DEIM Forum 2010 F5-3

多次元的なWeb空間マイニングを行うデータベースシステムの実現：制約条件の一般化

齋藤　太陽† 大森　匡† 星　守†

† 電気通信大学大学院情報システム学研究科　〒182–8585 東京都調布市調布ヶ丘 1–5–1
E-mail: †{saito,omori}@hol.is.uec.ac.jp

あらまし　近年，Web 空間上の互いに興味を持ち合うページ集合（コミュニティ）を抽出する Web コミュニティの研究が重要となってきている．また，個人や状況に応じて情報をパーソナライズして調べることが特に重視される．著者らの研究室では，多様な分析視点から Web 上のコミュニティマイニングを行う目的で，データキューブモデルに基づいた多次元制約の下で Web 構造マイニングを行うデータベースシステムを提案してきており，昨年度までに，FROM 型と TO 型制約によるデータキューブ問い合わせ処理の基本演算体系が示された．本稿では「ある特定部門についてより詳細に知りたい」「あるキーワード関連コミュニティが知りたい」というような一般的制約述語に対応した場合について評価を行い，問い合わせ木を示して本システムの有効な利用法を評価する．

キーワード　データウェアハウス，Web マイニング，データキューブ

A Report on Database Systems for Multi-Dimensional Web Mining By Using General Data-Cube Constraints

Taiyo SAITO†, Tadashi OHMORI†, and Mamoru HOSHI†

† The University of Electro-Communications, Graduate School of Information Systems,
Chofugaoka 1–5–1, Chofu, Tokyo, 182–8585 Japan
E-mail: †{saito,omori}@hol.is.uec.ac.jp

Abstract　Recently, the web-community mining has been researched. Moreover, personalization is very important. Last year, we proposed database systems for web-community mining by multi-dimensional constraints based on the data cube model and described basic operations for the data cube query processing using the “FROM-constraints” and “TO-constraint”. In this paper, we evaluate this database systems by using general data cube constraints (for example, “a part of departments in more details” or “communities based on a given keyword”).

Key words　data warehouse, Web mining, data cube

1. はじめに

近年，Web 上で仮想組織の活動が盛んになったことに伴い，Web 空間上の互いに興味を持ち合うページ集合を抽出する Web コミュニティの研究が重要となっている [1] [2]．また，個人や状況に応じて Web 空間情報を個別適応 (personalization) して調べることも重要であり，そのために，パーソナリゼーションに対応した Web データベースへの問い合わせ処理システムが研究されている [3] [4]．そこで，著者らは多様な分析視点から Web 上のコミュニティマイニングを行うことを目的として，データキューブモデルに基づいた多次元制約の下で Web 構造マイニングを行うデータベースシステムを’05 年から提案してきた．そして，特定組織（大学）のドメイン内の Web 空間を対象にして，本システムによるコミュニティ分析機能の有効性や効率的な問い合わせ処理方法を述べてきた [6] [7].

著者らの提案システムは，「どのドメインから見て重要なコアか」という「FROM 型制約」と，「指定したドメインに関して周囲から見たときに重要なコアはどれか」という「TO 型制約」に基づいた多次元制約空間をデータキューブモデルと考えて，その制約下で Web コミュニティ問い合わせを行うデータベースシステムである．例えば「情報部門と電気部門の関係から見て活発な集団を知りたい」場合，事前に生成した Web マイニング用のデータキューブからスライスやロールアップ演算に相当する操作を行って，Web 空間構造分析でいうコア（完全 2 部グラフ）を多次元制約下で求め，それに基づいたコミュニティ構造をコアコミュニティグラフとして出力する．

昨年までに，著者らは文献 [6] [7] で，FROM 型と TO 型制約によるデータキューブ問い合わせ処理の基本演算体系を示してきた．ただし，データキューブのスキーマとなる制約条件として対象 Web 空間を 3 領域に直和分割した特定の場合のみ (大学ドメイン全体を 3 分割した) を用いていた．しかし，より一般的には，データキューブのスキーマとして対象空間の一部分だけを詳細に調べたい場合や,「キーワード “ロボット” にヒットするページから前方 N ホップ以内の空間を対象にする」などの一般的な制約条件から成る多次元制約空間を扱いたい．本稿では，このような一般的制約条件を許す場合について，提案システムの有効性を示す．

2. 多次元的な Web 空間マイニング

この節では，著者らが提案している多次元的な Web 空間マイニングシステムについて述べる．

2.1 準備 1: 多次元制約下のコア計算について

Web 構造マイニングの分野では，Web ページをノード，リンクをエッジとしたグラフにおける完全 2 部グラフをコアと呼び，コアに基づいた Web コミュニティ分析を行うことが多い．後述するように，コアは高頻度アイテムセットとして計算できる．一方で，著者らは，時間や場所などの多次元制約条件の下で行うログデータマイニングを，データキューブの枠組に沿って実行する「アイテムセットキューブ」という計算システムを提案し，対話的なログ分析で有効性を示してきた [5].

そこで我々は，適当な多次元制約の下でコア計算を行いたいという問い合わせを，アイテムセットキューブの時と同様に，データキューブモデルとして表し，データキューブに対応した演算体系を使って与えられた制約を満たす Web コミュニティ集合を求めるデータベースシステムを提案してきた [6] [7].

制約条件として，Web ページにおけるリンクの参照関係をもとに，どのドメインのページ集合からリンクが張られているかに着目した「FROM 型制約」と，どのドメインのページ集合に対しリンクを張っているかに着目した「TO 型制約」を考え，この 2 次元上で制約を与える．さらに,「いつの時点の Web 空間データを使うか」の時間軸を加えて，合計 3 次元のデータキューブモデルで制約空間を表す．

2.2 準備 2: コアの計算方法

本システムでは，コアの計算方法として，始点数 i 終点数 j のコア ((i,j)- コア) を，高頻度アイテムセットとして求める．そのため，まず Web リンク構造を表すデータとして，図 1 左側のように，各ページ r_s について，r_s へのリンク入力関係を表すレコード (リンクレコード) を用意する．つまり，r_s への全ての入方向リンクの始点ページを $i_1, i_2, ..., i_k, ...i_n$ として，これらをアイテムとしたレコード $[r_s, i_1, i_2, ..., i_k, ..., i_n]$([終点ページ, 始点ページ $i_1, i_2, ..., i_n$]) を作り，このレコード集合から高頻度アイテムセットを求める．

すると，$\{i_1, i_2, ..., i_k\}$ を k-アイテムセットと考えているためこれらを含むリンクレコード数が当該 k-アイテムセットのサポート数になる．その結果，計算するサポート数の最小値 (最小サポート数 s) を決めた時，リンクレコード集合 D において

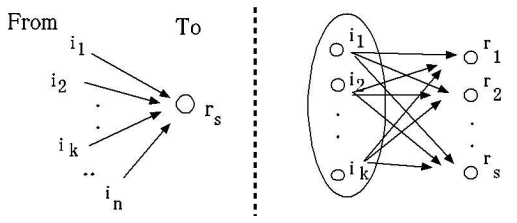

図 1　コアの計算方法

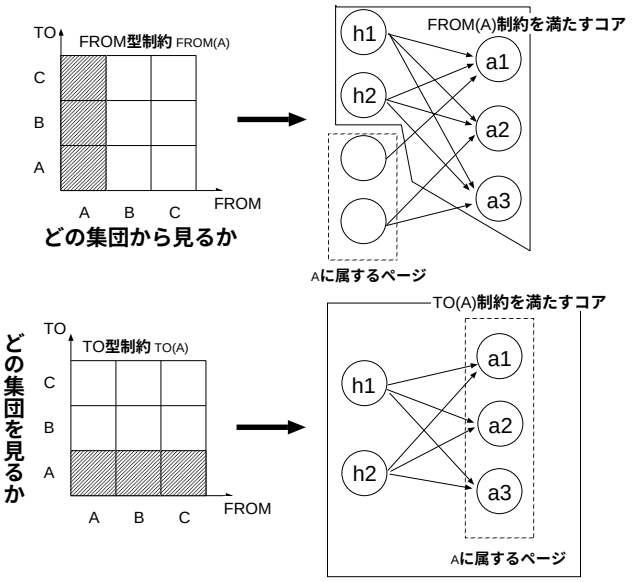

図 2　FROM 型制約と TO 型制約

高頻度 (サポート数 s 以上) となるアイテムセットを I とし，I を含む全てのレコードの終点ページ集合を A とすると，求まる高頻度アイテムセットは，I を始点集合，A を終点集合としたコアになっている．以下，コアの始点側ノードを，そのコアのハブ (hub) ノードと呼び，コアの終点側ノードを，オーソリティ (authority) ノードと呼ぶ．

2.3 多次元的制約に基づく Web マイニング問い合わせ機構

提案システムは，コア分析のために「FROM 型制約」と「TO 型制約」と呼ぶ 2 種類の制約条件を用意しており，これらの制約と時間軸の合計 3 次元のデータキューブモデルに基づいて多次元的な Web 空間分析を行う．

分析対象とする Web 空間は組織別の階層構造を持つ．例えば，電気通信大学（UEC）のトップドメインの下に，A:情報分野のドメイン，B:電気系ドメイン，C:その他のドメインがある．そこで，我々の研究では分析用の多次元制約として，これらのドメインにおける，誰にとって誰が重要であるかを分析するために,「FROM 型制約」と「TO 型制約」を定義した（図 2）．

FROM 型制約は,「どのドメインのページからリンクを張られているか」に着目した制約である．今，対象とする Web 空間を A，B，C の 3 つのサブドメインにわけて考える．このとき,「FROM 型制約を A に設定する」とは,「ドメインの A のページを少なくとも 1 つ始点に持つようなリンクレコードだけを対象にしてコアを求める」ことを意味する．こうして求まるコアを,「FROM(A) 制約を満たすコア」と呼ぶ．図 2 の上部

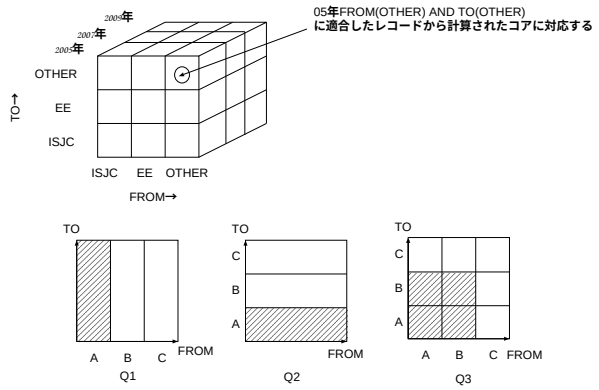

図 3　多次元制約下の問い合わせ例

に，その例を示す．FROM(A) 制約を満たすコアは，ドメイン A から見たときに重要なコアを表している．

一方，TO 型制約は「どのドメインページにリンクを張っているか」に着目したものである．「TO 型制約を A に設定する」とは，「ドメイン A のページを終点に持つようなリンクレコードだけを対象にしてコアを求める」ことを意味する．こうして求まるコアを，「TO(A) 制約を満たすコア」と呼ぶ．このコアは，ドメイン A に属するページだけを終点として持つコアである (図 2 の下部)．TO(A) 制約を満たすコアは，Web 空間全体からドメイン A を見たときに重要なコアを表している．

以上の結果，多次元制約として FROM 型制約と TO 型制約を与えると，この制約下の問い合わせは，図 3 のようなデータキューブモデルで表すことができる．例えば，図 3 の Q3 は，制約として，「FROM(A or B) And TO(A or B)」を表しており，問い合わせとしては，この制約を満たすコア集合を求め，それらに基づいたコミュニティ集合を求めたいという要求である．

この要求に答えるため，本システムでは，利用者が多次元制約として FROM 型制約や TO 型制約を与えると，データキューブによる多次元制約下のコア計算を行った後に，そのコアを極大コアに直してその集合からコアコミュニティグラフと呼ぶコミュニティ間の関連性を表すグラフを作成しランク付けして答えを返す．

コアコミュニティグラフとは，与えられた FROM 型制約/TO 型制約下で求まった極大コア集合において，共通する authority ノードを 2 つ以上持つ極大コアを 1 つの同値類として 1 ノード化 (「コアコミュニティノード」と呼ぶ) し，2 ノード間の有向辺を，当該ノードに含まれるページ間のリンクに基づいて与えて作った有向グラフである．また，このグラフ構造上で PageRank に似せたランク計算を行い，重要なコアコミュニティノードを判定する．図 4 に，2005 年の対象空間全体 (uec.ac.jp 内の約 10 万ページ) について計算したコアコミュニティグラフと上位ランクのノード順位を示す．

一般的には，データキューブのスキーマを決めると，それに基づいて「A または B から見て重要なコミュニティを求めよ」という FROM(A or B) や，「周囲から A または B を見て重要なコミュニティを求めよ」といった TO(A or B) のように多様な問い合わせが考えられる．図 3 の Q3 が表す「FROM(A or B) And TO(A or B)」は，「A と B の関係性から見て重要なコミュニティを求めよ」という意味に相当する．また，コアの計算では，計算に用いる内部リンク (同一サイト内のリンク) 数や，計算対象とするコアの大きさ (始点数と終点数の範囲) に制限をつけなければ実質的ではない．事前にあらゆる制約やパラメータ値についてコアを計算しておくことは現実的ではない．従って，本システムの技術上の課題は，事前にどのようなデータキューブを計算しておき (実体化処理)，それらを使った代数演算 (スライスやロールアップに相当する) を使って，与えられた問い合わせに答えることが効率的か，を示すことである．

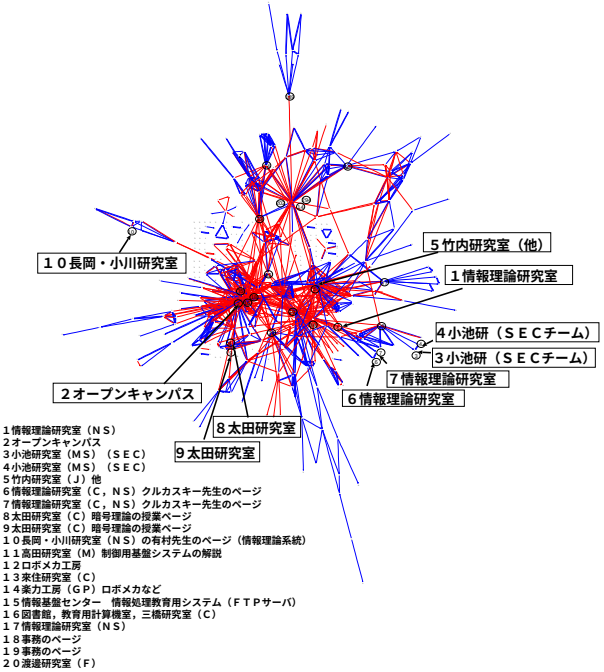

図 4　05 年 b=8,s=4 制約なしのコアコミュニティグラフ

2.4　処理方法の概要

文献 [6] [7] では，電気通信大学全体のドメイン (1:ALL) の階層構造を，情報部門 (2:ISJC), 電気部門 (3:EE), その他部門 (4:OTEHR) にわけて，この多次元制約スキーマに対して FROM 型/TO 型制約問い合わせと，それを組み合わせた複合問い合わせの処理方法を提案した (図 5)．

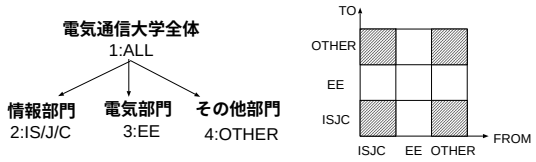

図 5　これまでのスキーマ階層と問い合わせ例

代数演算としては，あらかじめ計算されたコア集合から FROM 型制約か TO 型制約を満たすコア集合を抽出する処理方法 (スライス演算に相当) としてフィルタリング法を持つ．また，ロールアップに相当する処理として，マージ法と呼ぶ演算を提案している．マージ法は，FROM(A) 制約 (または TO(A) 制約)) を満たすコア集合と FROM(B) 制約 (または TO(B) 制約) を満たすコア集合から FROM(A or B)(または TO(A or

B)) を満たすコア集合を差分コア計算により求める演算である．

事前に用意すべきデータキューブとしては，図 6 に示す 3 つの Web マイニング用データキューブを用意した．(ここで，パラメータ b は，ページの入辺が全て内部リンクで総数 b 以上のときに当該内部リンクを削除することを意味し，コア計算の妨げとなる内部リンクの削除数を決める下限閾値である．また，s は極大コアを計算するときの最小サポート数である(注1))：

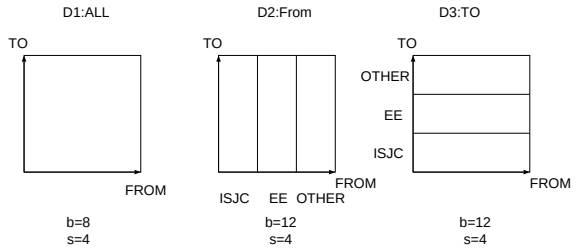

図 6　事前に実体化しておくキューブ (昨年まで)

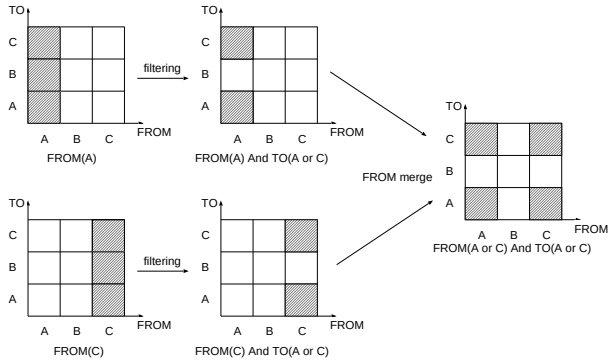

図 7　複合問い合わせを表す演算木

D1: FROM 型/TO 型制約無しで実体化を行い，極大コアを格納したデータキューブ (b=8, s=4).

D2: 制約を FROM(ISJC), FROM(EE), FROM(OTHER) 各々とした場合に実体化を行い，各場合の極大コアを格納したデータキューブ b=12, s=4).

D3: 制約を TO(ISJC), TO(EE), TO(OTHER) 各々とした場合に実体化を行い，各場合の極大コアを格納したデータキューブ (b=12, s=4).

データキューブ D1 は，全体を大まかに計算したい場合に用い，フィルタリング法によって各問い合わせに答える．D2 や D3 は，より詳細なコア計算を求める問い合わせを実行するときに用いる．例えば，複合問い合わせとして「FROM(A or B) And TO(A or B)」を $b = 12, s = 4$ で計算するときは，図 7 のように，TO 型のフィルタリング法と FROM 型のマージ法を組み合わせて実行する．

以上をまとめると本システムは，基本 DB 演算として直接実体化演算，FROM 型制約と TO 型制約各々におけるフィルタリング演算とマージ演算，グラフ作成とランキング処理，類似度比較演算（Jaccard Similarity Join）を持ち，これらを組合わせた問い合わせ木を作って実行する Web マイニング用データベースシステムである．

(注1)：b で枝刈り後，コアの始点数 2, 終点数 4 以上を条件に Pruning 処理 [1] してから Apriori でコア計算している．

3. 制約条件の一般化

3.1 一般化された制約条件への対応

データキューブとしてより利用価値を出すためには，これまでのスキーマ階層の場合のみではなく，一般化された制約条件に基づいた多次元制約を扱うことが必要である．ここで，一般化された制約条件とは「ある特定部門についてより詳細に知りたい」や,「指定キーワード K を満たすページから前方 N ホップ以内にある」というような制約条件のことである．正確に言うと，ページに関するブール述語 $p_i(i = 1, 2, \ldots)$ を与えた時に，p_i を満たすページを始点 (または終点) に持つリンクレコードを使って, 図 2 の場合と同様に FROM(p_i) 制約 (または TO(p_i) 制約) を満たすコアを定義し，これによって多次元制約下のコア計算を行う．

図 8 右は，上記のような一般的制約条件 p_5, p_6, p_7, p_8 を対象とした FROM 型制約と TO 型制約から構成されたデータキューブのスキーマである．

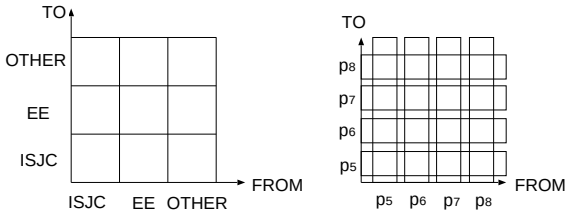

図 8　一般制約条件に対応した問い合わせ例

注意すべき点は，これらの制約は，対象空間全体 (ALL) を被覆しないし，直和分割にもなっていない，ということである．また，これらの一般的制約条件を事前に予想することはできないから，利用者が実行時に与えることが想定されるし，b, s といったコアの細かさもより厳しい条件になる．例えば，b=16 や 20 では空間全体 (ALL) や FROM(OTHER) のコアは計算コストが高過ぎて予め用意できないか用意することが現実的でないが，一般的制約条件なら可能となる場合である．

前回のスキーマでは，$b = 12, s = 4$ で用意した D2, D3 をより詳細なコミュニティ構造を求める場合に用いていた．しかし，実際には，D2 は空間全体を被覆しているので，必要なコアは必ず FROM(X) (X=ISJC, EE, OTHER) のどれかに含まれるから，問い合わせごとにフィルタリング演算をこれらに適用すれば求めることはできる．つまり，必ずしもマージ演算を使う必要はない．

一方，今回の図 8 右のような一般制約条件の場合,「FROM (p_5 or p_6) And TO(p_5 or p_6)」の問い合わせを行うには，図 7 のような複合問い合わせ処理が必要である．今までのフィルタリング演算やマージ演算の処理方法は，一般的制約条件でも成

立する(注2).

したがって，問題となるのは，一般的制約条件を許したときに，データキューブとして何を，どの詳細度に応じて用意するか，というデータキューブとしての維持戦略である．

3.2 新しく用意するデータキューブと詳細度

一般的制約の例として，制約条件 p_5，制約条件 p_6，制約条件 p_7，制約条件 p_8 についてそれぞれを IS(情報システム学部門) のみ，J(計算機科学部門) のみ，C(情報通信部門) のみ，M(機械部門) のみという制約条件に設定し，現在のシステムでどこまで詳細度を細かくできるか調べた．

その結果，C については b=14 まで，J,M については $b = 18$ まで，IS については b=20 まで細かくしても FROM 型制約，TO 型制約で求めることができた．

例として，b=16,s=4 FROM(IS) の例 ('05 年の uec.ac.jp 下) を図 9 に示す．

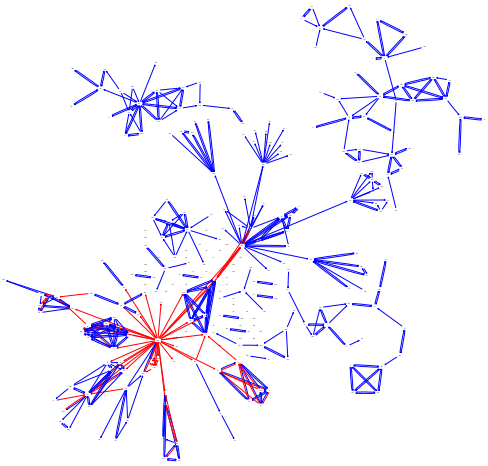

図 9　b=16,s=4 FROM(IS) のコアコミュニティグラフ

このように，今まで使えなかった「ある特定部門についてより詳細に知りたい」といった一般的制約が使えるようになった．FROM(ISJC) では b=12 までしか求めることができなかったが,「IS のみ」ではコアの細かさ (詳細度)b をさらに深くしてより細かい分析も可能となる．

次に，これらを利用して，一般的制約条件による複合問い合わせを行う場合を考える．利用者の想定する制約条件に応じて必要最小限のデータキューブのみを実体化して，そこから要求に応じた問い合わせに答えるという方針に基づいて事前に用意するキューブを決める必要がある．新しく考える実体化範囲の案として，b=8 での電気通信大学全体 (ALL)，b=12 での FROM(ISJC), FROM(EE), FROM(OTHER) のみを常時持っておき，b=14 から b=20 程度の極めて詳細なコア計算を行いたいときは一般的制約条件を利用して「IS のみ」のようなデータキューブを随時生成/更新するという戦略がある (図 10)．すなわち，図 10 では，一般的制約条件 p_5, p_6, p_7, p_8 の 4 つについて $b = 16, s = 4$ で各々FROM 型制約を満たすコア集合を保持するデータキューブとして D4 を維持し，同じく TO 型制約を各述語に応じて計算し保持するキューブとして D5 を維持する．一般化制約述語の変更にあわせて，D4 と D5 は適宜更新して良い．

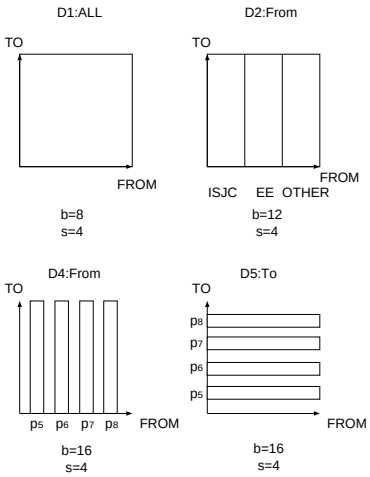

図 10　実体化するキューブの案

また，今まで用意していた b=12 の TO 型制約用の D3 は，D2 から計算できるため今回は省いている． $b = 16$ で FROM(IS) 制約と FROM(J) 制約を D4 として実体化した後に「IS と J の関係性についてより詳細に (b=16 で) 知りたい」という問い合わせが与えられた時は, D4 の FROM(IS) と FROM(J) からフィルタリング法とマージ法の組み合わせによる複合問い合わせを用いて答えることができる(注3).

残る問題は，一般化制約述語によって求まる情報の品質，上記の戦略で維持されるデータキューブの記憶量，および，複合問い合わせ処理木の計算時間を調べ，問い合わせごとの直接計算よりも効率的であることを示すことである．

4. 評　　価

一般化制約述語の例として,「特定のドメインをより詳細にみる場合」を扱う．制約条件 p_5，p_6 についてそれぞれを IS(情報システム学部門) のみ，J(計算機科学部門) のみという制約条件に設定し，コアの詳細度を b=16, s=4 に固定した場合のコアコミュニティの品質，実体化したキューブの記憶コスト，実体化する際の処理時間，について評価実験を行った．実験環境として，CPU:2.66GHz，メモリ:3GB のマシンを用いており，対象の Web 空間として 2005 年の電気通信大学全体のデータ (URL 数:108235 ページ) で実験を行った．

4.1 コアコミュニティの品質評価

b=8 と b=16 の FROM(IS or J) And TO(IS or J) の結果に基づき，上位 20 位を jaccard 類似度 (4%以上) で比較した．b=8 と b=16 の順位を比較した対応表を図 11 に示す．

(注2)：文献 [6] の 6 節, [7] の 5, 6 節のアルゴリズムにおいて，A ドメイン，B ドメインを各々p_5, p_6 に置き換えても成立する．

(注3)：例えば，FROM 型マージ法のアルゴリズムでは，差分コアの終点ノードに制約を付けても成立するため図 7 のような複合問い合わせにも対応できる．また，始点数に上限をつけても成立するためコア計算方法を変えた場合にも適用できる．

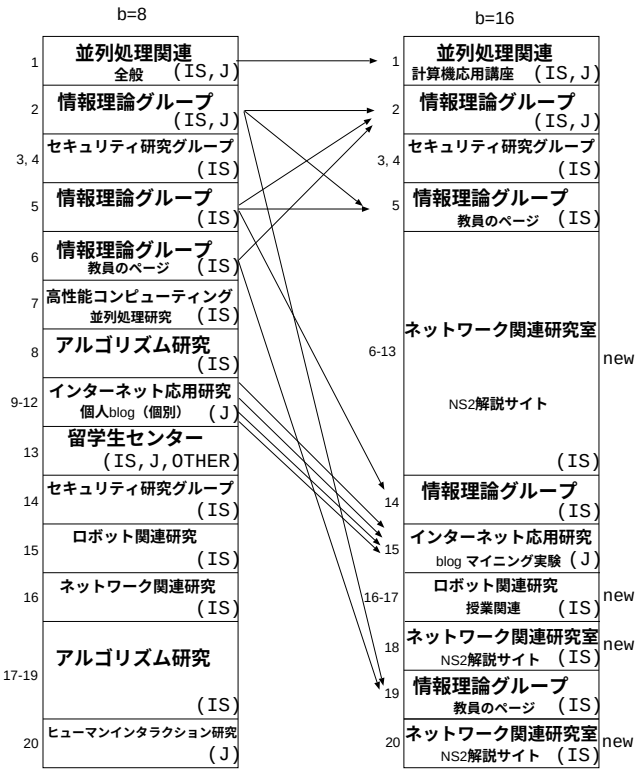

図 11　b=8 と b=16 の jaccard 類似度 (4%) による比較

1 位のサイトについて，b=8 の場合，並列関連ページが集まったコアコミュニティノードであったが，b=16 で見た時，その中のある特定の研究室サイトが目立って現れていることに気づくことができた．

また，b=16 の 6 位から 13 位にネットワークシミュレータの解説サイトが出てきている．これは b=8 のとき 85 位以降に出てくるノードである．

b=16 で詳細度をあげてみた場合に 15 位に出てきたコアコミュニティノードは blog マイニングに使われる実験用の blog サイトであった．これは，b=8 の 9 位から 12 位に出ていた個人の個別 blog と対応していることが分かった．

これらのことから，図 7 のように特定の制約間の関係を求める複合問い合わせについて，b=8 のみで大まかにみる場合だけではなく，b=16 を用いてより詳細にみることで，細かいコミュニティ分析が行えることを示した．

4.2　キューブの記憶量とマージ演算のコスト

まず，D4 と D5 に必要な記憶量を表 1 に示す．次に，D4 と D5 の上で行われる FROM 型マージ，TO 型マージの問合わせとして，FROM(IS or J)，及び，TO(IS or J) について，その $b = 16$ のときの計算時間 (秒) を表 2 と表 3 に示す．比較対象として，マージ法を使わずに直接コア計算を行った場合 (直接実体化) の処理時間も示した．各表の「差分コア」の欄は，マージ法による差分コア計算の時間であり,「後処理」欄は，極大コア集合からコミュニティノードを求めてグラフ構造作成，ランク計算，の総時間である．表 4 に，各マージ法実行時の差分コア数を示す．これらの結果から分かるように，一般的制約条件で選択レコード数が減ると差分コアが少なくなるため，毎回 FROM(IS or J) のような直接再実体化を行うよりも，マージ法の使用が効率的である．

表 1　D4, D5 に対応した記憶量 (b=16)

問合わせ	コア数	ページ数	ノード数
FROM(IS)	1933	20282	264
TO(IS)	1930	20256	263
FROM(J)	2338	24679	169
TO(J)	2334	24645	166

表 2　FROM(IS or J) の処理時間（秒）(b=16)

内訳	直接再実体化（秒）	マージ法 (秒)
極大コア	69	-
差分コア	-	9
後処理	10	11
合計	79	20

表 3　TO(IS or J) の処理時間（秒）(b=16)

内訳	直接再実体化	マージ法
極大コア	69	-
差分コア	-	1
後処理	10	10
合計	79	11

表 4　FROM 型マージ，TO 型マージ問合わせの記憶量 (b=16)

問合わせ	コア数	ページ数	ノード数
FROM(IS or J)	4274	44980	435
FROM(IS or J) の差分	3	19	-
TO(IS or J)	4267	44922	430
TO(IS or J) の差分	3	21	-

4.3　複合問合わせの処理時間

次に，複合問合わせである FROM(IS or J) And TO(IS or J) について b=16 で記憶量と計算時間を求めた．計算結果におけるコア数，ページ数，ノード数を表 5 に示す．複合問合わせには，TO 型フィルタリングしてから FROM 型マージを行う方法 (方法 1) と，FROM 型フィルタリングしてから TO 型マージを行う方法（方法 2) の 2 種類がある．

表 5　複合問合わせの記憶量 (b=16)

問合わせ	コア数	ページ数	ノード数
FROM(IS or J) And TO(IS or J)	4267	44922	430
TO フィルタ+ FROM マージの差分	1	7	-
FROM フィルタ+ TO マージの差分	3	21	-

表 5 に示すように，必要な際に差分を求め，既に実体化されている単一制約条件と組合わせて複合問合わせを行えば，b=16 の FROM(IS or J) And TO(IS or J) を事前に実体化して持っ

ておくという記憶コストが無くなる．

複合問い合わせ時の方法 1 と 2 の処理時間を，表 6 に示す．直接再実体化ではなくマージ法を用いた複合問合わせを行うほうが処理時間が短くて済む．

表 6　FROM(IS or J) And TO(IS or J) の処理時間（秒）(b=16)

内訳	直接再実体化	方法 1	方法 2
極大コア	70	-	-
差分コア	-	8	7
フィルタリング A	-	5	5
フィルタリング B	-	5	5
後処理	11	11	11
合計	81	29	28

4.4　問い合わせ処理木の例

本システムで図 12 の左のようなスキーマ構造を与えて，図 12 右のような図 10 の D4 に対応するキューブを作り，その上でいくつかの問い合わせ例を実行し，得られる情報の内容と実行時間を調べた．

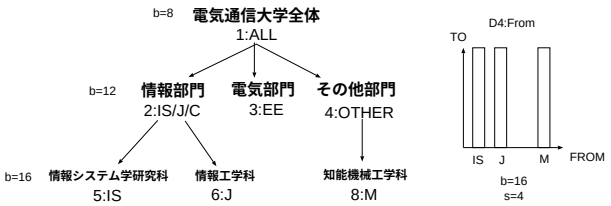

図 12　今回の例で使用するスキーマ構造

4.4.1　Q1:IS と M の関係性で重要なコミュニティ

問い合わせ例 Q1 として「IS と M の関係性を見たときに重要なコミュニティは何か」という問い合わせを与えられたとき，複合問い合わせ FROM(IS or M) And TO(IS or M) を行うことでこの問い合わせに応答することができる．この例の問い合わせ木を図 13 に示す．

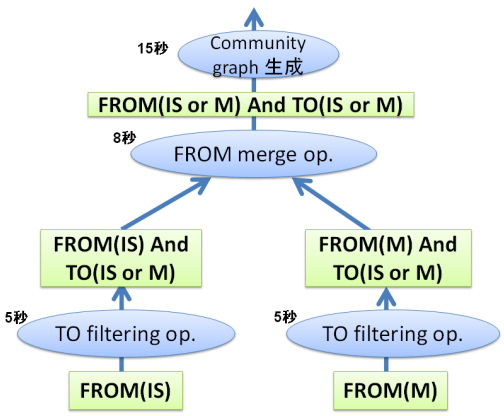

図 13　問い合わせ例 Q1 に対応する問い合わせ木

結果のコミュニティグラフを図 14 に示す．

また，単一制約で問い合わせを行なった場合と複合問い合わせで関係性を見た場合の違いをみるために，Jaccard 類似度 (4%) で FROM(IS), FROM(M) ,FROM(IS or M) And TO(IS or M) を比較した．

上位 20 位の分析結果を図 15 に示す．

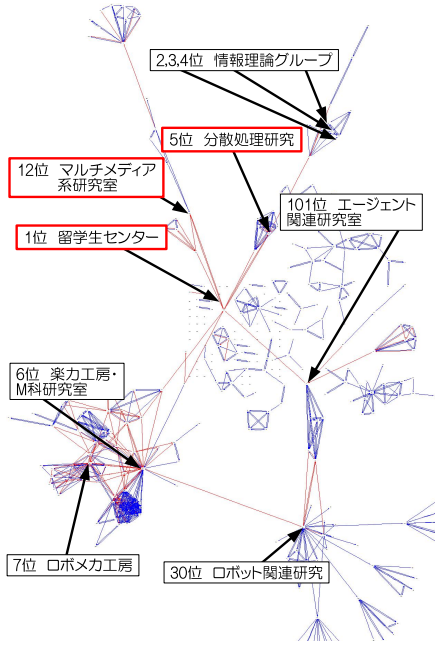

図 14　b=16, FROM(IS or M) And TO(IS or M) のコミュニティ

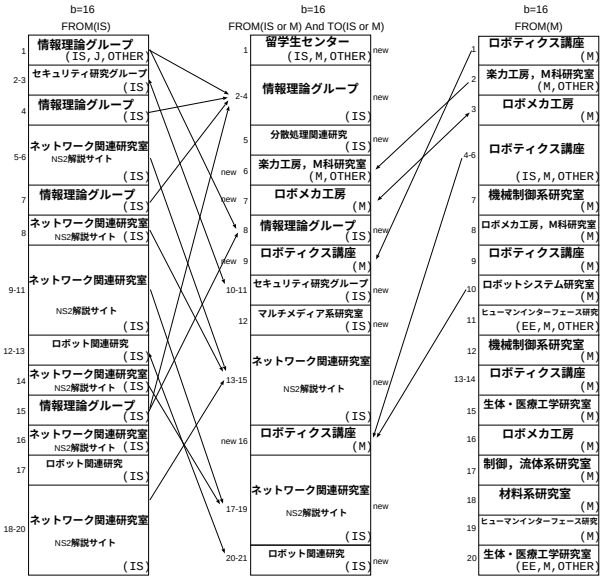

図 15　b=16 FROM(IS or M) And FROM(IS or M) の上位 20 位比較

FROM(IS or M) And TO(IS or M) の中に，FROM(IS) のコミュニティが多く出現している．複合問い合わせの 1 位である留学生センターは，FROM(IS) の 115 位と 4%で類似するノードであり，IS と M の関係性でみたときにとても目立つコミュニティである．FROM(M) の 20 位の中に留学生センターのページがコアに含まれているが，1 位のコミュニティとは対応しない．

FROM(M) でみると 1 位がロボティクス講座，2 位が楽力工

房，3 位がロボメカ工房であるが，IS と M の関係性でみたときには順位が逆転しており，楽力工房とロボメカ工房の順位が上がっている．このことから，楽力工房，ロボメカ工房が IS と M の関係性でみた場合により重要なコミュニティであるということが分かる．この他，複合問い合わせの 5 位である分散処理関連研究は，FROM(IS) で見たときの 25 位に完全一致する．また，12 位のマルチメディア系研究室は FROM(IS) の 115 位と完全一致するため，これらは IS と M の関係性でみたときに順位があがったコミュニティであることがわかる．

4.4.2 Q2:FROM(IS or J) top40 に含まれる FROM(IS) top40 は何か

問い合わせ例 Q2 として,「IS と J からなるコミュニティtop40 を見たときに，IS の top40 に対応しているものはどれか」というような，興味のある情報を提供する問い合わせに対して次の問い合わせ木を作り応答する．この例における問い合わせ木を図 16 に示す．

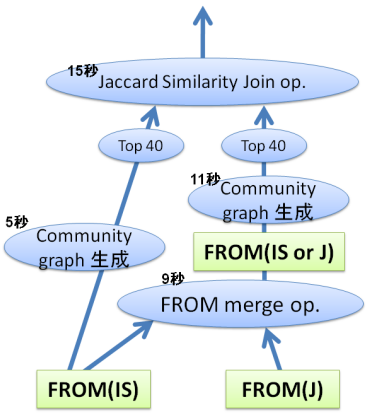

図 16　問い合わせ Q2 に対応する問い合わせ木

問い合わせ例の結果として，FROM(IS or J) の top40 のうち FROM(IS)top40 に属するノードは 32 ノード出てきており，8 ノードが消えていることが分かった．消えているノードは 17, 23, 26, 34, 36, 37, 39, 40 位であるが，これらは無くなったわけではなく top40 以下には全て存在している．また，FROM(IS or J) のコアコミュニティの中に FROM(IS) のノードが split や merge されて出てきている様子もみることができた．

このように，本システムでは既に実体化されたキューブを組合わせることで 40 秒程度で「IS と J からなるコミュニティ top40 を見たときに，IS の top40 に対応しているものはどれか」というような問い合わせを行うコミュニティ分析を行うことができる．

この他の例として,「あるキーワード周辺の関連コミュニティを知りたい」という場合についても図 12 とは別のスキーマを用いた場合でも同様の評価を行っている [8].

5. おわりに

本稿では，著者らが提案してきたデータキューブモデルに基づいた多次元制約下の Web 空間マイニング向けデータベースシステムについて，従来のおおまかなドメイン別だけでなく，一般的な制約を許す場合を対象にデータキューブの実体化維持方法と複合問い合わせの処理方法を述べた．一般的制約条件の例として今回新たに許した制約は，大学ドメインの特定の狭い Web 空間領域を表すもの (「IS のみ」「J のみ」など) と，指定キーワードを含む Seed ページから前方 N ホップ以内の空間」などである．想定した利用方法は，このような一般化制約条件を使った多次元制約問い合わせで，かつ，より内部リンクを考慮して詳細なコアを求めてコミュニティ計算したい場合である．

提案したデータキューブの維持戦略は，一般化制約条件に対応して FROM 型制約を満たすコア集合と TO 型制約を満たすそれを別々にデータキューブとして維持し，条件の変更に対応してこれらのキューブを更新するものである．問い合わせ処理方法は，従来から提案しているフィルタリング演算とマージ演算，Jaccard Similarity Join などの基本演算からなる問い合わせ木を作って実行する．本稿では，特定組織ドメイン内の Web 空間を対象に，この手法により得られるコミュニティ分析能力の有効性を示してきた．例えば，FROM(IS or J) And TO(IS or J) の b=8 と b=16 を比較し，詳細ドメインでみた場合にある特定のコミュニティが目立って現れていることが分かった．さらに，コミュニティ分析を行う具体的な問い合わせ木の例を提示し，Jaccard Similarity Join を用いて「IS と J からみた際に IS からみたコミュニティがどのように split, merge されてでているか」といったコミュニティ分析が 40 秒程度で行えることを示した．

以上により，問い合わせ処理時間が直接再計算によりも提案したマージ演算や複合問い合わせ処理手法が効率的であることを示し，本システムである特定の空間を自分の知りたい空間にパーソナライズして様々なコミュニティ分析が有用に行えることを示した．

文　献

[1] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, Andrew Tomkins, “Trawling the Web for emerging cyber-communities,” WWW8/Computer Networks, Vol.31(11-16), pp.1481–1493, 1999.

[2] 豊田 正史, 吉田 聡, 喜連川 優, “ウェブコミュニティチャート: 膨大なウェブページを関連する話題を通して閲覧可能にするツール,” 電子情報通信学会論文誌, D-1 Vol. J87-D-1 No.2, pp.256–265, 2004.

[3] S.Raghavan, H.Garcia-Molina, “Complex Queries over Web Repositories,” VLDB 2003, pp.33-44, 2003.

[4] Pedro DeRose, Warren Shen, Fei Chen, AnHai Doan, Raghu Ramakrishnan, “Building Structured Web Community Portals: A Top-Down, Compositional, and Incremental Approach,” VLDB 2007, pp.399-410, 2007.

[5] T.Ohmori, M.Naruse, M.Hoshi, “A New Data Cube for Integrating Data Mining and OLAP,” ICDE Workshop on Data Mining and Business Intelligence, paper ID 3, IEEE, 2007.

[6] 栗原 大輔, 大森 匡, 星 守, “Web 構造分析を目的とした多次元データマイニング構造の効率化”, DEWS2008, D1-6, 2008.

[7] 張 洪鋒, 大森 匡, 星 守, “Web 構造分析を目的とした多次元データマイニング機構の効率化：To 型制約問い合わせの処理方法,” DEIM フォーラム 2009, E7-3, 2009.

[8] 齋藤 太陽, “多次元的な Web 空間マイニングを行うデータベースシステムの実現：分析条件一般化への対応,” 電気通信大学大学院情報システム学研究科 2009 年度修士論文, 2010.