P2P 環境における RDF データを対象とした Faceted Search の実現

齋藤　真衣†　　渡辺知恵美†

† お茶の水女子大学大学院人間文化創成科学研究科　〒 112-8610 東京都文京区大塚 2-1-1
E-mail: †{saito.mai,chiemi}@is.ocha.ac.jp

あらまし　今日個人で所有されている多種多様なデータを互いに公開・検索し合うことで，より活発な知識共有に結びつけることが可能である．また P2P における様々なデータ検索手法が提供されており，P2P アプリケーションはそれらの検索手法を利用したデータ検索を実現している．しかし人によってつけたメタデータが統一されていない場合，簡単なキーワード検索だけで所望のデータに辿り着くことは困難であり，ユーザにスムーズな検索を促す検索インタフェース実現のためにはユーザの検索を支援するためのデザインを考慮する必要がある．本稿では P2P アプリケーションの検索デザインとして，検索条件を予めリストアップしてユーザに提示して検索目的が曖昧なユーザに対して対話的に絞込み検索を行う手法として近年注目されている Faceted Search に着目し，メタデータを記述する枠組みとして RDF を対象とした上で，P2P で Faceted Search 実現のための手法を提案する．

キーワード　P2P，ファセット検索, RDF

A cross-search mechanism on the P2P overlay network using Faceted Search for RDF data

Mai SAITO† and Chiemi WATANABE†

† Graduate School of Humanities and Sciences, Ochanomizu University
E-mail: †{saito.mai,chiemi}@is.ocha.ac.jp

Abstract　By development of the technologies for P2P network, we can share various type of data such as relational table, xml data and so on, and we can query the data not only by simple keyword search but also SQL-like query expression. Applications which are developed by using these technologies should design query interface so that it helps users to reach any information they want. Especially on P2P network, when there is no rule for adding metadata of the objects, it is not easy to find appropriate keywords for searching. In this paper, we focus on ’faceted search’, which is one of design patterns for helping query behaviours, and we investigate some architectures for applying faceted search on P2P network.

Key words　P2P，Faceted Search, RDF

1. はじめに

今日さまざまなデータが個人で所有されている中，それを互いに公開・検索することで，より活発な知識共有へと結びつけることが可能である．ファイル共有システムなど個々人の間で P2P による様々なデータ検索手法が研究されており，分散ハッシュテーブル（DHT）のキーによる検索やそれを応用させたキーワードによる部分一致検索，リレーショナルデータベース（RDB）を P2P で扱うためのスキーム PIER [8] 等が提案されている．P2P アプリケーションはこれらの検索手法を利用し，ユーザの必要とするデータの検索を実現する．検索時，ユーザは目的のデータを取得するまでに問合せを行ってその結果を得，そしてその結果をもとに問合せを行うという操作を繰り返す．ユーザが所望するデータに辿り着けるかどうかは問合せの条件に左右されることから，ユーザのスムーズな検索を支援するような検索インタフェースのデザインを考慮することが重要となる．特に P2P アプリケーションの場合には P2P にて共有するデータに所有者が銘々にメタデータをつける場合，人によって同じオブジェクトに対しても異なるメタデータをつける場合もあり，簡単なキーワード検索のみで必要なものを絞り込むことが難しい場合も考えられる．そこで先行研究 [12] では，P2P アプリケーションにおける検索デザインとして Faceted Search が有用と思われることから，Faceted Search を実現するためのリレーショナルスキームの提案，及び P2P における検索コストの比較, [13] では，P2P におけるメタデータの均等な分散管理方法について述べた．

本稿ではメタデータを記述する枠組みとして RDF(Resource Description Framework) [1] に着目し，P2P で RDF データを対象とした Faceted Search の実現に適切なシステム構成を提案する．2 節で前提知識として対象データ構造と Faceted Search，3 節で関連研究を紹介し，4 節で P2P における Faceted Search，5 節で実験，最後に 6 節でまとめと今後の課題を提示する．

2. 前　　提

本節ではまず想定する状況と対象データ構造を述べ，それを踏まえた上で考慮すべき検索時のデザインパターンの中で着目した，Faceted Search について詳細を示す．

2.1 想定する状況と対象データ構造

本項では例えばある P2P アプリケーションにおいて，ユーザは互いにレビューした書籍のデータを共有することを想定し，以下議論を行う．想定する状況において，ユーザは所有するデータに対し，そのデータがどんなデータであるかを定義するためにメタデータを付与する．例えば書籍データに対して，“ 2009 年に出版された書籍のタイトル ”といったように，その属性を組み合わせて記述することで相互関係を辿ることができる．このとき付けるべきメタデータにルールを定めず，構造に従えば任意に付けられるとすると，例えば＜ genre：小説＞など＜属性名：属性値＞といった形で，タグのようにユーザの自由に付与することが考えられる．このような場合，別のユーザにとっては自分が求めたいデータに対してどんなタグが付けられているのか分からず，すぐに目的のデータに辿り着くことが困難となってしまう．特に漠然とした目的を持ったユーザにとっては，興味のあるデータに辿り着くためには，何をキーワードに指定すればよいか見当がつかない．そこでそのようなユーザに対して検索を支援することが求められる．通常ユーザは，何かのアプリケーションを通して検索を行うことが多いため，ユーザが検索に利用するアプリケーションのデザインパターンの観点から，検索をサポートする必要があると考えられる．

またメタデータの記述方法の一つとしては，図 1 に示したような RDF が挙げられる．RDF とは，トリプルと呼ばれる主

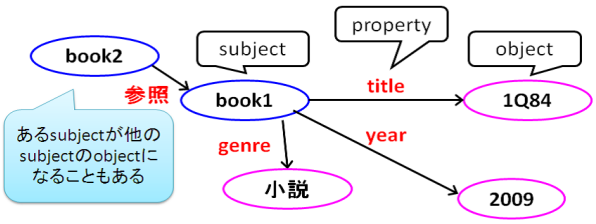

図 1　RDF トリプル

語（subject），述語（property），目的語（object）の 3 つの要素から成る．subject はリソース,property は subject の特徴や subject と object の関係，object は subject と関係するリソース，もしくは property の値を示す．RDF に従ってメタデータを記述することで柔軟な表現が可能となり，その代表例としては人に関する情報を記述する FOAF(The Friend of a Friend) [2] や，音楽データに関する情報を記述する MusicBrainz [3] がある．以下，本稿では RDF で記述されたメタデータを対象とする．

2.2 Faceted Search

一般的なアプリケーションでは [9] で示されているように，多様なデザインパターンがある．RDF データの検索方法として Faceted Search を用いたものが [5] [6] で提案されており，本稿では Faceted Search に着目して以下議論を行う．Faceted Search とは，データの検索条件として属性やメタデータ項目を予めリスト表示しておき，それを選択することで検索クエリなしでユーザを目的のデータに導くことができるインタフェースである．検索対象データの属性名をファセット，その属性値をファセット要素としたとき，Faceted Search ではまずファセットを選択し，次にそのファセット要素を選択するという操作を繰り返すことで，検索対象データを絞り込んでいく．リストからファセットを選択すると，そのファセット要素と該当件数がリストアップされ，ファセット要素が選択されると，絞り込まれた検索対象データの持つファセットが新たにリストアップされる．すなわち検索を進めるごとに集約演算が行われ，検索対象データを絞込んでいく．Faceted Search の代表例として，Flamenco Search [10] が挙げられる．図 2 に Flamenco Search のインタフェースを示す．Flamenco Search では，受賞年や受

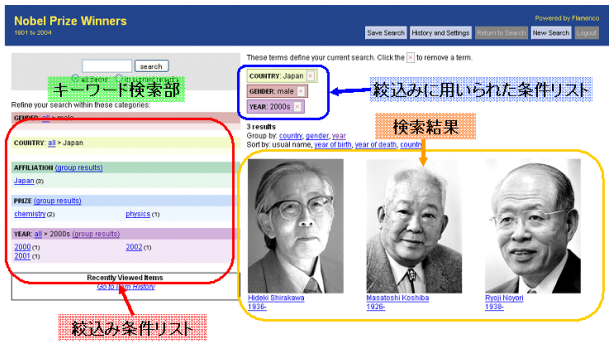

図 2　Flamenco Search

賞分野を指定することで歴代のノーベル賞受賞者の中から該当データを絞り込むことができる．絞込み検索の流れを Flamenco Search を例に詳しく述べる．まず歴代のノーベル賞受賞者全員のデータに対し，例えば属性名＝ GENDER の属性値＝ male で絞り込む．この 1 回の絞込みにより，属性名＝ GENDER の属性値=male という属性を持つノーベル賞受賞者のデータが検索対象データとなり，検索対象データが持つ属性とその該当件数がインタフェースに表示される．さらにその検索対象データに対し，属性名＝ COUNTRY の属性値=Japan という属性で絞り込むと，GENDER=male と COUNTRY=Japan を共に満たすノーベル賞受賞者のデータが検索対象となり，同様に次の絞込み条件として検索対象データの属性と該当件数がユーザに提示される．

Faceted Search の特徴としては，以下の 2 点が挙げられる．

(1)　集約処理を繰り返す
(2)　絞込みによって問合せが繰り返し発行される

3. 関連研究

本稿では対象データを RDF データとすることから，本節では関連研究として RDF データを P2P で共有する方法，さらに P2P 上で処理するために PIER と組み合わせることから，RDF データを RDB に格納する方法について紹介する．

まず RDF データの P2P における共有に関して，RDFPeers [7] や RDFCube [11] などが提案されている．RDFPeers とは分散ハッシュテーブル（DHT）を用いて RDF トリプルをネットワーク上のノードに格納し，その検索を実現する分散 RDF データベースである．DHT とはキーと値をペアにしたハッシュテーブルを各ノードで分散管理し，キーによって効率的な検索を行うものである．RDF トリプルの subject, property, object それぞれをキー，RDF トリプルそのものをキーに対する値とし，1 つのトリプルに対して 3 つのキーが割り当てられる．subject, property, object のいずれかを条件とすれば，該当するトリプルを得ることができる．それに対し RDFCube とは subject, property, object に基づいた三次元構造のハッシュ空間 (RDFCube) に RDF データを写像し，それが RDFCube の一定領域内に存在するか否かをビットで表現する．導入したビットの演算を行うことで，RDF データのハッシュ値の範囲を絞り込んだ後，RDF データそのものの結合演算を行い，分散環境での結合演算の効率を図る．

また,Adabi 氏らによって RDF データの RDB への格納に関する研究が提案されている [4]．この研究では，RDF データを RDB に格納する 3 つのアプローチとして，Triple Store, Property Tables, Vertical Partition が挙げられている．また [4] では，検索アプリケーション例として彼らが開発している RDF の Faceted Search による検索システム Longwell [5] をあげ，当該システムで Faceted Search による検索で発行される問合せ文をあげ，それらの性能を評価している．

RDF データを P2P で扱う技術 [7] [11]，及び RDF データを Faceted Search で検索するための技術 [6] はそれぞれ提案されているが，それらを組み合わせたものはこれまで確認されていない．

4. P2P における Faceted Search の実現

本節では P2P において Faceted Search を実現するため，4.1 項で RDF データに対する問合せオペレーション，4.2 項で P2P における問合せパターン，4.3 項で DHT による分散管理，4.4 項で RDF データの格納方法について述べ，4.5 項で問合せの流れについて示す．なお本稿で想定する P2P ネットワークは，関連研究の多くと同様にセンターサーバのないピュア P2P である．

4.1 RDF データに対する問合せオペレーション

RDF データに対する主な問合せオペレーションとし，て [6] で示された中から本稿で扱うものを以下に挙げる．なお図 3 に示した RDF データを用い，簡単な問合せ例として挙げる．

- **Selection**（単一条件による絞込み）

例)“ genre:小説 ”である書籍を求める（図 3(a)）

- **Intersection**（複数条件による絞込み）

例)“ genre:小説 ”かつ“ year:2009 ”である書籍を求める（図 3(b)）

- **Inverse selection**（逆選択）

例)“ title:ノルウェイの森 ”である書籍に“ 参照されている ”書籍を求める（図 3(c)）

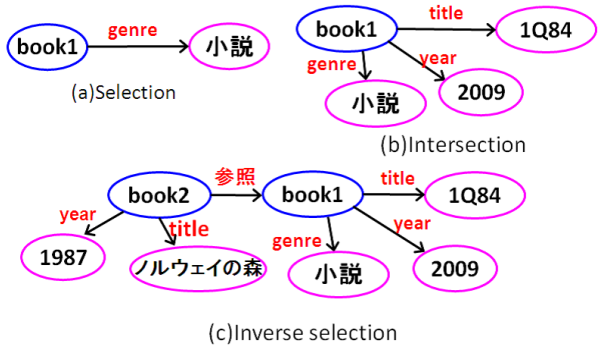

図 3　問合せオペレーションとその RDF データ例

4.2 P2P における問合せパターン

想定するピュア P2P に関して，探索方法をもとに大まかに以下の 3 つに分類できる．(図 4)

A　key とそれに対応する value をネットワーク上で探索し，効率的に検索する方法（構造的 P2P 型）（図 4(a)）
ハッシュ関数によって算出された key と，その value のペアを各ノードで分散管理し（DHT），検索時には key を指定することで，その値を取得することが可能である．

B　各ノードが周囲のノードにメッセージを送り，連鎖的に転送していく方法（フラッディング型）（図 4(b)）
転送先に該当ノードが見つかればよいが，ネットワーク内の全ノードに対して確実に検索が行えない可能性が高い．

C　ネットワーク上の全ノードに対して一斉に問合せを発行する方法（ブロードキャスト型）（図 4(c)）
問合せを受け取った全ノードからの集約結果を取得する．

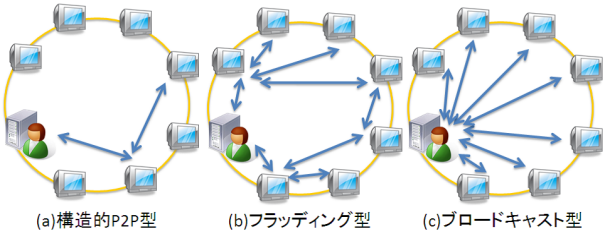

図 4　P2P における問合せパターン

2.2 項で述べたように Faceted Search の特徴の 1 つとして，集約演算を繰り返すことから，C のブロードキャスト方式による問合せ結果の取得，集約処理が有効であると考えられるが，もう 1 つの特徴として挙げられる問合せの繰り返しも考慮する必要がある．1 回の絞込みごとに問合せを発行した場合，C の方法では毎回ブロードキャストの実行が必要となり，トラフィッ

クの増加が見込まれる．それに対し A の方法では効率的にノードを探索でき，B や C に比べて，問合せを複数回重ねてもトラフィックが極端に増加することはないことから，本稿では DHT を用いて Faceted Search を実現することとする．

4.3 RDF データの格納方法

本稿では RDF データを扱うためにリレーショナルデータに変換し，DB へ格納することとする．先行研究 [12] では [4] で提案された 3 つの RDF データの格納方法に関し，P2P 上で処理を行った場合について考察した結果，Faceted Search における絞込み検索実現のためにはタプルの結合が必須であり，それによる処理時間の増大を考慮した結果予め結合処理を行ったタプルを格納する方法が効率的であると分かった．

4.1 項で挙げた各オペレーションに関し，まず単純な RDF トリプルを 1 タプルとして格納することで単一条件により該当 subject を絞り込む Selection を実現できるが，RDF データ構造は図 3(b) で示したように 1 つの subject に対して複数の property，object が定義されていることが一般的である．そこで [12] で提案したように，subject を自己結合させて生成したタプル（図 5(a)）を格納することで，その構造をリレーショナルデータとして表すことができ，複数条件を満たす subject を絞った上でその subject が持つその他の property, object の集約処理，すなわち Intersection が可能となる．さらに図 3(c) に

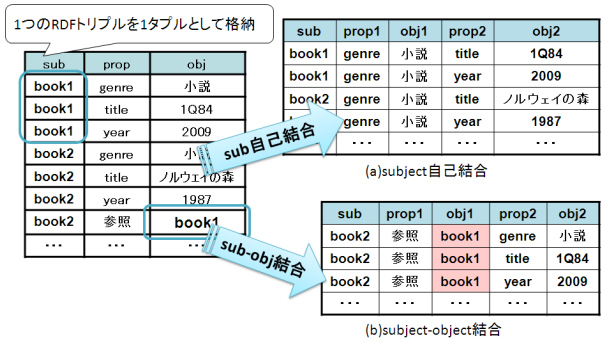

図 5　各オペレーション実現のためのテーブル構成

あるように，RDF ではある subject が他の subject の object として定義される場合がある．そこで図 5(b) のように subject と object の結合によって生成したタプルでデータ構造を表し，Inverse selection オペレーションを実現できると考えられる．

4.4 DHT における分散管理

前述の通り，DHT では（key，value）のペアを各ノードで分散管理し，検索時には指定した key によってそれに対応する value を取得することができる．Chord や Pastry といったアルゴリズムが提案されており，どのアルゴリズムにおいても以下のような関数がある．

DHT への登録：*put(key, value)*

DHT からの取得：*get(key)*

つまり key，value を決めるだけで，あとはアルゴリズム任せで DHT を構築できることから，何を key，何を value として指定するかが重要である．Faceted Search では検索条件として property や object が指定されたときに，それに該当する subject が持つその他の property, object の情報が必要であることから，本稿では検索条件となる property や object を key，それに該当するタプルを value とする．put したペアに対し，検索時にはユーザが 1 回目に指定した条件を key とし，get によって得られた該当タプルに対して集約処理を行う．

4.5 想定する問合せの流れ

本稿で想定している検索の Step を以下に示す．

Step1. 検索条件リストの中から 1 回目の絞込み条件（例えば genre:小説）を指定する

Step2. 1 で指定された条件を key とし，それを保持するノード（Node S）で該当タプルを絞り込む

Step3. 集約結果として次の検索条件候補，及びその該当件数を取得する

Step4. 3 で取得した条件の中から，ユーザは次の絞込み条件（例えば year:2009）を選択する

Step5. 2〜4 を繰り返し，目的のデータを絞り込む

Step2 において，本稿では図 6 に示した 2 通りの手法を提案する．

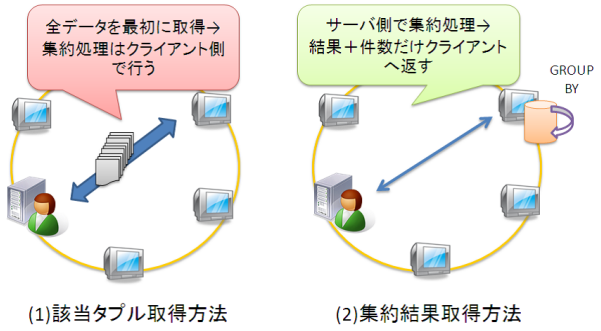

図 6　集約処理方法

♣　手法 (1)

Node S から該当タプルをそのまま取得し，ユーザ側で集約処理を行う方法

1 回目の絞込み条件（genre:小説）に該当する全タプルを取得し，その後の絞込み検索は P2P 上で問合せることなく，ユーザ側で実行する．ただし 4.1 項で述べたように，RDF のデータ構造を表すためにタプルを結合しているためタプル数が多く，問合せごとに該当タプルをそのまま取得するとトラフィックが増大してしまうことが考えられる．

♣　手法 (2)

Node S 側で集約処理を行い，集約結果のみを取得する方法

絞込みごとに P2P 上で問合せを行い，その都度該当ノードで集約処理を行う．例えば，まず genre:小説という条件で絞り込む場合，genre:小説という key に該当する Node S で絞込みを行い，さらに year:2009 という条件で絞り込む際には 1 回目の絞込み条件 genre:小説を key に持つ Node S 側で，genre:小説かつ year:2009 という条件に該当するデータを絞込み，ユーザ側は集約結果（次の検索条件候補とその該当件数）のみを取得する．絞込み条件の数だけ P2P 上での問合せが必要となるが，ユーザ側では集約結果のみを受け取るため，トラフィックは抑えられる．

また同じく Step2 において 1 回目の絞込み条件を key として固定すると，1 回目に指定されやすい条件を key として割り当てられたノードに対し，問合せが集中してしまうことが想定される．そこで Faceted Search における問合せでは絞込み条件に順序関係がないことから，条件が複数になった場合にはその中から key とする条件をランダムに抽出し，問合せごとに key を変えることで，検索を進めてもあるノードだけに処理が集中しないようにすることが有効と考えられる．

なお，Step1 で最初にユーザに提示すべき全検索条件はあるサーバ（Start Server）でリストとして保持しておき，ユーザは Start Server から取得した条件リストから 1 回目の検索条件を選択することとする．

5. 実　　験

4.5 項で述べた Step2 における 2 つの提案手法に関して，人工データを用いて実験を行い，2 台のノード上で 2 回の絞込みに要する検索時間を比較した．実験に用いた RDF データは 10,000 種類の subject に対し，property を 10 種類，object を 20 種類ずつランダムに定義したものである．ここでは Intersection オペレーションを想定して subject の自己結合を行い，その結果生成されたおよそ 9,000,000 タプルを検索対象とした．4.4 項で述べたように，本稿では検索条件を key にタプル全体を P2P 上で分散管理することから，生成された 9,000,000 タプルのうち，ある条件を key にして 1 台のノードで管理するタプル数は property，object によって異なる．そこで，1 回目の絞込みに用いる各条件（property と object の各ペア）を変えて問合せを行い，各条件ごとの subject 選択率（該当 subject 数/全体の subject 数）別の処理時間の比較を行った．なお，実験では 2 回目の絞込みにおける subject 選択率を 50%に固定し，1 回目の絞込み時の subject 選択率のみを変動させた．

手法 (1) では，まず 1 回目の条件に該当するタプルのみを保持する Node S からそのまま取得し（①），クライアント側のオンメモリ DB へタプルを挿入する（②）．メモリ DB 側で集約処理した結果（③）が 1 回目の絞込み結果となり，2 回目の絞込みはメモリ DB 上で行う（④）．(図 7)

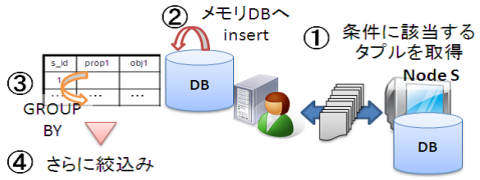

図 7　手法 (1) による実験の流れ

上記の各過程における問合せ例を以下に示す．

① SELECT *
FROM Triples

② Insert into Triples_onMemory values (sub=”book1”, prop1=”genre”, obj1=”小説”,prop2=”year”, obj2=”2009”)

③ SELECT distinct prop2, obj2, count(*)
FROM Triples_onMemory
GROUP BY prop2, obj2
ORDER BY count(*)

④ SELECT distinct t1.prop2, t1.obj2, count(*)
FROM Triples_onMemory t1, Triples_onMemory t2
WHERE t1.sub=t2.sub AND t2.prop2=’year’ AND t2.obj2=’2009’
GROUP BY t1.prop2, t1.obj2
ORDER BY count(*)

同様に手法 (2) における Node S 側での問合せ例を以下に示す．

- 1 回目の絞込み

```
SELECT distinct prop2, obj2, count(*)
FROM Triples
GROUP BY prop2, obj2
ORDER BY count(*)
```

- 2 回目の絞込み

```
SELECT distinct t1.prop2, t1.obj2, count(*)
FROM Triples t1, Triples t2
WHERE t1.sub=t2.sub AND t2.prop2=’year’ AND t2.obj2=’2009’
GROUP BY t1.prop2, t1.obj2
ORDER BY count(*)
```

各手法における subject 選択率別の処理時間を示した図 8 より，選択率が低いとき，すなわち 1 つのノードで管理しているタプル数が少ないときは手法 (1) の方が処理時間が少ないが，選択率が高くなるにつれ，手法 (2) の方が計算時間を低減させられることが分かる．手法 (1) では，該当 subject があまり絞

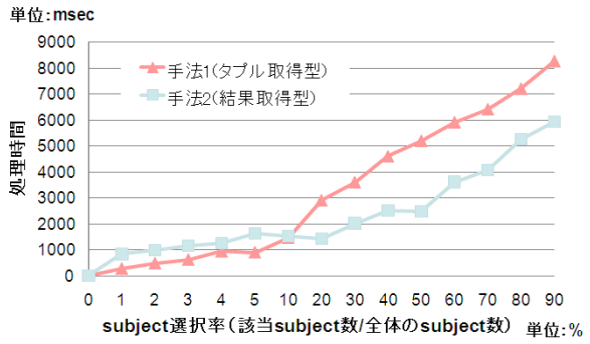

図 8　subject 選択率別の処理時間

られていない場合，該当する多くのタプルをそのまま取得し，さらにそのタプルをクライアント側の DB へ挿入するため，処理時間が増大していると考えられる．以上のことから，まず手法 (2) によって Node S で絞込みを行い，ある程度該当 subject が絞られたら手法 (1) を用いてタプルごと取得し，その後の絞込みはクライアントで行うことで処理時間，トラフィックを抑えることとする．ただし，2 つの手法をどの時点で切り替えるかを決定する必要がある．また，ネットワークの帯域幅や安定性によって，各手法の処理時間に影響があると考えられる他，

クライアント側ノードがシンクライアントの場合，手法 (2) で Node S 側で処理を行うことが望ましい一方，そもそもネットワーク全体で共有するデータ量が少ない場合は，最初から手法 (1) を選択すればよいことから，さまざまな環境，状況に沿ったシステム構成が求められると考えられる．

6. まとめと今後の課題

本稿では柔軟な表現ができる RDF によって記述されたメタデータを検索対象とした上で，我々ユーザが用いるさまざまな検索パターンの中でもユーザのよりスムーズな検索をサポートする Faceted Search に着目し，P2P における Faceted Search 実現のための手法について述べた．今後はさまざまな環境，状況における実験を行うと共に，検索時のトラフィックやデータの選択率，データ更新時の処理についても考察を進めていきたい．

文　献

[1] World Wide Web Consortium：Resource Description Framework(RDF).“ http://www.w3.org/RDF/ ”

[2] The Friend of a Friend (FOAF) project“ http://www.foaf-project.org/ ”

[3] MusicBrainz“ http://musicbrainz.org/ ”

[4] D. Abadi, A. Marcus, S. Madden, and K. Hollenbach.: “ Scalable Semantic WebData Management Using Vertical Partitioning, ” In *Proc. the 33rd International Conference on Very Large Data Bases*, pp. 411–422, September 2007.

[5] Longwell“ http://simile.mit.edu/wiki/Longwell ”

[6] E. Oren, R. Delbru, and S. Decker.: “ Extending faceted navigation for RDF data, ” In *Proc. the 5th International Semantic Web Conference*, pp. 559–572, November 2006.

[7] M. Cai and M. Frank.: “ RDFPeers: A Scalable Distributed RDF Repository based on A Structured Peer-to-Peer Network, ”In *Proc. the 13th international conference on World Wide Web*, pp. 650–657, May 2004.

[8] R. Huebsch, J. Hellerstein, N. Lanham, B. T. Loo, S. Shenker, and I. Stoica.: “ Querying the Internet with PIER, ” In *Proc. the 29th International Conference on Very Large Data Bases*, pp. 321–332, September 2003.

[9] P. Morville: “ Search Patterns, ” In *Proc. iA SUMMIT*, April, 2008.

[10] Flamenco Search: UC Berkeley School of Information The Flamenco Search Interface Project “ http://flamenco.berkeley.edu/index.html ”

[11] 的野晃整, M.Said, 小島功：“ RDFCube: 分散 RDF データベースのための三次元ハッシュ索引, ”日本データベース学会論文誌, Vol.4, No.4, pp5-8(2006).

[12] 齋藤真衣, 渡辺知恵美：“ P2P 環境における Faceted Navigation インタフェース実現のための諸検討, ”情報処理学会研究会報告, 2008-DBS-146, pp.283-288 (2008).

[13] 齋藤真衣, 渡辺知恵美：“ P2P 環境における Faceted Navigation インタフェースの実現, ”第 1 回データ工学と情報マネジメントに関するフォーラム (DEIM2009), C9-4(2009).