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

理学専攻　情報科学コース　齋藤 真衣（指導教員：渡辺知恵美）

1　はじめに

今日個人が所有する様々なデータを互いに公開・検索することで，活発な知識共有が可能である．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結果を得，それをもとに問合せを行うという操作を繰り返す．ユーザが所望のデータに辿り着けるかどうかは問合せ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，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，同じオブジェクトに対しても，人によって異なるメタデータをつけることも考えられ，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困難となる．本稿では P2P において，検索デザインとして有用と思われる Faceted Search の実現について述べる．

2　前提

2.1　想定する状況と対象データ構造

例えばある P2P アプリケーションにおいて，ユーザが互いにレビューした書籍データを共有することを想定した場合，ユーザは所有するデータにメタデータを付与する．この際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自由に付与すると考えられるが，別のユーザにとっては所望のデータに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．特に検索目的が漠然としているユーザには，何をキーワードに検索すればよいか見当がつかない．そこで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から，アプリケーションのデザインの観点から検索をサポートする必要がある．

またメタデータの記述方法として，RDF を対象とする．RDF とはトリプルと呼ばれる主語（subject），述語（property），目的語（object）の 3 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．

2.2　Faceted Search

Faceted Search とは，データの検索条件として属性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，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，データが絞り込まれる．Faceted Search の代表例 Flamenco Search[2] では，受賞年や受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者が絞り込まれる．Faceted Search の特徴は，以下の通り．

(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　P2P における Faceted Search の実現

3.1　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に関し，図 1 の RDF データを用いて簡単な問合せ例を挙げる．

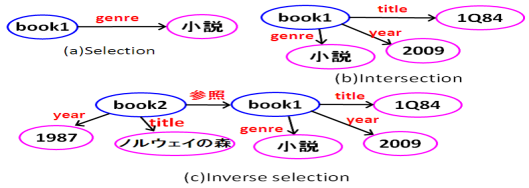

図 1: 問合せオペレーションとその RDF データ例

- **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 1(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 1(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 1(c)）

3.2　P2P における問合せパターン

想定するピュア P2P の大まかな分類を以下に示す．

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（DHT 型）

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）

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し，集約結果を得る方法（ブロードキャスト型）

2.2 項の通り，Faceted Search では集約を繰り返すことから，C による問合せ結果の取得，集約が有効と考えられるが，もう 1 つの特徴である問合せの繰返しも考慮すると，C では毎回ブロードキャストを要し，トラフィックの増加が見込まれる．それに対し A では効率的に検索でき，問合せを重ねても B や C に比べてトラフィックが増加しないことから，DHT を用いることとする．

3.3　RDF データの格納方法

本稿では RDF データをリレーショナルデータに変換し，DB へ格納する．[3] では [1] で提案された 3 つの RDF データの格納方法に関し，P2P 上での処理について考察した結果，Faceted Search 実現にはタプルの結合が必須であることから，予め結合処理を行う方法が効率的であると分かった．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 Selection を実現できるが，RDF データは 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（図 1(b)）．そこで subject の自己結合タプル（図 2(a)）によって，複数条件を満たす subject を絞り，その subject が持つその他の property, object の集約，すなわち Intersection が可能

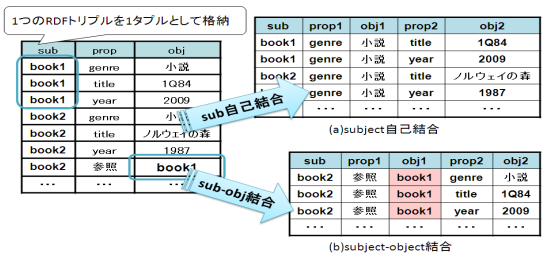

図 2: 各オペレーション実現のためのテーブル構成

となる．またある subject が他の subject の object として定義される Inverse selection（図 1(c)）では，subject と object の結合タプル（図 2(b)）を格納する．

3.4 DHT における分散管理

DHT では（key，value）の組を各ノードで分散管理し，検索時には key を指定することで対応する value を取得できる．各アルゴリズムには以下の関数がある．

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，検索条件となる property や object を key，その該当タプルを value として put を行い，検索時にはユーザが 1 回目に指定した条件を key とし，get で得られたタプルに対して集約を行う．

3.5 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．

1. 検索条件リストの中から 1 回目の絞込み条件を指定
2. 1 で指定された条件を key とし，該当ノード（Node S）に問合せ
3. 集約結果（次の検索条件候補とその該当件数）を取得
4. 3 で取得した条件から，ユーザは次の条件を選択
5. 2〜4 を繰り返し，目的のデータを絞込み

Step2 において提案する 2 つの手法を以下に示す．

(1) Node S から該当タプルを取得し，ユーザ側で集約を行う方法
1 回目の条件で該当タプルを取得し，その後の絞込みは P2P 上で問合せることなく，ユーザ側で行う．ただし結合によって増加したタプルを問合せごとに取得すると，トラフィックの増大が見込まれる．

(2) Node S で集約を行い，結果のみを取得する方法
絞込みごとに P2P 上で問合せをし，その都度該当ノードで集約を行う．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 上での問合せを要するが，ユーザ側では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 Step2 において 1 回目の絞込み条件を key に固定すると，1 回目に指定されやすい条件の key の該当ノードに対し，問合せが集中してしまう．そこで Faceted Search では絞込み条件に順序関係がないことから，複数条件の場合，key とする条件をランダムに抽出し，問合せの集中を避ける．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でリストとして保持しておき，ユーザは取得したリストから 1 回目の条件を選択する．

4 実験

3.5 項で述べた提案手法に関して，2 台のノード上で絞込み 2 回に要する処理時間の比較を行った．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．Intersection を想定して subject の自己結合を行い，その結果生成された 9,000,000 タプルを検索対象とした．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，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 選択率を変動させた．

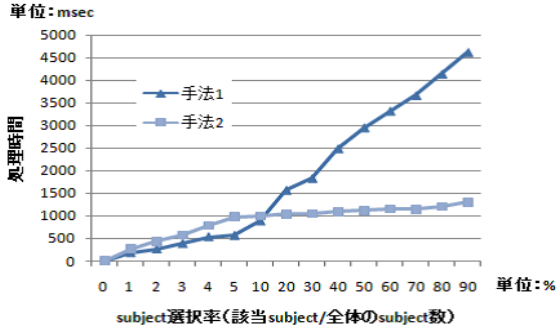

図 3: subject 選択率別の処理時間

図 3 に示した実験結果より，選択率が低いときは手法 (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，ある程度該当データが絞られたら手法 (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．

5 まとめと今後の課題

本稿では RDF データを対象とし，よりスムーズな検索を促す Faceted Search を P2P で実現する手法について述べた．今後はネットワーク全体のトラフィックについても考察を進めると共に，2 つの提案手法をどの時点で切り替えるかについて，決定する必要がある．

参考文献

[1] D. Abadi, et al.: “ Scalable Semantic WebData Management Using Vertical Partitioning, ”In *Proceedings of VLDB2007*, pp. 411–422.

[2] Flamenco Search:
“ http://flamenco.berkeley.edu/index.html ”

[3] 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).