いものが下位に来ており，キーによって検索できる数が単調なファセットほど下位にランキングされていることが言える．ランキングの値が 0 となっている「key」は論文データに対して一対一で対応しているためキー単調性が 0 となりランキングの値も 0 となっている．

ここでファセット「year」からキー「2006」と選択し検索を行うと次の図 8 のようにランキングの値が変化する．注目してもらいたいのは「journal」よりも右側のランキング値が 0 となっている点である．このようにしてランキングの値が 0 となるファセットの特徴としては各キーによって検索できる検索対象がごく少数であることが挙げられる．このようにすることで単調なファセットを削減することができる．以下同様に図 9，図 10 と遷移していく．

6. 関連研究

本章ではファセット検索についての関連研究について述べる．ファセット検索を使った手法についての研究とファセットのランキングに関する研究，ファセットを使った情報検索のためのフレームワークの研究を以下で紹介する．

6.1 ファセット検索を使った研究

まずファセット検索を使った研究を二つ紹介する．

ひとつは，Oren らによって提案された RDF を対象としたファセット検索 [10] である．この研究では RDF の構造を考慮した手法が展開されている．有向グラフの向きを利用した検索

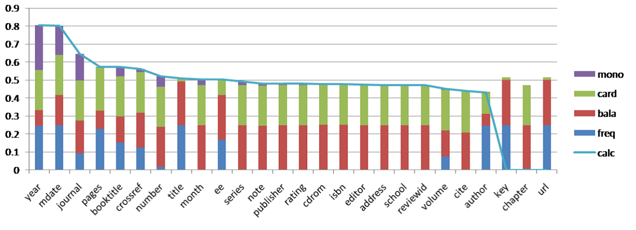

図 7　At the beginning

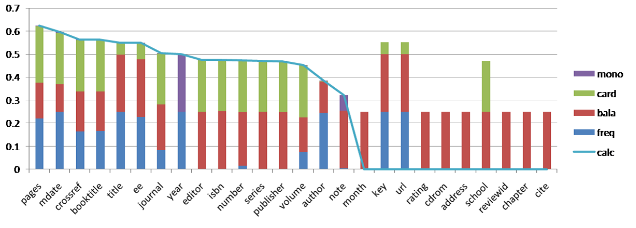

図 8　year=2006

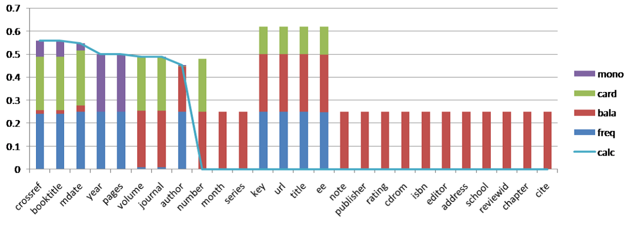

図 9　year=2006 & pages=1-5

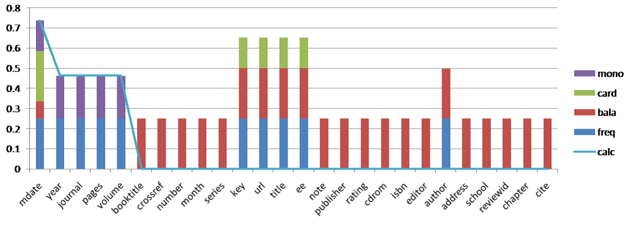

図 10　year=2006 & pages=1-5
& journal=ACM SIGSOFT Software Engineering Notes

方法がなされている．

もうひとつは，Koren らによって提案された超巨大ファイルシステムを対象としたファセット検索 [11] である．超巨大ファイルシステムのデータ検索を情報抽出の一部と位置づけファセット検索を適応させている．

6.2　ファセットのランキング

次にファセットのランキングを扱った研究を紹介する．

Roy らによって構造データベースにおけるファセット検索を最小限の操作で検索できるようにすること [12] が提案されている．手法としては，ファセットとキーを選択するという操作を決定木の一部とみなし，それらを合わせた決定木の高さが最小になるようにするといものである．しかし，この手法はデータの構造が明確に決まっているためにできる方法で，XML などの半構造データには適応させがたい．

また，前述の Oren らの手法 [10] では，ファセット頻度，ファセット均衡度，キー濃度という指標を用いてランキングを行おうとしていた．だが結局それらの有用性は示したものの，その利用法は検討課題としており実現されていない．

本研究ではこの頻度，バランス，濃度のほかにキー単調性という概念を追加することで更なるランキングの精度向上とランキング手法の実現を目指す．

6.3　ファセット検索フレームワーク

最後にファセット検索のためのフレームワークの研究 [13] を紹介する．これは Tzitzikas らによって *FleXplorer* として提案された．ファセット検索のミドルウェアとしてのフレームワークが提案されており，検索の高速化を目標としている．またファセットの階層構造を扱うための手法として階層構造を記憶するためのテーブルを用いている．

本研究とは階層構造を考慮するという点は共通であるが，その手法としてテーブルを用いるという点で本研究とは異なる．また，この研究ではファセットとキーの抽出については触れられておらず，この点に関しても本研究とは異なる．

7. ま と め

半構造データである XML におけるファセット検索インターフェースを生成するためのフレームワーク *FoX* を提案した．*FoX* と他のファセット検索に関する研究との違いは，扱うデータが半構造データである点やファセットの順序等を考慮している点である．扱うデータが半構造データであるために正確にデータの構造を知ることは困難である．これはファセット検索インターフェース提供者にとっては問題である．その点に対して本研究では DataGuide を用いてインターフェース提供者に提示する手法を提案した．これによってファセットのキーが XML の部分木になってしまうケースに対し構造の意味を喪失することなく表現できるようになった．また，ランキングについては四つの指標を用いることでインターフェース提供者の好みに合った，または検索対象 XML データに適した，ランキングを行えるようにした．これを基にファセットのランキングを動的に行えるようにした．ランキングの動向を評価することで動的にランキングできることの有用性を示した．今後の展望としては，階層構造の定義や検索結果の形式をインターフェース提供者が自由に定義できるようにすることでよりインターフェース提供者の意向に沿ったインターフェースを設計できるようにする．

謝辞　本研究の一部は科学研究費補助金（# 19024006）による．
本研究の一部は e サイエンスのためのオーバレイネットワークに基づく XML データ流通機構の研究（# 19700083）による．

文　　献

[1] “The World Wide Web Consortium”. http://www.w3.org/.
[2] 天笠俊之，石井理修，吉江友照，建部修見，佐藤三久，“XML データを対象としたファセット検索インターフェースの生成,” 情報処理学会研究報告「デジタルドキュメント (DD)」，vol.2008，no.53，pp.7–13，2008.
[3] “XML”. http://www.w3.org/XML/.

[4] “W3C XML Query”. http://www.w3.org/XML/Query/.

[5] “XML Path Language”. http://www.w3.org/TR/xpath/.

[6] Ryen W., White, B. Kules, Steven M., Drucker, and M.C. Schraefel, “Supporting exploratory search,” Introduction, CommuSupporting Exploratory Senications of the ACM, vol.49, no.4, pp.36–39, 2006.

[7] M. Ley, “DBLP Bibliography”. http://www.informatik.uni-trier.de/ ley/db/.

[8] “The Flamenco Search”. http://flamenco.berkeley.edu/.

[9] R. Goldman and J. Widom, “DataGuides: Enabling query formulation and optimization in semistructured databases,” VLDB, eds. by M. Jarke, M.J. Carey, K.R. Dittrich, F.H. Lochovsky, P. Loucopoulos, and M.A. Jeusfeld, pp.436–445, Morgan Kaufmann, 1997.

[10] E. Oren, R. Delbru, and S. Decker, “Extending faceted navigation for RDF data,” International Semantic Web Conference, eds. by I.F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, vol.4273, pp.559–572, Lecture Notes in Computer Science, Springer, 2006.

[11] J. Koren, A. Leung, Y. Zhang, C. Maltzahn, S. Ames, and E.L. Miller, “Searching and navigating petabyte-scale file systems based on facets,” PDSW, ed. by G.A. Gibson, pp.21–25, ACM Press, 2007.

[12] S.B. Roy, H. Wang, G. Das, U. Nambiar, and M.K. Mohania, “Minimum-effort driven dynamic faceted search in structured databases,” CIKM, eds. by J.G. Shanahan, S. Amer-Yahia, I. Manolescu, Y. Zhang, D.A. Evans, A. Kolcz, K.-S. Choi, and A. Chowdhury, pp.13–22, ACM, 2008.

[13] Y. Tzitzikas, N. Armenatzoglou, and P. Papadakos, “FleXplorer: A framework for providing faceted and dynamic taxonomy-based information exploration,” DEXA Workshops, pp.392–396, IEEE Computer Society, 2008.