学位請求論文

分散した情報資源の統合的利用法に関する研究

1995 年 12 月 20 日 提出

指導教官　安達 淳　教授

東京大学大学院工学系研究科
電気工学専攻　37097

西澤 格

分散した情報資源の統合的利用法に関する研究

西澤 格

A Study of the Integrated Utilization Method for Distributed Information Resources.

By Itaru NISHIZAWA

Recent growth of high speed computer networks in its scale and performance, and the increasing affluence of information have made centralized data processing difficult. Therefore, one of the greatest concern of users has come to how to obtain useful information from enormous information scattered on the networks. The term "Integrated Utilization Method" in this research means that users can access to multiple information resources (IRs) without concerning their locations and heterogeneities. There exist two essential problems we must solve before we can handle multiple IRs with integrated method. One is the organization problem of IRs and the other is the heterogeneity problem among IRs.

We must first consider the model of information space to discuss the organization problem of IRs. We can classify the information space organization models into two categories. One is an index type and the other is a navigation type. But no information systems based on these models satisfies users' demands for now. Next, we consider the heterogeneity of IRs. We denote each database to which we want to make integrated access as a component database. The conventional schema integration method makes global schema among component databases in the schema integration process if the user want a mechanism of systematic retrieval. But this approach causes problems because the sets of the component databases change dynamically in accordance with user's query.

To solve the above-mentioned problems, we dynamically construct a virtual database (VDB) according to the user's query. On the VDB, there exists a consistent virtual functional dependency (VFD) that is calculated by the component databases' FDs. Using VFD, we can integrate component databases with consistency. Furthermore, the size of answers obtained by our method is larger than that of obtained by the conventional query processing method. We also proved that our algorithm is sound and complete when information among the component databases is consistent. The assumption is reasonable in the process of the integrated access to the component databases. On the effectiveness of the query processing, sound and complete answers can be efficiently calculated when the VFD formed in the VDB is acyclic among the nodes that the query refers to. If the VFD is cyclic, this is the case of calculating the transitive closure of functional dependencies, and the cost would be large. But, as the system can detect a loop of VFD before the query is processed, the user can choose the strategy for the query processing either to get a correct answer or to get all answers with large calculation.

We have already mentioned that the size of the answers obtained by our method is larger than that of obtained by the conventional query processing method. Then, the next interest is the size of the answers obtained by our method compared with the conventional method. We investigated the size of the answers by computer simulation and verified its validity. Furthermore, we develop a bibliographic information retrieval system employing the schema integration method described herein such that integrated access is obtained for bibliographic information written in BibTeX and verified its validity with practical data.

As concluding remarks, we investigated IRs' heterogeneity and organization problems more precisely so as to deal with more general purposes, and examined a database clustering method to support the IRs' selection problem and the attributes' mapping problem that appear in the database integration process. Finally, we proposed a system DIRECTOR which solves the problem of heterogeneity and organization, and realizes the integrated access to the multiple IRs.

目 次

図目次

表目次

第１章

はじめに

本研究および本論文を概観する.

1.1 研究の背景

近年，コンピュータネットワークの高速化の流れは目覚しいものがあり，その規模も爆発的に大きくなっている．これに伴い，世界中の主要な研究機関，企業，大学等はネットワークで接続され，各機関内の計算機は高速のLANで結ばれているというのが一般的になりつつあり，現在ではさまざまな人々にとって，電子メールを始めとするネットワークサービスは必要不可欠なものになっている．このネットワークの発展は各要素技術の発展がその原動力となっていることはいうまでもないが，アメリカ合州国の情報ハイウェイ構想を筆頭とした，ネットワークを情報システムのインフラストラクチャとみなした行政側の援助も見逃すことができない大きな要因であろう．

ネットワークの高速化，規模の増大とは別に，計算機自身の価格性能比もすさまじい勢いで向上している．またメモリについてもその集積度は順調に上がり続けており，磁気ディスク装置についても大容量小型化，高速化が進んでいる．このように高速なCPU，大容量で高速のメモリ，磁気ディスク装置を備えた高性能・低価格な計算機が気軽に使用できるようになってきた．これらの計算機はネットワーク接続機構を標準装備している場合がほとんどであり，このため研究者のみならず一般の人々のネットワークに対する関心も大きくなってきていた．それに加えて1995年にはインターネットという言葉がマスコミで大きくとりあげられ，前述の安価な計算機上で稼働する，ネットワーク接続をサポートしたオペレーティングシステムが発売され，全世界的に驚くべき売上を示した．さらに，ネットワークプロバイダと呼ばれるインターネットへの接続をサービスする企業が数多く設立されたこともあって，従来ネットワークを研究目的で使用してきた研究者のみならず，広く一般の人々もインターネットに参加できるようになった．

さらに，1995年はWorld Wide Web (WWW) と呼ばれる分散情報システムが爆発的に普及した年でもあった．現在では全世界的に，大きな企業，大学の各研究室のほとんどがWWWのサーバを立ちあげており，個人的なインターネット参加者たちも自分のホームページという形でネットワークに情報を発信することが可能となっている．このためネットワーク上の情報は指数関数的に増加し，様々な情報が錯綜することとなった．いかに大容量の記憶装置をもってしてもこれらの増え続ける情報を集中的に管理することは現実的には不可能であり，この分散する情報資源の中からいかにして自分に有用な情報を得るかということはネットワークを利用している人々にとって一番の関心事となってきている．

ネットワークにおける情報の利用という目的で，現在様々な情報システムが稼働している．例えば前述のWWWを始めとして，Gopher，WAIS，Archie等をその代表としてあげることができる．これらのシステムは広く利用されているが，現状ではこれらのシステムを用いて実用的な情報を体系的に獲得するのは難しい．これはGopherやWAIS, Archieは予め作成した索引に大きな影響を受けること，WWWは基本的に発見的にしか情報を得ることができないことがその主な原因である．発見的な情報検索とは情報の大海をさまよっているうちに偶然有用な情報を見つけるというような意味で，Mosaic等のブラウザでWWW空間を眺めることが広く「ネットサーフィン (net surfing)」と呼ばれていることはよくその実体を表しているといえる．

1.2 研究の目的と意義

本研究における「情報資源の統合的利用」とは究極的には分散する複数の情報資源に対して，ユーザがその位置，異種性を意識せずにアクセスできるということを意味する．具体的にいうと，ユーザが複数の情報資源が存在することを意識せずに問合せを発行すれば，何らかのメカニズムによって実際は複数の情報源からその答えが集められユーザに示されるというものである．このように複数の情報資源を統合的に取り扱おうとする際には，解決しなければならない大きな二つの問題がある．一つは情報資源の管理の仕方とそれに対するアクセス方法に関する問題，そしてもう一つは各情報資源の間に存在するさまざまな異種性に関する問題である．

情報資源の管理の仕方とアクセス方法について議論するにあたり，まず最初に情報空間の管理のモデルを考慮する必要がある．現在の情報システムでは情報空間の管理モデルは大きく二つに分類することができる．一つは索引型でそしてもう一つは航行型である．索引型のシステムは体系的な検索が可能であるという長所を持っているが，サーバが必要となり，ネットワークの巨大化によって情報量が激増している今日では，検索時のサーバへのアクセスの集中や，管理する情報のメンテナンスなど，情報量の大きさに起因する問題は無視できないものとなるという問題点を抱えている．一方，航行型のシステムは，検索時のサーバへのアクセスの集中の問題，管理する情報のメンテナンスなどの問題を回避できるという長所を持っているが，ユーザが情報の存在する場所を認識しておく必要があることと，(現在のシステムでは) 情報発見という色彩が強く，実用的な情報検索を行うのは難しいという問題点がある．ところが，実際に情報資源を探索する際には，システムから情報資源へのアクセスはユーザから見えないこと，つまり情報の位置透明性が確立されることと，それらの情報に対して，ブラウジングのみならず体系的な検索が可能であることが要求される．またシステム的な観点からはある特定のサーバに負荷が集中するのは好ましいことではない．これらの要求を満たすためには上記のモデルの直接の適用ではうまくいかない．

次に情報資源の異種性の問題について考える．異種性という言葉について考えると，情報の表現の異種性，情報の構造の異種性などの様々なレベルの異種性が存在することがわかる．複数の情報源に対する統合的な問合せ処理は，この異種性の存在から一般には困難な問題であるが，異種データベースの研究分野においていくつか議論がなされている．異種データベースの研究分野においては，データベースの異種性の中で，本論文で取り扱う主要な異種性を (データベース) スキーマの異種性と呼び，これらのスキーマの異種性を克服するために複数のスキーマを統合することをスキーマ統合と呼ぶ．また，本論文における個々の情報源を要素データベースと呼ぶ．複数の異なるスキーマの統合においては，各要素データベースのスキーマおよびデータの一貫性を損なわないような工夫が必要とされるが，従来の研究では大域スキーマ，連邦スキーマなどと呼ばれる大域的な統合スキーマを作成する手法が一般的であり，これらは静的に構成される場合が多い．このアプローチは対象とするデータベースが確定している場合には問題ないが，1.1節で述べたように変化し続ける情報を取り扱おうとする際には問題が起こる．

1.3 研究の概要

1.2節で述べた問題を解決するために，本研究ではユーザの問合せに対して動的に仮想データベース (virtual database) を構成し，ユーザからの問合せを処理する．この仮想データベース上では各要素データベース上で成り立つ関係である関数従属性を利用して，全体として矛盾の無い仮想関数従属性 (virtual functional dependency) を計算する．この仮想関数従属性の情報を用いることによって，各要素データベースを矛盾無く統合することができる．さらに，本論文で議論する問合せ処理手法は，同時に複数の要素データベースを取り扱うことによって，それらに独立にアクセスを行う場合よりもさらに多くの情報を得ることができるという特徴がある．関係データベースシステムを対象としたアプローチで，各タプルについてインスタンスレベルでのルールを与え，データの実体の同一性問題を解決することによってデータの一貫性を保持しようとする研究 [40] も行われているが，一般的にデータベースの統合過程では対象とするインスタンスは膨大な数となってしまうために，実用性に欠ける．それに対して本研究における手法は，タプルの同一性問題については各要素データベース上におけるスキーマレベルでの一貫性を考え，空値の考え方を導入することにより，実用的なスキーマ統合の手法となっている．

問合せ処理を行う際には，解の正しさと完全性，および計算量 (効率) の問題にも注意をはらう必要がある．まず解の正しさと完全性については，統合を行う要素データベース間でその情報に矛盾が無い場合には，本論文で述べる問合せ処理手法は正しくかつ完全であることを証明した．各要素データベース間でその情報に矛盾が無い場合という条件は，統合アクセスを行う際には妥当な仮定であると考えられる．また，問合せ処理の効率については前述の仮想関数従属性が閉路を作らない場合には，多項式時間での計算で解を求め得るアルゴリズムを示した．もし仮想関数従属性が閉路を作る場合には推移閉包を求めるという高価な処理を行う必要があるが，閉路を作るか否かは問合せ処理を行う前に検知できるため，ユーザは全ての解を求めるのかあるいは一つ解を求めればいいのかというような，問合せを行う目的によって問合せ処理のアルゴリズムを切替えることにより対処することができる．

本研究における問合せ処理では各々のデータベースに対して検索を行った場合よりもより多くの解を求め得ることは既に述べたが，実際にどの程度の解が増えるかということは問合せ処理における大きな興味の一つであるため，解の大きさについてシミュレーションを行い，本手法の有効性を検証した．さらに実際に BibTeX 形式で記述された文献データベースを取り扱うプロトタイプシステムを実装し，実際のデータを用いて問合せ処理手法の有効性の検証を行った．

また，さらに一般的な対象を考えた場合に考慮しなければならない情報資源の異種性の問題，および情報の組織化の問題についても検討を行い，複数の情報資源を統合的に取り扱う際の問題点を論じ，情報の選択と属性間の写像を与えるための支援としてデータベースのクラスタリングについての検討を行った．そして，異種性を解決し統合的なアクセスを実現するシステム構成とシステム構成要素について議論し，複数の情報資源を統合的に取り扱うためのシステムである DIRECTOR の提案を行った．

1.4 本論文の構成

まず，2章では関連する研究についてサーベイを行う．3章では統合的な問合せ処理を実現するための手法を提案し，その解の正しさ等について述べる．4章では3章で提案した問合せ処理手法の有効性について論じ，5章ではこの問合せ処理手法の応用として，BibTeX 文献データベースを対象としたシステムを紹介する．6章ではさらに一般的な対象を考えた場合に考慮しなければならない問題についてまとめ，複数の情報資源を統合的に取り扱うためのシステムである DIRECTOR の提案を行う．7章では本研究全体の考察を行い，最後に8章でまとめとこれからの展望を述べて終りとする．

第 2 章

関連研究

分散した情報資源を統合的に取り扱おうとする際には，

1. 情報資源が構成する情報空間の組織化・管理の方法
2. 情報資源の発見方法
3. 情報資源の異種性 (heterogeneity) の吸収方法

などの問題を解決する必要がある．1は情報源へのアクセス方法にも大きく関係し，このことから2とも関連が深いのは明らかである．2の情報資源の発見方法とは，膨大な情報からいかにして有用な情報を見つけるかというものであり，3の異種性の吸収方法とは，異種性を持つ複数の情報資源を，ユーザから見た場合には一つの情報資源として取り扱えるようにするというものである．まず2.1節において実際のシステムの例をあげながら組織化の方法と情報資源の発見方法について述べる．また2.2節では有用な情報資源を選択するという観点から，情報資源の分類，データベースのクラスタリングについての研究を紹介する．3の異種性に関しては2.3節で論じ，異種データベースの研究中におけるスキーマ統合の手法を紹介する．最後に2.4節では本研究の問合せ処理で用いる，データベースを論理として取り扱う研究を紹介する．

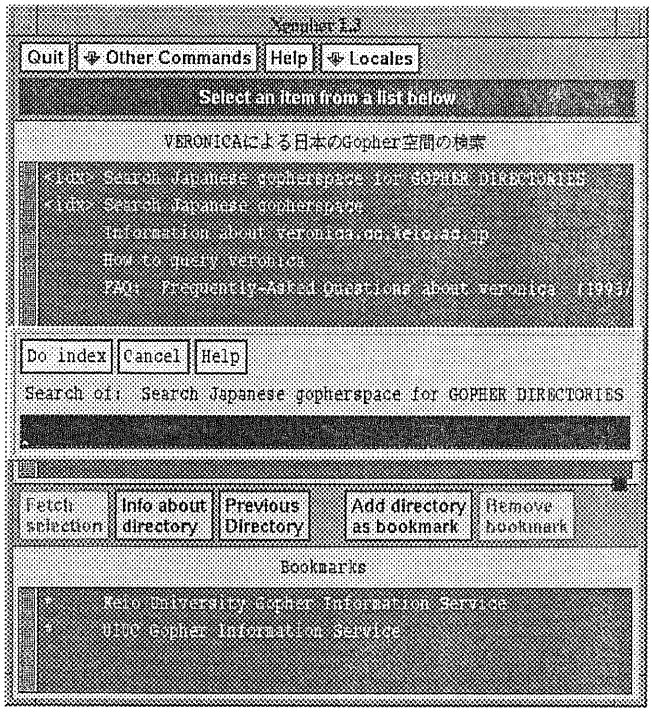


図 2.1: Gopher による情報空間の探索

2.1 分散情報システム

1990 年代は ATM を代表とするネットワーク技術の発展により，100Mbps 以上の速度でデータ転送が可能となっており，ネットワーク上で取り扱う情報を集中的に管理することの限界から分散情報システムの研究が盛んに行われている．分散情報システムを取り扱う際に問題となるのは資源発見 (Resource Discovery) および資源管理 (Resource Management) の問題である．本節ではインターネット上で実際に稼働しているシステムや，研究ベースのシステムを紹介しながら，そのシステムにおける資源発見，資源管理について述べる．

2.1.1 インターネット上で実際に利用可能な情報システム

1995 年現在インターネット上ではさまざまな情報サービスシステムが使用されている．例えばコンピュータネットワークを利用する際に不可欠な Domain Name System (DNS) [54, 55] もこのような情報サービスシステムの一つである．本節ではこのように広く用いられている情報システムである，Gopher, WWW (World Wide Web), WAIS, Archie を簡単に紹介する．

Gopher[41, 51] は 1991 年にキャンパスにおけるヘルプ機能の合理化目的でミネソタ大学で設計・開発された広域情報システムである．Gopher はクライアント–サーバモデルに基づいており，情報源の組織化に階層構造を用い，階層メニューによってリソースをカテゴライズする．このメニューはシステム上のリソースをダイナミックに反映する (図 2.1参照)．Gopher は基本的にメ

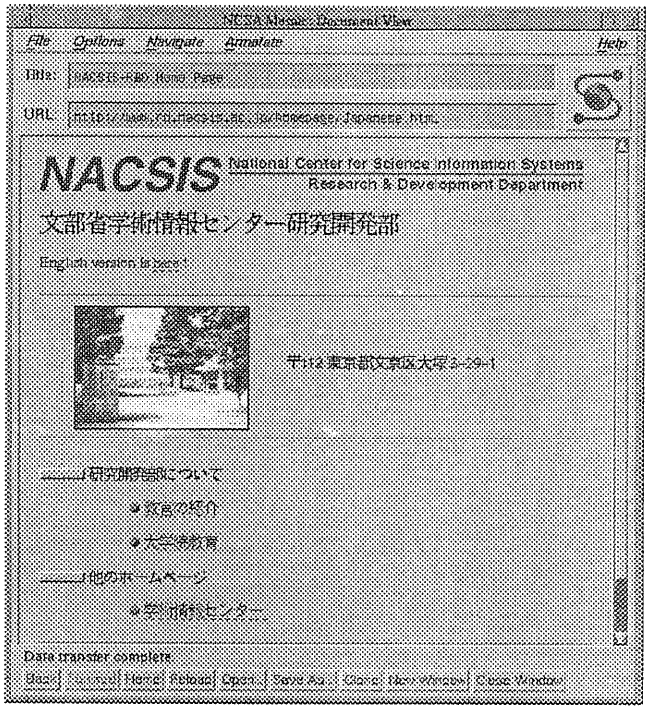


図 2.2: NCSA Mosaic (NACSIS R&D's homepage)

ニューを辿っていくことによって情報空間を航行するため，そのままでは情報の検索には適さない．Gopher による情報検索を可能とする目的でネバダ大学で開発されたソフトウエアが Veronica[94] である．Veronica は定期的に Gopher サーバからデータを集め，索引を作ることによって AND, OR, NOT, ワイルドカードなどを用いた検索が可能となる．

WWW (World-Wide Web)[5] は分散ハイパーメディアシステムを構築することを目標としており，情報サービスにハイパーテキストの概念を採り入れている．文書は文書中の他の部分や，画像，音声，動画あるいは別の文書にリンクされており，ユーザはそのリンクをたどることによって，情報空間を航行することができる (図 2.2参照)．このリンクは URL (Uniform Resource Locator)[6] という形で記述される．URL の記法は図 2.3 のようになっており，記法中に明示的にホストのア

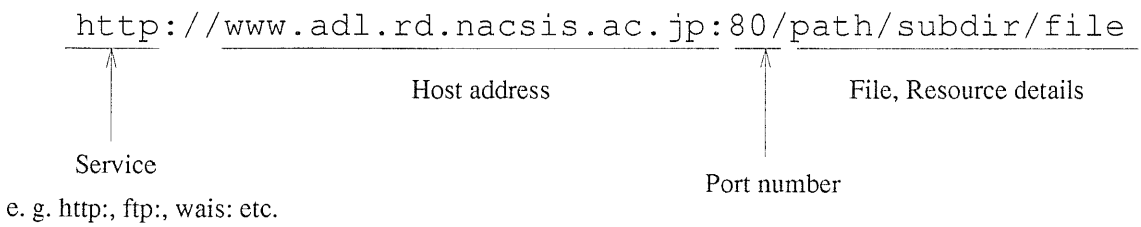


図 2.3: URL の記法

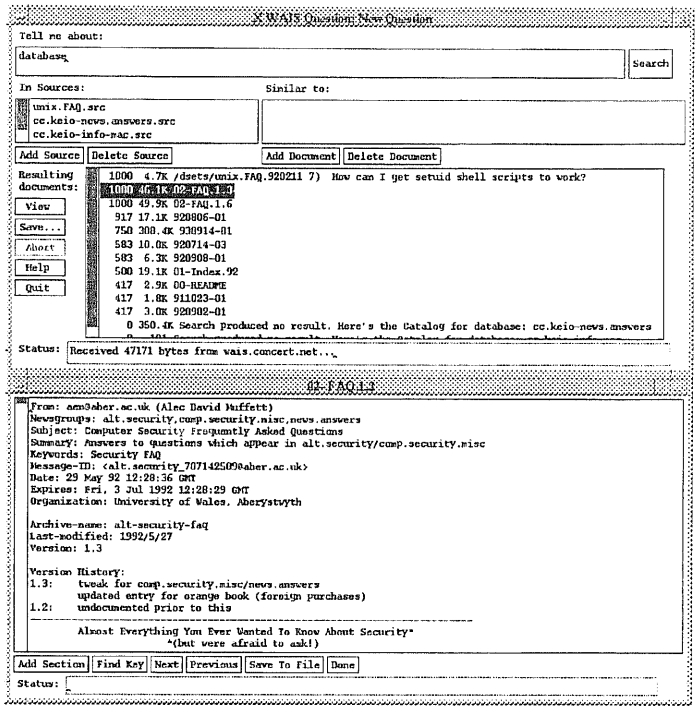


図 2.4: WAIS のインタフェース

ドレスとファイルのパスが現れるため，情報の仮想化はが完全に達成されているとはいい難い．

WWW はその文書の記述法として，簡単な「マークアップ言語」である HTML (Hypertext Markup Language) を用いることができる．「マークアップ言語」とはフルテキストに対して，そのテキストのブロックの表すセマンティクスに応じた「タグ」を付けるものであり，SGML (ISO 8879:Standard Generalized Markup Language, [78] はその簡単な入門) が有名である．また通信のプロトコルとしては HTTP (Hypertext Transfer Protocol) を用いる．

WWW ではユーザが必要としている情報を直接検索する方法がないことが，情報検索システムという観点から WWW を見た場合の弱点といえるが，WWW 空間を探索するロボットの研究が進んでいる [73]．例えば WWWW (the WWW Worm) [50] は HTML ファイルを辿りながら URL のタイトル文字列からインデックスを構成し，ユーザからの検索要求に対して URL を返す．

WAIS (Wide Area Information Server)[32] プロジェクトはクライアント-サーバモデルに基づいた分散情報検索システムである．WAIS は情報検索用のプロトコルである ANSI Z39.50[65] の拡張として位置付けられる．主要な拡張点は Type-3 の問合せとして，“Relevance Feedback Query” をサポートしたというものである．このサービスは現在既にインターネット上で稼働しており，X-Window 上で動くインタフェースも使用可能となっている (図 2.4 参照)．この WAIS はサーバ同士の得点づけなどの概念をサポートしているが，基本的にサーバの選択はユーザが行うものとなっている．

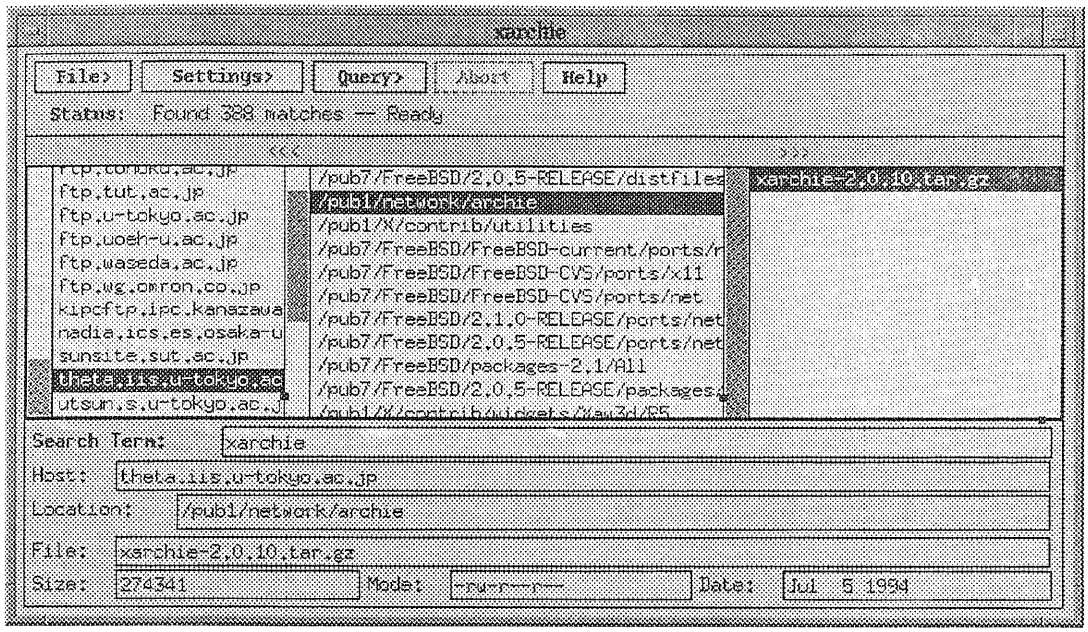


図 2.5: Xarchie

Archie[22] は Filename データベースとしてよく知られている．これはインターネット上の多くの FTP サイトに存在するファイルの名前を索引として保持している．X-window 上で稼働する Archie のクライアントである Xarchie(図 2.5 参照) は分散ファイルシステムである Prospero[60] を用いている．

ここで注意するべきことは Gopher と Veronica の組み合わせの検索対象はメニューのみ，WWW と WWWW の組み合わせの検索対象は URL のタイトルのみで，文書内部に関しては全く関知しないということである．これはつまり，メニューあるいはタイトルに依存するところが大きく，問合せと文書の関連性が小さいという事態がしばしば起こる．これに対して，WAIS は文書の内部検索を行い，文書に得点をつけることにより，検索結果に順序づけを行う．WAIS はクライアント-サーバモデルに基づいた分散情報検索システムであり，情報検索用のプロトコルである ANSI Z39.50 の拡張として位置付けられる．WAIS は複数のサーバに対して同時に問合せを行えることがその特徴の一つであるが，基本的にサーバの選択はユーザが行うものとなっている．また，Archie はファイル名の検索というようにその対象が限定されているが，実際に検索が可能であり，現在では最も実用的なシステムの一つである．しかしながらクライアントが増えるに従って，サーバへのアクセスの集中は大きな問題となりつつある．表 2.1に 4 つのシステムの特徴をまとめる．

2.1.2 研究段階のシステム

Gopher, WWW は内容検索を行わず，WAIS はいわゆる「データベースのためのデータベース」を用いるが，これは人間の手でメンテナンスされ,「時代遅れ」になってしまう傾向にある．情報

表 2.1: 4 つのシステムの特徴

	WAIS	Gopher	WWW	Archie
本来の対象	情報検索	キャンパス内情報サービス	CSCW	ファイル名データベース
情報空間管理法	索引	有向グラフ	有向グラフ	索引
検索 (Query)	○	×	×	○
閲覧 (Browse)	×	○	○	×

システムを使うユーザから見れば関連のある文献あるいは興味のある情報を蓄積しているサイトを教えてくれるようなシステムが望まれる．本節ではこのような目的で研究されているいくつかのシステムを紹介する．

CNRI (Corporation for National Research Initiatives) が Stanford 他 5 つの大学と行った CSTR (Computer Science Technical Reports)[17] プロジェクトは，ネットワーク上に公開されている計算機科学分野のテクニカルレポートに対して統合的な検索を行えるような枠組を作ることを目標においており，Stanford 大学ではそのインプリメンテーションのとして，GlOSS (Glossary Of Servers Server)[28] を開発した．GlOSS では

1. 集中化索引を作るのは不可能である
2. 全てのデータベースに対して検索を行わない
3. 内容検索を行う

という前提をおいている．この実現のためには，実際に検索を行う前にどのデータベースに対して検索を行うかを決定する機構が必要となる．GlOSS では各データベース上の文書中に現れる単語に注目し，注目する単語が現れる文書の数を数え，その情報をインデックスとして持つ．

$$\frac{\prod_{i=1}^{n} freq(t_i, db)}{DBSize(db)^{n-1}} \tag{2.1}$$

実際にどのデータベースに検索を発行すべきかを決定する際には (2.1) 式を指標として用いる．ここで db, t_i はそれぞれデータベース，キーワードを表し $freq(t_i, db), DBSize(db)$ はそれぞれ db 中でキーワード t_i を含む文書の数，db 中の文書数を表す．つまり，直観的には (2.1) 式は db 中で指定されたキーワードが全て現れる期待値を表すものとなっている．[28] では INSPEC データベースに対して単純にインデックスをつける場合と比較して，本手法では約 2.3%の容量で済み，しかも INSPEC, COMPENDIX 等の 6 つのデータベースに関して，本手法を適用することにより，問合せに対して 84%以上の文献を見つけることができたことが示されている．

現在 ACM, Bellcore, WAIS Inc., NASA, Xerox PARC 等が参加している SIDLP (Stanford Integrated Digital Library Project) [86] は 2.1.2節で述べた CSTR プロジェクトの流れをくむプ

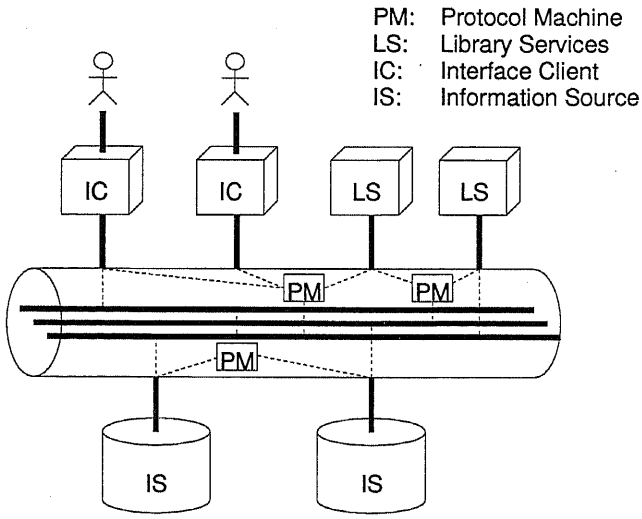


図 2.6: Information Bus

ロジェクトであり，とかく注目されがちなマルチメディア情報はとりあえず対象外とし，その主眼を情報の統合においている．SIDLP は一つの統一されたリポジトリを作ろうというわけではなく，個人的に管理されているような異種 (heterogeneous) 情報を統合しようとする試みであり，それらの情報を「普遍的な」ライブラリとして取り扱えることを目標にしている．

SIDLP は情報を統合するという目標を達成するために，Information Bus(図 2.6 参照) という概念を定義する．Information Bus 中のプロトコルマシンは言語，プロトコルなど，ディジタルライブラリの活動を記述するのに十分な構成要素からなり，FTP, SGML, HTML, Gopher, MIME などの各プロトコルに対して統合化されたアクセスを提供する．また，ユーザに検索の “手がかり (glue)” を与えるために，PIA (Personal Information Assistant) といういわゆるエージェントを用意する．PIA は

1. 共有の権限を持っている他の PIA へのゲートウェイとなる

2. 他の PIA の持っている情報のタイプについての情報を持つ

ことによってユーザの検索に対応する．SIDLP では著作権の問題についても考慮しており，高解像度なものをこれで印刷するという secure printer の概念や，ベンダが文献そのものではなく，文献を生成できるプログラムを転送し，ユーザが文献を読むためにこのプログラムを動かすとベンダにメッセージが送られるというような active document という概念を考えている．

MINERVA[90] は Mediator と呼ばれる自律的に動作するシステム要素を備えることによって，複数の情報資源を統合的に取り扱う．MINERVA では DIOE (Distributed Information Operating Environment) という環境を考える．DIOE は Dispatcher, Query Mediator, Source Mediator というモジュールからなる (図 2.7参照)．Query Mediator は知識を使って確率的に適当な粒度の問

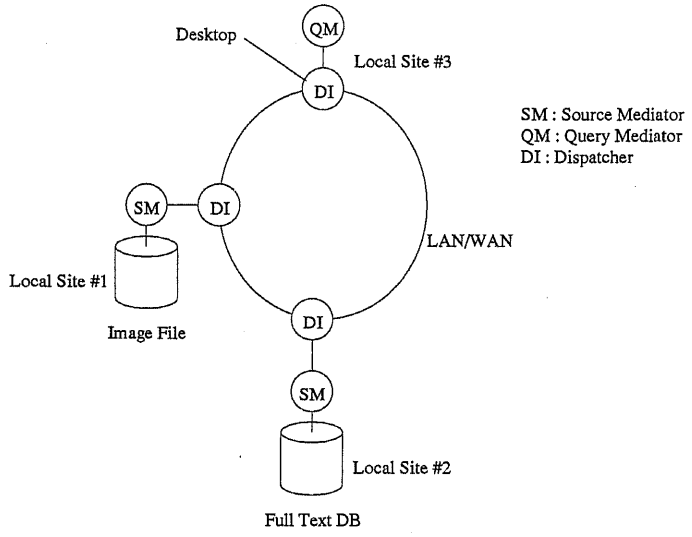


図 2.7: MINERVA DIOE

合せを構成し，各情報源から返された解を結合する．Source Mediator は局所的な知識を使って問合せを適当な粒度に分解し，解を生成する．Dispatcher は他の Dispatcher への接続を行うための知識を有し，問合せ，解などのメッセージを配送する．Indie[18] は分散インデックスを構成し，Indie broker と呼ばれる自律的に動作するモジュールを利用することによって，他の broker やリポジトリの情報を得ることができる．これらのシステムにおける「自律的な動作」は定義が曖昧で客観的な判断が難しいが，現在注目されつつあるエージェントとの関連で研究が盛んな分野でもある [72]．例えば客観性については [35] では異種分散エージェントの解の一貫性と，大域解への収束を保証する progressive negotiation と呼ばれる戦略を提案し，形式的な議論を行っている．

ROBIN[62, 63] は異なった情報源を取り扱う際にそれらの情報源のアクセスに必要な情報を管理するサーバを設定し，その情報を RO (Resource Object) というモジュールを用いることによって管理した (図 2.8参照)．概念の写像を適切に行うために，DNRO (Database Name Resource Object) と SMRO (Semantics Mapping Resource Object) という 2 つの RO を用いた 2 段階の写像を行っている．しかしながら，RO に保持する情報をいかにして作成するかという議論が不十分であった．

2.2 データベースの分類

2.2.1 分類学の概要

分類についての研究は 1960 年代まで遡る．分類 (classification) とは 1961 年に Simpson によって与えられた系統分類学 (systematics) における定義をもっと一般的に改めたもので「分類とは生物間に見られる関係に基づいて生物を集団 (あるいは集合) に整理することである」となってい

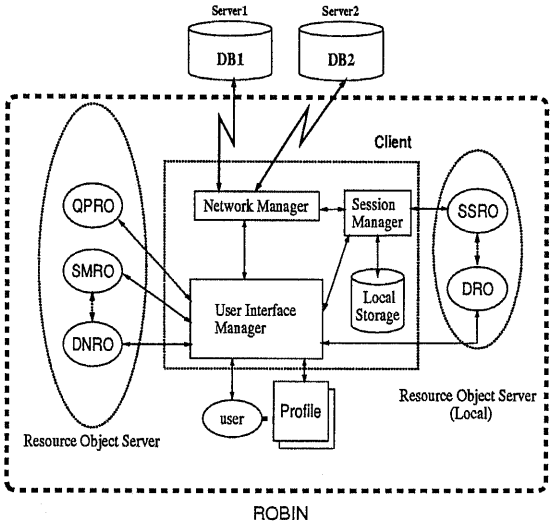


図 2.8: ROBIN のモデル

る．分類学 (taxonomy) とは「分類に関する理論的研究であり，それには分類の根拠，原理，手順，規則なども含まれる」と定義されていた [84]．この，もともとは生物の種を分類する目的で発展してきた研究分野である分類学は今日では科学・工学などをはじめとする様々な分野で利用されるようになっている．分類を行うためには操作的分類単位 (OTU) 間の関係を把握し，類似度を計算する等の処理を行うことになる．類似度の計算を行うためにはデータをデータマトリックスという形で記述するのが一般的であり，そこから類似度を表す係数を抽出する．この係数についてはOTU 間の距離に基づく距離係数を用いる方法，状態の一致の度合に基づく連関係数を用いる方法，OTU のベクトルのペア間の独立性を測定する相関係数を用いる方法，統計の技法を用いる確率的類似度係数を用いる方法などをあげることができる．エントロピーを用いる方法もこの確率的類似度係数法の一種と考えることができる．

2.2.2 最近の研究

[53] では分類という言葉には二つの異なる意味があるとしている．一つは発見の集合が与えられ，そこからデータ中のクラスあるいはクラスタの存在を確立すること．そうしてもう一つはたくさんのクラスの存在は仮定されており，新たな発見をどのクラスに分類するかの規則を確立しようとするものである．前者は教師なし学習あるいはクラスタリングと呼ばれ，後者は教師あり学習と呼ばれる．

近年の分類の研究には大きく分けて三つの流れがある．一つめは統計的なアプローチ，二つめは機械学習からのアプローチ，そして三つめはニューラルネットワークからのアプローチである．[53] ではこの三つについて以下のような特徴づけが行われている．

- 統計的なアプローチは明示的にある確率モデルに基づき，統計学の技法を利用する．つま

り，問題の構造化等に人間の介在を仮定するのが普通である．

- 機械学習においては，例の列から手続きを学習するということが行われ，遺伝的アルゴリズム，帰納論理プログラム，決定木を用いた手法が盛んに研究されている．機械学習では，オペレーションに人間の介在を仮定しない．

- よく知られているように，ニューラルネットワークはいくつかの階層から構成され，その出力はフィードバックを伴い，入力とは非線形の関係にある．本手法に基づいたクラスタリングは人間の振舞を真似ようというものであるが，理論づけは難しい．

2.2.3 データベースとクラスタリング

データベースとクラスタリングの関係の比較的新しい研究を概観する．まず，[96] ではデータベースは正しい情報のみを格納しているのではないという立場をとり，そこから有用な知識を得るためにデータベースの decomposition を行う．3 種類の DB 空間 instance space, probability space, learning space を考え，instance space から probability space への変換には probability distribution matrix (PDM) を用いる．これは基本的には条件付確率の考え方に基づいたもので，要素は $P(x_i \mid x_j)/N$である．ここで，Nは属性数，$P(x_i \mid x_j)$ は属性値 x_jが観測された場合に属性値 x_iが観測される条件付確率を表す．値は離散化されている必要があるので，このために Attribute oriented clustering という手法を用いる．これは背景知識を利用して連続値を離散化するものである．応用の対象としては Wisconsin Breast Cancer DB を考えて実験を行っている．本手法では属性値が取り得るパターンが少ない場合を取り扱っているため，要素を $P(x_i \mid x_j)/N$ とする PDM は適当な大きさとなっているが実際のデータベースではこの PDM は大きくなってしまうと思われる．

[9, 10, 11] ではテキストデータベースを対象とした文献のクラスタリングについて述べている．ここで提案された Cover Coefficient (CC) コンセプトはクラスタリングの seed の数を決定し，クラスタリングを行う．CC コンセプトは vector space モデル [76] に基づいている．vector space モデルとは簡単に説明すると文献や問合せをベクトルで表現するというもので，文献 D_i，問合せ Q_jは式 (2.2) のように表現される．ここで，d_{ik}および q_{jk}はそれぞれ文献，問合せに語 (term)k が現れた時に 1，そうでない時に 0 となる．また，語に重み付けを用いる場合もあり，この場合には d_{ik} (q_{jk}) は 0 か 1 であるとは限らない．

$$D_i = (d_{i1}, d_{i2}, \cdots, d_{it}), \quad Q_j = (q_{j1}, q_{j2}, \cdots, q_{jt}) \tag{2.2}$$

複数の文献を取り扱う際には，ベクトルを行列に置き換えるのは自然な発想である．つまり，文

献 $D_1,\cdots,D_N$，語 $T_1,\cdots,T_t$に対して，Term-Document Matrix Dを以下のように定義する．

$$D = \begin{array}{c} \\ D_1 \\ D_2 \\ \\ D_N \end{array} \begin{array}{c} \begin{array}{ccc} T_1 & T_2 & T_t \end{array} \\ \begin{bmatrix} d_{11} & d_{12} & d_{1t} \\ d_{21} & d_{22} & d_{2t} \\ & & \\ d_{N1} & d_{N2} & d_{Nt} \end{bmatrix} \end{array} \tag{2.3}$$

[11] では文献のクラスタリングを

1. partitioning type: クラスタは共通の文献を持たない
2. overlapping type: クラスタは共通の文献を持つ
3. hierarchical type: 階層型に文献を整理する

という 3 つの型に分類している．CC コンセプトに基づいた C^3M (CC-based Clustering Methodology) クラスタリングは 1 の partitionaing type に属する．C^3M アルゴリズムでは D 行列から C 行列を計算する．この C 行列の要素 c_{ij} は非形式的には

$$c_{ij} = \sum_{k=1}^{n} (d_i \text{から} t_k \text{を選択する確率}) \times (t_k \text{から} d_j \text{を選択する確率}) \tag{2.4}$$

と表される．つまり本モデルは two-stage probability model に基づいている．このように定めた C 行列は

1. $i \neq j, 0 \leq c_{ij} \leq c_{ii}$
2. $\sum_{k=1}^{m} c_{ik} = 1$
3. d_i 中の語が他の文献で使用されなければ $c_{ii} = 1$，そうでなければ $c_{ii} < 1$
4. もし $c_{ij} = 0$ ならば $c_{ji} = 0$．同様にして $c_{ij} > 0$ ならば $c_{ji} > 0$．ただし，一般に $c_{ij} \neq c_{ji}$
5. $d_i = d_j$ の時またその時に限り，$c_{ii} = c_{jj} = c_{ij} = c_{ji}$

などの文献の分類に好ましい性質を持っている．C^3M クラスタリングはこの C 行列の性質を利用してクラスタリングを行う．$\delta_i = c_{ii}, \psi_i = 1 - \delta_i$ をそれぞれ，d_i の decoupling coefficient, coupling coefficient と呼ぶ．また $\delta = \sum_{i=1}^{m} \delta_i / m$ を用いてクラスタ数 n_c を $n_c = \sum_{i=1}^{m} \delta_i = \delta \times m$ によって定める．また，d_i の cluster seed power P_i を

$$P_i = \delta_i \times \psi_i \times \sum_{j=1}^{n} d_{ij} \tag{2.5}$$

によって定める．第 1 項はクラスタの分離具合 (separation of cluster)，第 2 項はクラスタ内の文献の関連度 (connection among the documents within a cluster) を表す．第 3 項は正規化のため

のものである．C^3Mは seed-oriented の文献クラスタリング手法であるので，cluster seed power の大きい順にn_c個の文献が seed として選ばれ，seed となった文献とそうでない文献の c を計算し，クラスタリングを行う．

[75] では science の分野の論文の citation に着目し，その参照関係を行列で表現する．その行列 (非対称行列) に対して，TRCA (Total-relationship cluster analysis) という手法を用いる．本手法ではクラスタの数を制御できるのが一つの特徴である．考え方は単純で，citation の有無に重みをかけたものの和を使ってパラメータを設定するものである．有効性については実際に専門家とシステムによるクラスタリングの結果を比較している．231 個の CAD/CAM 関係の論文と，140 個の原子核物理関係の論文を用いて実験を行った結果，原子核物理関係の論文についてはシステムがあるクラスタに属すると判断したものについては専門家の意見と全て一致し，システムが境界領域にあると判断した論文は 12/140 個であった．

[9, 10, 11, 75, 96] はともにデータベース中の文献のクラスタリングである．実際に論文としては最近のデータベースの全文化の流れを受けて，語と文献の対応を考慮したクラスタリング手法についてのものが多いようである．

[52] では，AAH (Attribute Abstraction Hierarchy) という手法について述べている．本手法では属性 A, B，その値 a1, a2, b について，A=a1, B=b かつ A=a2, B=b なる関係が成り立つ時，値 a1 と a2 の間に関連があると見なし，正規化した関連度を計算し，この値を用いて bottom-up に binary-clustering を行う．専門家に属性の重みを与えてもらうことにより属性による重み付けを行うことも考慮している．目的は検索の際に正しい解が見つからない場合に似た解を見つけようとする CQA (Cooperative Query Answering) System に応用するためのもので，例えば C-21 という飛行機を検索した場合に全くそのもの (C-21) が見つからなかった場合には C-12 や C-20 を見つけるというものである．

[85] は図書館 (library) の比較を行おうとする試みである．まず問合せを本などのデータを対象とする data dependent query とある特定分野に関して 2 つの図書館を比較するというような data independent query に分類する．後者はデータレベルの比較だけではなく，構造レベルの比較 (structure level comparison) を必要とする．データレベルの比較に対しては従来のクラスタリング手法 (single linkage algorithm, K-mean algorithm 等) の組合せによって対応が可能であるが，構造レベルの比較には不十分であることを述べ，Knowledge-based クラスタリングを応用した手法を紹介している．基本的な考え方は階層化したクラスタのサイズを比較するという単純なものである．筆者らは [57] でも Knowledge-based クラスタリングについて述べているが，対象を文献とした場合の背景知識として ACM computer reviews category hierarchy を用いており，目次を見ながら人間が対応づけを行い，本を例えば (((I522 ∨ I532 ∨ I515)(1, 4)) ∧ ((I515 ∨ I531)(2, 4))) のように表現する．I522，I532 等が ACM computer reviews category hierarchy 中の要素であり，$I_n(1, 4)$ は 4 つの章を持つ本で 1 つの章に I_n が含まれていることを表す．

[4] は Conceptual Clustering(以下 CC) の手法を用いて，データベーススキーマの統合等を行うための手法を提案している．CC を用いるとこれまでのクラスタリング手法で用いていた閾値

の決め方，妥当性等の問題を回避することができるとしている．クラスタリングを行うにあたり，既存のクラス C_1 と新しいインスタンス I_1 に対して，COMPLIES$(C_1, I_1) = \{\text{TRUE, FALSE}\}$ というインスタンスがクラスに含まれるかどうかのチェックを行う関数を定義し，インスタンス I_2, I_3 に対してその共通部分となるクラス C_2 を求める INTERSECT$(I_2, I_3) = C_2$ という関数を定義している．本論文はデータベースが大きくなった場合，全てのインスタンスが共通に持つ性質は少なくなっていくと主張し，この例外に対処するためにこの INTERSECT を用いるというものである．INTERSECT の実際の処理はグラフマッチングで行われるが，このコストに関しての議論は行われていない．スキーマ統合には CANDIDATE データモデルを用いる．CANDIDATE データベースは $\{C, I, A, D\}$ という 4 項組で表現される．ここで，Cはクラス，Iはインスタンス，Aは属性，Dは互いに共通部分を持たないクラスのリストで構成される disjoint decompositions を表している．なお，本モデルではクラスと属性には階層が存在する．

[52] は属性値から属性間の関係を類推するもので，データベースからの知識発見 [26] の中の一つの大きな話題となっている．[57, 85] は分類の単位が文献ではなく図書館であるという点が興味深い．[4] では CANDIDATE データモデルを用いることによってスキーマ統合に役立つとしているが，クラスの定義等の制約が大きく，自動化は難しいように思われる．

2.2.4 データベースの有用性の指針

データベースの有用性の指針という言葉は，その有用性というものが目的によって異なることなどから非常に抽象的である．実際には，検索を行ってみなければその有用性はわからない．しかしながら 2.1.2節で説明した GlOSS[28] はその指針を与える一つの方法を提供した．本プロジェクトはネットワーク上に公開されている計算機科学分野のテクニカルレポートに対して統合的な検索を行えるような枠組を作ることをその目標においていた．

GlOSS では全文データベースを対象としたアプローチであるが，関係データベースを対象とする場合にはそのまま適用することはできない．そこで，ここではまず問合せによって生成される関係の大きさに着目する．[93] では質問処理コストの近似的評価法を行うために関係 Rの大きさに注目している．ここで関係の大きさとは関係 Rの組の個数のことを指し，関係 Rの大きさの期待値を $|R|$ と表現する．関係 Rはドメイン D_i $(i = 1, \cdots, k)$ の直積の部分集合である．以下では D_i, Rの大きさを d_i, r と書くことがある．[93] における制約は，各属性 A_i $(i = 1, \cdots, k)$ (A_iは各定義域の名前として定義されている) に任意の要素 $e \in D_i$が現れる確率は等しく $1/d_i$としている点である．今，直積を$\otimes$，選択演算をσ，射影演算をπ，結合演算を $\bowtie$ で表すものとすると，

$$|R_1 \otimes R_2| = r_1 r_2 \tag{2.6}$$

$$|R_1 \cap R_2| = r_1 r_2 / d_1 d_2 \cdots d_m \tag{2.7}$$

$$|R_1 \cup R_2| = r_1 + r_2 - r_1 r_2 / d_1 d_2 \cdots d_m \tag{2.8}$$

$$|\sigma_c R| = r / d_{i_1} \cdots d_{i_k} \tag{2.9}$$

$$|\pi_A R| \sim d_{i_1} \cdots d_{i_k} \left\{ 1 - \left(1 - \frac{1}{d_{i_1} \cdots d_{i_k}} \right)^r \right\} \tag{2.10}$$

$$|R_1 \bowtie R_2| = r_1 r_2 / d_{i_{s_1}} \cdots d_{i_{s_p}} \tag{2.11}$$

などとなる．ただし，ここで R_1 と R_2 は同じ定義域 $D_1 \cdots D_m$ を持つものとし，選択演算においては $C = \{A_{i_1} = e_1, \cdots, A_{i_k} = e_k\}$，射影演算においては $A = \{A_{i_1}, \cdots, A_{i_k}\}$，結合演算においては属性の組 $A_{i_{s_1}}, \cdots, A_{i_{s_p}}$ と $A_{j_{t_1}}, \cdots, A_{j_{t_p}}$ (但し，各属性の定義域は等しいとする) を用いて結合を行うものとする．

また，データベースからの知識発見の研究においては，データベースをルールの集合とみなすが，このルールの良さについての尺度を与えようとする研究がある．知識発見では現在では統計情報や確率をいろいろな局面で用いるのが主流となっており，例えば [15] ではデータベースから得られた知識を分類するための手法としてχ二乗検定を用いており，[83] ではルールに確率を入れることによって，確からしさを表現している．この確率を入れた場合，ルールは以下のような形式で記述される．

$$\text{If } \mathbf{Y} = y \text{ then } \mathbf{X} = x \text{ with provability } p$$

確率 p はデータから見積られるが，[83] ではβ分布を用いることによってサンプル数が少ない場合に対処しており，確率を導入した場合のルールの「良さ」についてはシャノンの情報のエントロピーの概念を応用して尺度を与えてる．

2.3 情報源の異種性

2.3.1 異種データベース

従来分散データベースと一般的に呼ばれていたものは，異種性・自律性を求めるものではなくデータを分散させることによってその信頼性および可用性を高めようとするものであった．これに対して要素データベースの自律性を実現しつつ，その異種性を吸収して要素データベースを統合しようとする試みがある．これらのデータベースシステムは，連邦型データベースシステム (FDBS, Federated Database System)，マルチデータベースシステム (MDBS, Multi Database System) [42, 74]，異種データベースシステム (HDBS, Heterogeneous Database System)[43, 82] などと呼ばれている．まずこれらの関係を異種性，自律性というキーワードによって分類する．

異種性という側面から考えた場合，これらのデータベースは全て異種データベースを扱うということを目標にして設計が行われている．つまり，連邦型データベースシステム，マルチデータベースシステムはともに異種データベースシステムの一種であるといえる．よって本論文では以後異種データベースシステムという言葉は用いず，連邦型データベースシステム，マルチデータベースシステムという言葉を用いるものとする．

連邦型データベースシステムとマルチデータベースシステムについてはその定義は多種多様であるが，連邦型データベースシステムが要素データベースの統合に重点をおいているのに対して

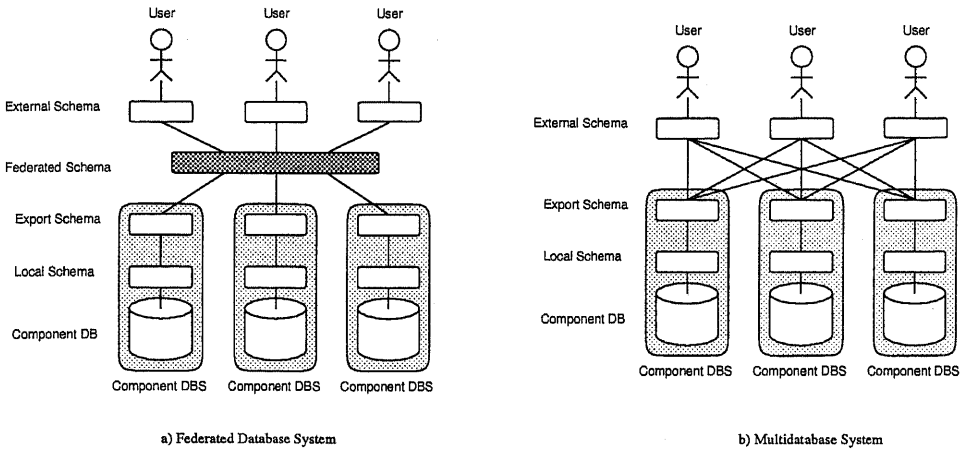


図 2.9: 連邦型データベースシステムとマルチデータベースシステム

マルチデータベースシステムは要素データベースの自律性の確保に重点をおいていると見ることができる．つまり，図 2.9に示すように，連邦型データベースシステムが各要素データベースのスキーマを統合した大域的な連邦スキーマ (Federated Schema) を持つのに対し，マルチデータベースシステムでは大域的なスキーマを持たないため，ユーザからはデータベースは一つには見えない．つまりマルチデータベースシステムは，ユーザが全域的なスキーマを用いないという所に最大の特徴がある．この結果，複数のデータベースには意味の違い，矛盾が存在することになるが，これらは自律性の一つの側面ととらえられる．マルチデータベースでは自律性の確立を最大の目標としているのであり，全域的なスキーマを共有する方法を否定するものである．

次に異種性について具体的に考える．[80] によれば，データベースの異種性として代表的には以下のようなものがあげられる．

1. **データモデルに関する異種性** データモデルとは取り扱うデータの概念体系を記述するためのモデルであり，階層モデル，網モデル，関係モデル，オブジェクト指向モデルなどをあげることができる．

2. **問合せ言語に関する異種性** 相異なるデータモデルのデータを操作するためには，一般的に異なる問合せ言語が用いられる．例えば関係データモデルについてはSQL という標準言語が存在するが，これについてもバージョンの違いなどが異種性の原因となっている．

3. **スキーマに関する異種性** スキーマに関する異種性については，構造に関する異種性，意味に関する異種性，表現に関する異種性をあげることができる．

本論文で取り扱う情報源の統合には主に 3 のスキーマの異種性が関連する．よって，このスキーマの異種性について，例をあげながらさらに詳しく説明する (図 2.10参照)．

- **属性名の相違** DB_1において学生番号となっている属性が，DB_2においてはID となっている．

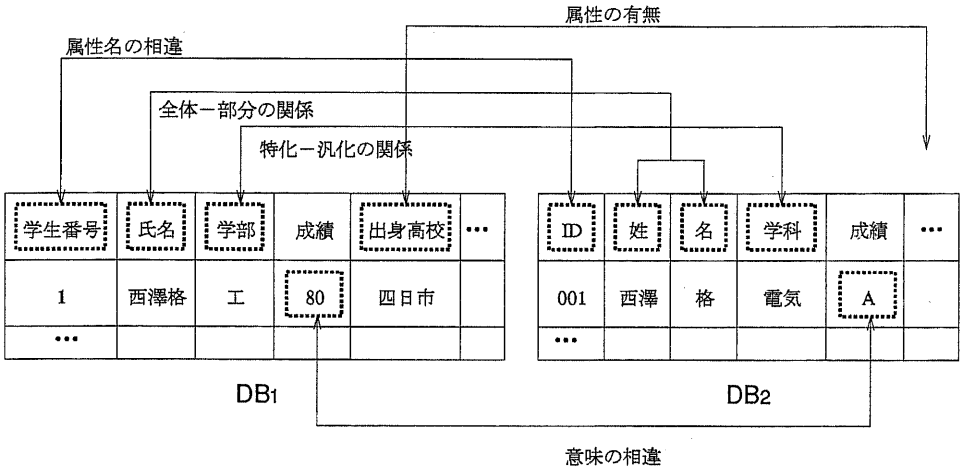


図 2.10: スキーマの異種性

- **意味の相違** DB_1においては成績は100点満点で記述されているのに対して，DB_2ではA～Eの5段階評価で記述されている．この時，成績はDB_1とDB_2において意味的に異種であるという．

- **データ構造の相違1: 全体-部分の関係** DB_1では名前として蓄積されているデータがDB_2においては姓と名にわけられている．

- **データ構造の相違2: 特化-汎化の関係** DB_1では学部が記述されているのに対して，DB_2では学科が記述されている．

- **属性の有無** DB_1における出身高校という属性に対応する属性がDB_2 には存在しない．

2.3.2 スキーマの統合

データベースのスキーマ統合については数多くの研究が行われている [34, 40, 43, 44, 66]．スキーマ統合とは複数の要素データベーススキーマを単一のスキーマに統合することであり，一般的には各要素データベース上のスキーマの構造，および意味の写像を行うことに相当する．各要素データベースのスキーマ間にはその構造および意味に違いが存在することはもちろんであり，ここではそのスキーマの違いを異種性と呼ぶ．つまりスキーマ統合においては各要素データベース上のスキーマの異種性をいかにして吸収するかが本質的な問題となる．連邦型データベースシステム [29, 44, 66] では，スキーマの統合は以下の手順で行われる．

1. 統合すべきスキーマおよびデータベースのオブジェクトを比較，解析する．

2. スキーマオブジェクト間の相互関係 (マッピング) を記述する．

これは簡単な仕事ではない．まず1では異種性が抽出されなければならないが，2つの属性間の異種性を発見するためのプロセスの自動化は簡単ではない．これは以下のような理由による [43].

- スキーマの意味の全てを完全に与える方法がないこと．
- 現在のデータモデルでは実世界の状態を完全に記述できないこと．
- 属性間の関連性を発見するためにはスキーマ内に書かれている以上の情報を必要とすること．
- 実世界には複数の視野と解釈があり，これは時間とともに変化すること．

このような理由から，スキーマの統合を行おうとする場合には，属性間の関係を，スキーマ統合を行う者があらかじめ与える必要がある．これが与えられた後，スキーマの統合が行われるがこれは以下のような手順による．

1. **スキーマの構造の変換** まずスキーマの構造の変換を行う．ここでは異なるデータモデルによって表現されたスキーマを共通のデータモデルへの変換を行う．データモデルの変換については議論が多いが，共通データモデルとしては実体-関連モデル，意味データモデルのような意味の表現が可能な概念的なモデルが好ましいとされる．例えば共通データモデルとして意味データモデルを用いると関係モデルのような従来のモデルでは表現できない付加的な情報を表現できることが知られている．
2. **属性の統合** 属性の統合とは属性間のマッピングを行うことである．この際には汎化によるアプローチが有力であるとされている [19]．これは同様の概念が異なるスキーマ中で異なった抽象化によって表現されている場合が多いためである．

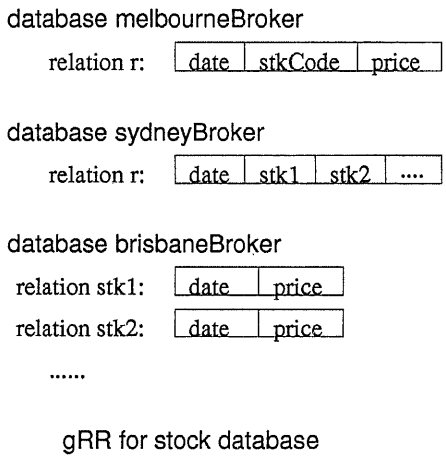
2.3.3 研究紹介

Pegasus プロジェクト [79] の目標は Oracle, Informix などの DBMS の統合のみならず，マルチメディアデータ，legacy アプリケーションなどもそのターゲットとして含んでいる．Pegasus モデルはオブジェクト指向モデルを基礎にした functional object model と呼ばれるもので，その構成要素は *types, functions, objects* などであり，アーキテクチャは機能によって4層構造となっている．Pegasus では各 External Data Resource Manager (EDRM) を用いるが，その際に EDRM と同じマシン上で稼働する Pegasus Agent を介してアクセスを行う．これによって他のサイトで Pegasus のかわりにある機能を実行することができる．

[21] では，スキーマ構造の異なるデータベースを取り扱うために，global reference relation (gRR) という概念を用いる．gRR は

gRR :	dbName	relnName	keyVal	attrName	attrVal

というフォーマットを持つ (実際の構成法については図2.11参照)．gRR には各データベースのインスタンスを格納しておく必要はなく，スキーマ情報のみを持つ．[21] ではスキーマは常に変化するも



gRR for stock database

dbName	relnName	keyVal	attrName	attrVal
melbourneBroker	r		date	
melbourneBroker	r		stkCode	
melbourneBroker	r		price	
sydneyBroker	r		date	
sydneyBroker	r		stk1	
sydneyBroker	r		stk2	
.....	.....	.....	.....	.....
brisbaneBroker	stk1		date	
brisbaneBroker	stk1		price	
brisbaneBroker	stk2		date	
brisbaneBroker	stk2		price	
.....	.....	.....	.....	.....

図 2.11: gRR の構成法

のであり以前に行った変換が現在も可能であるとは限らないという立場から *dynamic equivalence, statistic equivalence* という概念を定義している．問合せ言語としては relational calculus (RC) と Interoperable Database Language (IDL) を用いる．IDL は高階の関係言語で，gRR 上および各データベース上での表現力が等しいという特徴を持つ．

IRDS (Information Resource Dictionary System) は ISO によって進められたリポジトリに関する標準化作業によって成立した規格で，情報資源のメタ情報の格納方法を規定している．ここではリポジトリという語はソフトウェア開発保守におけるプロセスとプロダクトの統合管理のために対象ソフトウェアのデータや手続きの名前，属性，それらの間の関係を管理するものと定義されているものとする．IRDS[67] では「記述される対象」と「記述するもの」の無限に繰り返し可能な対象をメタ階層と呼び，リポジトリをメタ情報のファイルとしている．しかしながら，メタ情報として対象の属性を記述するかを特定することはできない [30]．IRDS と同様の標準規格として，ECMA (European Computer Manufacturers Association)/NIST (National Institute of Standards and Technology) で策定された PCTE (Portable Common Tool Environment)[2, 30, 77] をあげることができる．PCTE は元来プログラムのポータビリティを向上させるために設計されたものであ

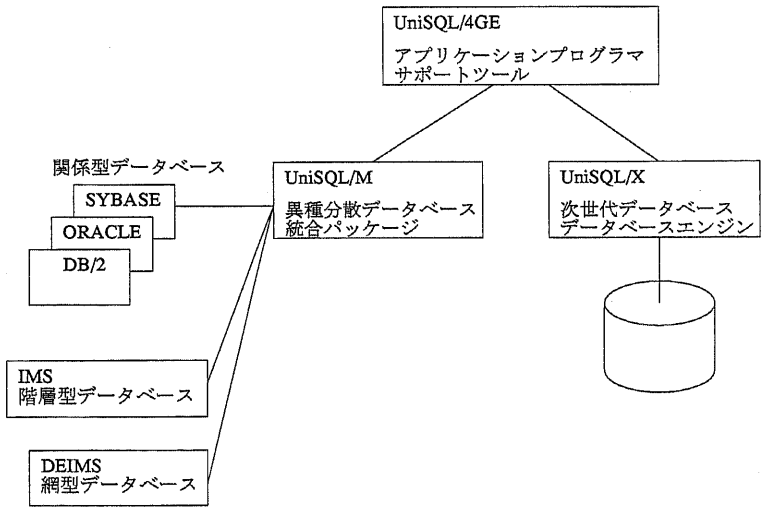


図 2.12: UniSQL の概要 (文献 [33] より引用)

るが，この規格の中にスキーママネージメントの項目が存在する．PCTE ではオブジェクトを利用し，各実体に型付けを行ない，その型に応じたルールを適用することによって，スキーママネージメントを行なっている．

[64] では IRDS と PCTE の比較が行われている．IRDS と PCTE はその対象範囲が異なり，IRDS がレコードベースのデータベースアプリケーションのソフトウェア開発のためのリポジトリのインタフェースであり，実行時にマッピングを行う機構を備えているのに対して，PCTE は専門的なアプリケーションを構築するためのオブジェクトベースのリポジトリのインタフェースであり，分散オブジェクトに対する透明性を備えている [64]．これらの 2 つの規格は両方とも CASE を対象としており，プログラム (システム) 構築を補助するという目的で作成されており，マッピングをどのようにして与えるかということについて述べている．

UniSQL[33, 37] は米国 UniSQL 社と NTT データ通信が共同開発した統合型データベース管理システムで，関係データモデルとオブジェクト指向モデルの統合方式を採用している．具体的には UniSQL は，UniSQL/4GE, UniSQL/X, UniSQL/M という 3 つの製品群から構成されており (図 2.12 参照)，UniSQL/4GE はデータベース応用ソフトウェアの開発環境を提供し，UniSQL/X は SQL/X という ANSI の SQL の機能データベース言語を用意し，オブジェクト指向への拡張を実現している．UniSQL/M はローカル DB のクラスまたはテーブルを proxy と呼ぶ仮想クラスの一種に写像し，それら proxy や他の仮想クラスを用いて，実際にアプリケーションで利用する仮想クラスを定義する [36].

[40] は自律して動作する異種データベースを統合する際のタプルが同じ実体を表現しているかどうかを決定するための手法について論じている．既存のデータベースを統合する際には，2 つのレベルの論理的異種性が現れる．つまりスキーマレベルの異種性とインスタンスレベルの異種

性である．本論文では，このインスタンスレベルの異種性を取り扱う方法を論じている．このような問題は，*instance integration*(以下 II) と呼ばれる．II はスキーマの構造 (属性のドメイン) および意味 (属性の意味) に矛盾がない場合を取り扱う．II を取り扱う問題では，以下の 5 つの方法がとられる場合が多い．

1. *Using key equivalence*
   共通のキーを用いる方法．この手法は当然ながら共通の属性を含んでいても共通のキーを持たない場合には適用できない．

2. *User specified equivalence*
   ユーザがマッピングを指定する方法．

3. *Use of probabilistic key equivalence*
   キーの一部の等価性を用いる方法．

4. *Use of probabilistic attribute equivalence*
   属性が同じであればその実体は等価であるとする方法．

5. *Use of heuristic rules*
   ヒューリスティックなルールを用いる方法．

[40] では取り扱うデータベースを関係データベースに限定する．また II の手法としてはキーによる方法をとるが，共通のキーを持たない場合を考慮して *extended key, extended key equivalence* という概念を導入する．extended key equivalence を実現するにあたり，共通キーを持たないタプルを *instance level functional dependency*(以下 ILFD) を用いることによってマッチングテーブルを作り識別する．このマッチングテーブルの構成法を図 2.13に示す．ILFD は属性レベルで与えられるのではなく，インスタンスレベルで与えられるため，場合によっては ILFD の数は膨大になる可能性がある．効率の面からこれらをテーブルとして保存しておくという対応をとる．

[81] はデータの複数のデータベース上の互いに関連するデータの一貫性について論じている．データの依存度は Data Dependency Descriptors (D^3) を用いて記述される．D^3は以下の 5 つの要素から構成される．

$$D^3 =< S, U, P, C, A > \tag{2.12}$$

ここで，S はソースデータオブジェクト，U はターゲットデータオブジェクト，P は *interdatabase dependency predicate* と呼ばれる述語で，取り扱うデータベースが関係データベースの場合には関係代数を用いて記述される．また，関係代数の selection, projection などの他に集約化オペレータα，推移閉包オペレータξが用意されている．C は *mutual consistency predicate* と呼ばれる述語で，時間や状態を含んだデータの一貫性を記述する．A は *consistency restoration procedures* と呼ばれ，一貫性が崩れた場合にそれを回復する手続きを記述する．一貫性は宣言形式で記述され，*interdependency schema (IDS)* と呼ばれる．これは D^3の集合である．

Table R

name	cuisine	street
Twin Cities	Chinese	Co.B2
It'sGreek	Greek	Front Ave.
Anjuman	Indian	Lesalle Ave.
VillageWok	Chinese	Wash.Ave.

Table S

name	speciality	country
Twin Cities	Hunan	Roseville
Twin Cities	Sichuan	Hennepin
It'sGreek	Gyros	Ramsey
Anjuman	Mughalai	Mpls.

extended key = {name, cuisine, speciality}

IFID :

I1:(E.speciality = "Hunan") –> (E.cuisine = "Chinese")

I2:(E.speciality = "Sichuan") –> (E.cuisine = "Chinese")

I3:(E.speciality = "Gyros") –> (E.cuisine = "Greek")

I4:(E.speciality = "Mughalai") –> (E.cuisine = "Indian")

I5:(E.name = "Twin Cities") $\wedge$ (E.street = "Co.B2") –> (E.speciality = "Hunan")

I6:(E.name = "Anjuman") $\wedge$ (E.street = "LaSalle Ave.") –> (E.speciality = "Mughalai")

I7:(E.street = "Front Ave.") –> (E.country = "Ramsey")

I8:(E.name = "It'sGreek") $\wedge$ (E.country = "Ramsey") –> (E.speciality = "Gyros")

Table R'

name	cuisine	speciality	street
Twin Cities	Chinese	Hunan	Co.B2
It'sGreek	Greek	Gyros	Front Ave.
Anjuman	Indian	Mughalai	Lesalle Ave.
VillageWok	Chinese	NULL	Wash.Ave.

Table S'

name	speciality	cuisine	country
Twin Cities	Hunan	Chinese	Roseville
Twin Cities	Sichuan	Chinese	Hennepin
It'sGreek	Gyros	Greek	Ramsey
Anjuman	Mughalai	Indian	Mpls.

Table MT_RS

R.name	R.cuisine	S.name	S.speciality
Twin Cities	Chinese	Twin Cities	Hunan
It'sGreek	Greek	It'sGreek	Gyros
Anjuman	Indian	Anjuman	Mughalai

図 2.13: マッチングテーブルの構成法

[20] は独立して開発されたデータベース間の定義域のミスマッチの問題を取り扱う．この定義域のミスマッチの問題はマルチデータベースの環境下のみならず，一つのデータベースで歴史的に蓄積されたデータを管理したい場合にも現れる問題である．本論文では仮想属性 “virtual attributes” なるものを考え，それに実際の属性をマッピングする．この際に一意にその値が決定できない場合はその値を部分値 (partial value) として表現する．部分値を利用することによって得られた結果は完全に正しいとはいえないものもその中に含んでしまうので，本論文では結果として得られたタプルに true, maybe という状態を表す情報を付加することによりその正当性を表現するパラメータとする．

2.4 データベースと論理

2.4.1 データベースとは

そもそもデータベースとは何かといえば，組織的に集められ管理されているデータの集合で，プログラムからは独立してデータベーススキーマによって定義されるもののことをいう．ここで，データベースにおけるスキーマとは，簡単にいうとデータベース中のデータの枠組のことであり，データベース中で扱うデータの定義を行うものである．

歴史的にみると，データはまずファイルという概念によって組織化された．このファイルというものはプログラム中で使用されるというだけのものであり，プログラムに従属するという形をとっており，ファイル相互の関連についてはあまり議論されるべきものではなかった．故に複数のファイル間の関係には一貫性がなく，冗長な構成となっている場合がほとんどであった．そこで，これらのデータをプログラムからきりはなし，データの一貫性，冗長性をなくし，さらに多数のユーザからの共通したアクセスが可能となるように統一的にデータを管理することが求められるようになった．

2.4.2 関係データモデル

関係データモデルは 1970 年に E. F. Codd が提案したデータモデル [16] で，それまでの網データモデルや階層データモデルとは異なり，そのモデルに基づいたデータベースシステムがどのようにインプリメントされるかというような物理的な制約からデータモデルを切り離したという物理データ独立性を確立したという点で画期的なものであった．関係データモデルの基礎となる数学的概念は集合論の関係 (relation) である．

関係モデルにおける関係を定義するためにはまず，定義域 (domain) を定義しなければならない．この定義域とは集合であり，学生の集合，年齢の集合などを例としてあげることができる．いま，このドメインを $D_1, D_2, \cdots, D_n$ と表すことにする．次に直積 (direct-product) という概念を導入する．直積とは集合 $a \in A$, $b \in B$ に対して，この直積を $A \times B$ で表し，

$$A \times B = \{(a, b) | a \in A, b \in B\} \tag{2.13}$$

に従って計算するものである．これらを用意した後，$D_1, D_2, \cdots, D_n$上の関係Rを$D_1 \times D_2 \times \cdots \times D_n$上の任意の有限部分集合と定義する．

次に本論文では議論の途中で関係代数特有の記号を用いるので，この説明を行う．関係データモデルには「関係代数」,「関係論理」という大別して2つのデータ操作言語体系が存在する．前者と後者は等価な能力を持つ．しかしながらその体系の言語に関しては関係論理に基づく言語は関係代数言語よりもしばしば高級であるといわれる．この理由は関係代数言語が手続き的であるのに対して関係論理言語が目的を達成する方法を述べる必要はなく，何が目的であるかを述べればよいという非手続き的なものであるからである．

関係代数はコッドにより提案された．コッドは関係に対する8つの演算を導入したが，これらは4つの集合演算と4つの関係代数に特有の演算に分けることができる．まず，4つの集合演算は，和 (union)，差 (difference)，共通 (intersection)，直積 (direct product) であり，関係代数特有の演算は，射影 (projection)，選択 (selection)，結合 (join)，商 (division) である．こららの8つの演算は互いに独立ではなく，5つの基本演算と呼ばれる，和，差，直積，射影，選択によって他の演算は表現することができる．以下の議論では和演算，共通演算，直積演算，射影演算，選択演算，結合演算をそれぞれ$\cup$, $\times$, σ, π, $\bowtie$ で表すものとする．

2.4.3 関数従属性とキー

関係データベースではその関係をいくつかの小さな関係に分割して保持している．一般の関係をこのように小さな関係に分割していくことを正規化するという．正規化を行う理由は正規化を行わない関係ではその情報の削除，挿入，修正の際に不都合が生じるためである．正規化にはいくつかの段階があるが，この過程において関係を形成する情報は無損失でなければならない．この分解が無損失であることを保証するための基準として関数従属性 [1] をあげることができる．関数従属性とは関係Rと，Rに含まれる属性集合の部分集合X, Yに対して，Xに含まれる属性の値を全て決めると，Yに含まれる属性の値が一意に決まる時，関係Rは属性XからYへの関数従属性を持つという．形式的な定義は以下のようになる．

スキーマ$R(X_1, X_2, \cdots, X_m, Y_1, Y_2, \cdots, Y_n)$ に関数従属性$X_1, X_2, \cdots, X_m \rightarrow Y_1, Y_2, \cdots, Y_n$が存在するとは，$R_i$を$R$の任意のインスタンスとした時，

$$\begin{aligned} s[X_1, X_2, \cdots, X_m] &= t[X_1, X_2, \cdots, X_m] \\ \Rightarrow s[Y_1, Y_2, \cdots, Y_n] &= t[Y_1, Y_2, \cdots, Y_n] \quad (\forall s, t \in R_i) \end{aligned} \tag{2.14}$$

が成り立つことである．但し，$\Rightarrow$ は論理的含意を表すものとする．

ある実体集合を取り上げたとき，一つの実体を一意に決める属性の集合の存在を仮定する．関係スキーマを$R(X_1, X_2, \cdots, X_m)$，関数従属性の集合をFとし，Yを$X_1, X_2, \cdots, X_m$の部分集合とすると，以下の条件が成り立つときYはRのキーであるという．

$$Y \rightarrow X_1 X_2 \cdots X_m \in F^+ \tag{2.15}$$

$$Z \not\rightarrow X_1X_2\cdots X_m \quad \exists Z \subset Y \tag{2.16}$$

ここでF^+はFの閉包 [92] と呼ばれ，Fによって論理的に含意される関数従属性の集合を表す．

一つの関係に対しては複数のキーが存在し得るので，これらを候補キーと呼び，その中の一つのキーを主キーとして指定する．関係データベースにおいてはこのキーは非常に重要な概念であり，一般的にはデータベースの設計者が実世界をデータベース化する際に意味的に大切なものをキーとして指定する．また，一般的にデータベースにはキー制約 (key constraint) が課され，主キーを構成する属性の値は空値であってはならないとされる．

2.5 論理としてのデータベース

データベースを論理的な述語の集合と見なし，データベースに関する種々の問題の解を述語論理を用いることによって演繹的に導き出そうとする試みがある [1, 3, 7, 24, 27, 61, 68, 69, 70, 71]．適用可能な問題としては，問合せ言語，問合せ評価，データベーススキーマの解析等があげられる．関係データベースは理論的な基礎がしっかりしており，論理と相性がよい．データベースを論理で表現する際には一般的に一階述語論理が用いられるため，まず一階述語論理について簡単に紹介し，次に関係データベースのと述語論理の関係について，Reiter の研究を中心としてまとめる．

2.5.1 第一階述語論理における論理式

まず第一階述語論理の体系における論理式の定義を行う．そのために最初に項の定義を行う．項は以下のように再帰的に定義される．

[定義 1]

1. 定数は項である．
2. 変数は項である．
3. fが n-変数関数，$t_1,\cdots,t_n$が項であれば $f(t_1,\cdots,t_n)$ は項である．
4. 全ての項は上の 1〜3 の規則によってのみ生成される．

この手続きによって項を定義した時，第一階述語論理における原子論理式は以下のように定義される．

[定義 2] P を n-変数述語記号，$t_1,\cdots,t_n$を項とした時 $P(t_1,\cdots,t_n)$ は原子論理式である．

原子論理式の定義ができたのでこの原子論理式と，5 つの論理演算子

$$\neg(\text{否定}),\ \wedge(\text{論理積}),\ \vee(\text{論理和}),\ \rightarrow(\text{含意}),\ \leftrightarrow(\text{同値})$$

と2つの限定記号

$$\forall(\text{全称記号}),\ \exists(\text{存在記号})$$

を用いて，第一階述語論理における整合論理式 (well-formed formulus 以下単に論理式とする) を以下のように帰納的に定義する．

[定義 3]

1. 原子式は論理式である．
2. F, G が論理式であれば $(\neg F)$, $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$, $(F \leftrightarrow G)$ は論理式である．
3. Fが論理式，x が F中の自由変数であれば $(\forall x)F$, $(\exists x)F$は論理式である．
4. 全ての論理式は 1, 2, 3 を有限回適用することによってのみ生成される．

2.5.2 論理式の解釈

第一階述語論理式の解釈方法を述べる．第一階述語論理式 Fの解釈は，空でない定義域 Dと，F中の定数，関数，述語の各々についての値の割り当てによって行われる．この値の割り当ては以下の手順で行われる．

1. 各定数に対して，Dの要素を割り当てる．
2. 各 n 変数関数に対して，$D^n(= \{(x_1, \cdots, x_n) \mid x_1 \in D, \cdots, x_n \in D\})$ から Dへの写像を割り当てる．
3. 各 n 変数述語記号に対して，D^nから $\{true, false\}$ への写像を割り当てる．

定義域 D上の論理式の解釈における論理式の真理値は，以下の規則によって $true$ か $false$ のいずれかに決定される．

1. 論理式 G, Hの真理値が計算されていれば，真理値表を用いて $(\neg G)$, $(G \wedge H)$, $(G \vee H)$, $(G \rightarrow H)$, $(G \leftrightarrow H)$ の真理値が計算できる．
2. $(\forall x)G$ は，G の真理値が Dの全ての要素 x について $true$ となる場合にのみ $true$ であり，それ以外の場合は $false$ となる．
3. $(\exists x)G$ は，G の真理値が Dの少なくとも 1 つの要素 x について $true$ となる場合 $true$ であり，それ以外の場合は $false$ となる．

2.6 関係データベースの述語論理による形式化

2.5節で第一階述語論理における論理式，2.5.2節でその解釈について述べた．本節ではその関係データベースへの適用について述べ，空値を含んだ関係データベースに対する問合せの評価に述語論理を用いた研究を紹介する．

本論文では3章における問合せ処理の際に，データベースを論理的な述語の集合とみなして議論を行っている．特に，関係データベースはその理論的な背景が論理と相性がよい．Reiter は空値を含んだ関係データベースを形式化し，問い合わせとそれに対する解を形式的に表現した [68, 69, 70, 71]．

2.6.1 Reiter による形式化と空値

空値とはデータベースにおいてその値が欠落していることを表す特別の値 [59] のことをいう．まず Reiter はこの空値に対し，“存在するが，それが何かはわからないもの” という意味づけを行った．Reiter はデータベースを論理式によって記述するにあたり，以下の 2 つの仮説を導入する．そしてこの 2 つの仮説に基づいて，データベース中の関係を論理式で表現する．

1. **閉世界仮説 (The Closed World Assumption)** 関係を表現する表に記述されている関係が世界の全てであるという仮説である．例えば，以下のような供給者–部品データベースを考える．

PART	SUPPLIER	SUPPLIES	SUBPART
p_1	A	$A\ p_1$	$p1\ p2$
p_2	B	$B\ p_2$	
p_3			

このとき，閉世界仮説により，このデータベースは述語によって以下のように表現される．但し，$E(x,y)$ は x と yが等しい場合にのみ $true$ となる関係を表すものとする．

$$(x)[\mathrm{PART}(x) \equiv E(x,p_1) \vee E(x,p_2) \vee E(x,p_3)] \tag{2.17}$$

$$(x)[\mathrm{SUPPLIER}(x) \equiv E(x,A) \vee E(x,B)] \tag{2.18}$$

$$(x)(y)[\mathrm{SUPPLIES}(x,y) \equiv E(x,A) \wedge E(y,p_1) \vee E(x,B) \wedge E(y,p_2)] \tag{2.19}$$

$$(x)(y)[\mathrm{SUBPART}(x,y) \equiv E(x,p_1) \wedge E(y,p_2)] \tag{2.20}$$

$$\tag{2.21}$$

2. **唯一名仮説 (The Unique Name Assumption)** 空値でない値はその違いを識別できるという仮説である．先ほどのデータベースの例では

$$\neg E(A,B),\ \neg E(A,p_1),\ \neg E(p_1,p_2),\ \ etc \tag{2.22}$$

である．

次に問合せを述語によって表現する．例えば先ほどのデータベースに対して，「1 個以上の部品を供給している供給者は?」という問合せは以下のように表現できる．

$$\langle x/\mathrm{SUPPLIER}|(\exists y/\mathrm{PART})(\exists z/\mathrm{PART})\neg E(y,z)\wedge \mathrm{SUPPLIES}(x,y)\wedge \mathrm{SUPPLIES}(y,z)\rangle \quad (2.23)$$

このように述語で与えられた問合せに対し，まずこれを分解していき $\langle \mathrm{x}/\tau|P(\mathrm{r})\rangle$ ，あるいは $\langle \mathrm{x}/\tau|\neg P(\mathrm{r})\rangle$ という形に変換する．ここで $P(\mathrm{r})$ は原子論理式で，全ての変数 x は r の中に現れるものとする．この形に変換した後 (2.17), (2.18), (2.19), (2.20) 式を用いて評価を行う．

この，問合せ評価に述語論理を用いるというアプローチによれば，関係データベースが空値を含まない場合には正しく (sound) かつ完全 (complete) な解を返すことが証明されており，空値を含む場合でも正しい解を返すことが示され，しかもそのアルゴリズムが与えられている [71].

第 3 章

統合的アクセスを実現するための問合せ処理手法

本章では問合せ処理手法について述べる．まず 3.1節で，想定する環境について述べ，問合せの処理の例を示す．さらに本アプローチの核となる仮想データベース (VDB, Virtual Database) と仮想関数従属性 (VFD, Virtual Functional Dependency) を定義する．次に 3.2節では仮想データベースおよび問合せ，解の形式化を行い，VFD を用いた問合せ処理によって得られる解の正しさと完全性について議論する．3.3節では問合せ処理のアルゴリズムについて述べる．

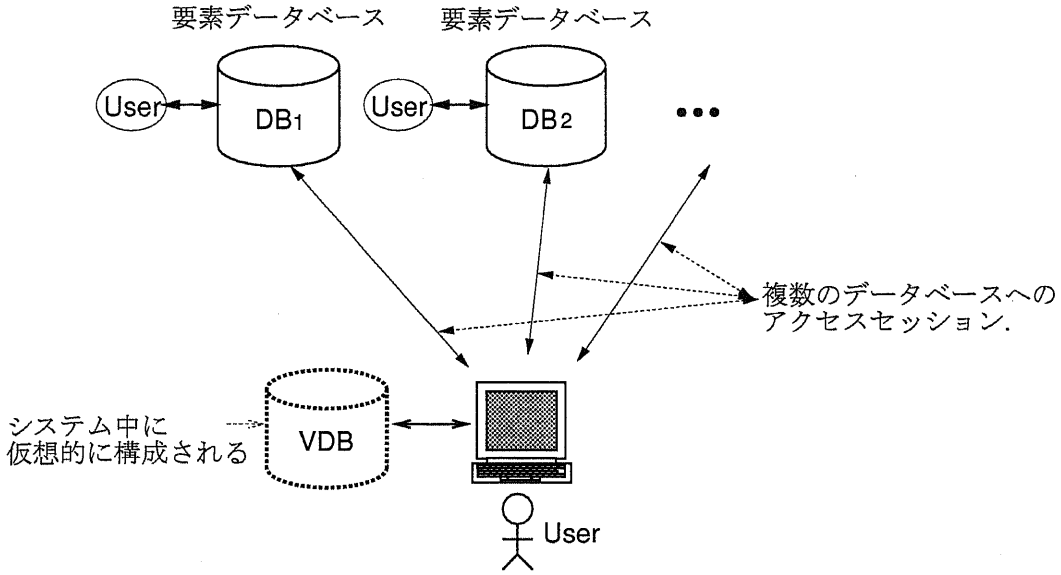


図 3.1: 複数データベースへの統合的アクセス

3.1 データベースへの統合的アクセス

3.1.1 想定する環境

本研究では図 3.1に示すように属性が異なる複数のデータベースが存在し，これにユーザが統合的にアクセスを行うという環境を想定する．各データベースのスキーマは属性集合に限定されており，これは異種データベース [43] 環境の一つの特殊な場合と考えることができる．各々のデータベースを要素データベース と呼ぶ．各要素データベースは独立した主体によって自律的に構成され，運用されているため，そのスキーマはその構造や実体の表現に違いがある．本章で説明する問合せ処理法においては，スキーマの構造および属性の意味 (semantics) に関しては対応表を与えるという仮定をおいている．データベーススキーマの構造および属性の意味はその設計者によって異なるため，その知識を持つ人間が写像を与えるのは自然な仮定であると考えられる．また本問合せ処理手法においては，実体の同一性に関しては，意味の対応表を与えるという仮定から，その属性値の一致によってその同一性を規定するものとする．スキーマの構造および属性の意味の写像を与えるという仮定の下では，意味の異なる属性は異なる名前の属性に写像される．つまり，この仮定の下ではスキーマにおける属性の有無の問題がスキーマ統合の本質的な問題となる．異なる要素データベースのスキーマに上述の写像を施した後の例を図 3.2(a) に示す．

問合せは仮想的に構成される仮想データベース (VDB) に対して発行される．この VDB の定義は 3.1.4 節でなされる．VDB は物理的に分散されたデータベースから構成され，VDB に対して発行された問合せは 3.1.3節で定義される仮想関数従属性 (VFD) を用いることによって変形され，各要素データベースに対して発行される．システムは各要素データベースから得られた解を併合

DB_1			DB_2	
国籍	著者	タイトル	著者	タイトル
英国	Thomas	データベース	Thomas	オペレーティングシステム
英国	Smith	情報検索	Smith	信号処理
米国	Robert	人工知能	William	データベース

(a) 要素データベース

VDB			
タプル ID	国籍′	著者′	タイトル′
1	英国	Thomas	データベース
2	英国	Smith	情報検索
3	米国	Robert	人工知能
4	ω_1	Thomas	オペレーティングシステム
5	ω_2	Smith	信号処理
6	ω_3	William	データベース

(b) 仮想データベース

図 3.2: 例で用いられる要素データベースと仮想データベース

し，ユーザに示す．

3.1.2 問合せ処理の例

図 3.2(a) に示す二つの要素データベース DB_1，DB_2を考える．この DB_1, DB_2は内容はよく似たスキーマの文献データベースではあるが，独立した主体によって運用されている．ユーザ側の観点からは，それぞれ別々にアクセスするよりも，図 3.1のような何らかのシステム的なメカニズムによって，両者の区別を意識せずあたかも一つのデータベースにアクセスするようにして，問い合わせができると都合がよい．図 3.2(a) からもわかるように DB_1, DB_2のスキーマは似ているが，DB_1にのみ関数従属性 “著者 → 国籍” が存在する．この例の場合，VDB は図 3.2(b) のように構成でき，VDB のスキーマの属性,「著者′」,「国籍′」の間には VFD として “著者′ → 国籍′” なる関係が成立する．ここで「著者′」,「タイトル′」などはそれぞれ要素データベース DB_1, DB_2 上の「著者」,「タイトル」を VDB 上に写像したものであり，$\omega_1, \omega_2, \omega_3$は空値を表す．以下の議論で便利なように VDB にはタプル ID を付加してある．

ここで「英国籍の著者の文献のタイトルをもとめよ」という問合せ処理を考える．まず，VDB 上のタプル ID1 および 2 に対応するタプルが DB_1から得られる．VFD と DB_1からの検索結果を用いることにより，Thomas および Smith の国籍が英国であることがわかり，DB_2からタプル ID4 および 5 に対応する解が得られる．これは VDB において空値ω_1およびω_2が英国と演繹されることに相当する．この結果，解は “データベース”，“情報検索”，“オペレーティングシステム”，“信号処理” となる．もし，同じ問合せを DB_1および DB_2に対して個別に発行したとすると，“データベース”，“情報検索” という二つの解しか得られないのに対し，本手法では論理的に正しくより大きな解集合を求めることができる．

3.1.3 VFD の定義

本論文で述べる問合せ処理手法では，仮想関数従属性 (VFD) を用いる．関数従属性とは関係データベースの属性間に成立する従属性の一つで，関係スキーマ Rと，Rに含まれる属性集合の部分集合 X, Yに対して Xに含まれる属性の値を全て定めると Yに含まれる属性の値が一意に定まるとき，属性集合 Xから属性集合 Yに対して関数従属性が存在するという．この関数従属性は関係データベースの正規化において重要な役割を果たす．

ここで以下の議論のための記号を定義しておく．要素データベース DB_i上での属性集合を U_i，DB_i上でのある一つの関数従属性を f，DB_i上での関数従属性の集合を FD_i，FD_iによって論理的に含意される関数従属性の集合 (FD_iの閉包) を FD_i^+ で表す．関数従属性は右辺がただ一つの属性しか持たないものに限定する．これは関数従属性に関する和と分解規則 [92] を適用することによって一般性を失わないことは明らかである．

関数従属性が個々の要素データベース上で成立する従属性であるのに対して，VFD はそれらを仮想的に統合した仮想データベース上で成立する従属性である．VFD は要素データベース上の関

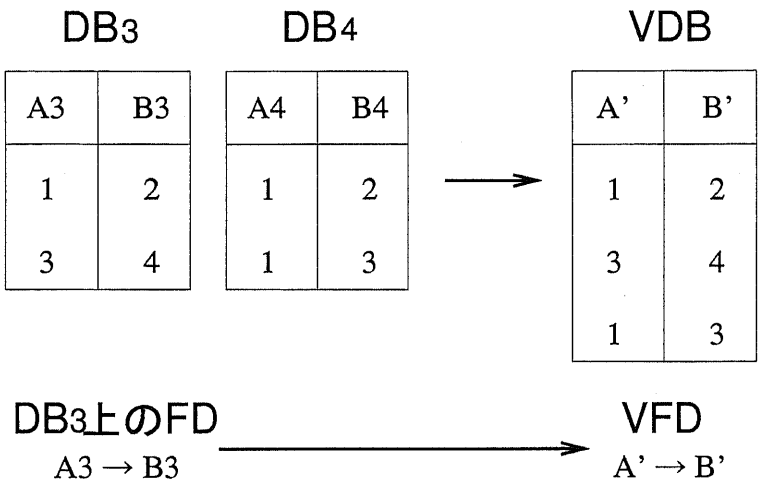


図 3.3: VFD の定義によって VDB 上に矛盾が現れる例

数従属性の集合から構成されるが，その定義のしかたによっては本研究で提案する問合せ処理において効果がなかったり，構成される VDB 上のスキーマが矛盾を含むものとなってしまう．

例えば式 (3.1) のように，すべての要素データベース上に存在する関数従属性を VFD として定めた場合，本研究の問合せ処理では求め得る解は個々の要素データベースに別々に問合せを発行した場合と，解集合の大きさは変化しない．また，式 (3.2) のように，ある要素データベース上に存在する関数従属性を VFD と定めた場合，3.1.4節で述べる VDB の属性定義からわかるように VDB のスキーマとその上のタプル間に矛盾が生じる．例えば図 3.3で，式 (3.2) のように VFD を定めるとすると，VFD は $A' \rightarrow B'$ となるが，これは VDB のタプル $(A', B') = (1,2), (1,3)$ と明らかに矛盾する．

$$VFD = \{f \mid (\forall i)(f \in FD_i^+)\} \tag{3.1}$$

$$VFD = \{f \mid (\exists i)(f \in FD_i^+)\} \tag{3.2}$$

そこで，本論文では VFD を以下の式 (3.3) のように定義する．式 (3.3) による VFD は直観的には注目する属性が存在するすべての要素データベース上で成立する関数従属性の集合である．

[定義 4]

$$VFD = \{f \mid (\forall i)(attr(f) \subseteq U_i \Rightarrow f \in FD_i^+)\} \tag{3.3}$$

ここで，関数従属性 f の両辺に現れる属性の集合を $attr(f)$ とした．

3.1.4 VDB の定義

スキーマ統合における VDB の概念は直観的には物理的に分散しているデータベースを合成するというもので，そのスキーマは各要素データベースのスキーマの和集合となる．なお，本論文ではデータベースのスキーマは各データベース上の属性の集合によって構成されるものとする．

以下の議論では，属性を $A_1A_2\cdots A_n$，属性の集合を X で表すものとする．タプル t，属性集合 X に対して $t[X]$ は属性集合 X からなる t のサブタプルを表す．例えばタプル $t=(v_1,v_2,\cdots,v_n)$ に対して $t[A_2A_3]$ は (v_2,v_3) である．また，ω と c が表現する実体が等しいことを述語 $eq(\omega,c)$ で表すものとする．

本研究では対象とするデータベースを関係データベースとしているため，対象とするデータベースは以下のような 3 項組で表現できる．

[**定義 5**] データベース DB はスキーマ R，タプルの集合 r，関数従属性の集合 FD を用いて以下の 3 項組で表す．

$$DB=\langle R,r,FD\rangle \tag{3.4}$$

[**定義 6**] i 番目の要素データベースを DB_i，DB_i のスキーマを R_i，VDB のスキーマを R' と表す．このとき，

$$R'=\bigcup_i R_i \tag{3.5}$$

[**定義 7**] DB_i のタプルの集合を r_i，そのタプルの集合に対応する VDB のタプルの集合を r'_i と表す．i 番目のデータベース中のタプル集合 r_i 中の j 番目のタプルを t_{ij} と表すとする．このとき，

$$t'_{ij}[A_k]=\begin{cases} t_{ij}[A_k] & A_k\in R_i\text{の場合} \\ \omega_{ijk}(\text{空値}) & \text{その他の場合} \end{cases} \tag{3.6}$$

である．但し，属性値の表現としてのそれぞれの空値は互いに異なるものとするつまり，$\omega_{ijk}\neq\omega_{i'j'k'}$ $(\forall i\neq i',j\neq j',k\neq k')$ である．

定義 7においては属性値の表現としての空値が互いに異なることを言及している．しかしながら，問合せ処理の際には VFD を用いることによって空値と実際の値，あるいは空値と空値が表現する実体が等しいと演繹される場合もあり得る．つまり，ある $\omega\in\Omega$ と $c\in C$ について $eq(\omega,c)$ が演繹されることがある．定義 12における閉領域公理の定義に注意されたい．

[**定義 8**] VDB のタプルの集合を r' と表すものとする．このとき，

$$r'=\bigcup_i r'_i \tag{3.7}$$

[**定義 9**] VDB を以下の 3 項組で定義する．

$$VDB=\langle R',r',VFD\rangle \tag{3.8}$$

VDB は実際に構成されるのではなく仮想的なものであり，実際には問合せは分解され各要素データベースに対して発行される．

3.2 仮想データベースの形式化

3.2.1 述語論理による形式化

VDB 上での VFD を用いた問合せ処理法によって得られる解集合は，VDB を構成せずに個々の要素データベースへ個別に問合せを発行した結果得られる解集合よりも大きくなる．これは関数従属性が，要素データベースにおいては空値となっている属性値を，他のデータベースから導出するためのルールの役割を果たすためである．そこで，この大きくなった解集合の正しさが問題となる．本節では VFD を適用して得られた解も論理的に正しく矛盾を含まないことを示す．論理的な正しさを示すために，まず VDB を論理式を用いた公理によって表現し，問合せとそれに対する解も論理式で表現し，解の正しさを証明する．まず最初に VDB の公理化を行う．

[定義 10] 関係言語 RL とはアルファベットと整合論理式の組である．

[定義 11] r を m 項関係を表す述語とし，Uを r の属性集合とする．Uの空でない部分集合 $A_1A_2\cdots A_k$，Uの要素 Bに対して，関数従属性は $A_1A_2\cdots A_k \rightarrow B$と表され，論理式では以下のように表される．

$$\forall x_1\cdots x_m, \forall y_1\cdots y_m\ r(x_1,\cdots,x_m)\wedge r(y_1,\cdots,y_m)\wedge eq(x_{A_i},y_{A_i})\wedge\cdots\wedge eq(x_{A_k},y_{A_k}) \Rightarrow eq(x_B,y_B) \quad (3.9)$$

ここで，x_{A_i}, y_{A_i}は属性 A_iに対応する変数を表す．

[定義 12] $C=\{c_1,c_2,\cdots,c_{n_c}\}$ は定数を，$\Omega=\{\omega_1,\omega_2,\cdots,\omega_{n_n}\}$ は空値を，$\neg, \vee, \wedge, \Leftarrow, \vdash$ はそれぞれ否定，論理和，論理積，論理的含意，論理的帰結を表す．有限個の定数 k ($k\in C\cup\Omega$)，有限個の変数，述語 $p_1, p_2, \cdots, p_l$および等価関係を表現する述語 eqに対して式 (3.10)–(3.17) で示される条件が成立するとき，一階の公理 Tは関係言語 RL の関係理論 (*relational theory*) であるという．

- 閉領域公理 [69]:

$$eq(x,c_1)\vee\cdots\vee eq(x,c_{n_c})\vee eq(x,\omega_1)\vee\cdots\vee eq(x,\omega_{n_n}) \quad (3.10)$$

- 唯一名公理 [69]: $c_i, c_j\in C(i\neq j)$ なる任意の定数に対して

$$\neg eq(c_i,c_j) \quad (3.11)$$

- 等号公理:

$$\text{反射律}\quad : eq(x,x) \quad (3.12)$$

$$\text{交換律}\quad : eq(x,y)\Leftarrow eq(y,x) \quad (3.13)$$

$$\text{推移律}\quad : eq(x,z)\Leftarrow eq(x,y)\wedge eq(y,z) \quad (3.14)$$

$$\text{代入律}\quad : \text{任意の } m \text{ 項述語 } p \text{ に対して，}$$

$$p(y_1, y_2, \cdots, y_m) \Leftarrow p(x_1, x_2, \cdots, x_m) \wedge eq(x_1, y_1) \wedge \cdots \wedge eq(x_m, y_m) \quad (3.15)$$

- 組公理:等価関係を表す述語 eqを除いた任意の述語 p に対して，Tは基底アトムの集合 K_pを含む.

$$\begin{aligned} K_p = \{ \quad & p(k_1^1, k_2^1, \cdots, k_m^1), \\ & p(k_1^2, k_2^2, \cdots, k_m^2), \\ \cdots, \quad & p(k_1^n, k_2^n, \cdots, k_m^n) \quad \} \end{aligned} \quad (3.16)$$

- 補完公理:任意の述語 p に対して Tは p についての補完公理を含む.

$$\begin{aligned} & p(x_1, x_2, \cdots, x_m) \Rightarrow \\ & \quad \{(eq(x_1, k_1^1) \wedge \cdots \wedge eq(x_m, k_m^1)) \vee \\ & \quad (eq(x_1, k_1^2) \wedge \cdots \wedge eq(x_m, k_m^2)) \vee \\ & \quad \cdots \vee \\ & \quad (eq(x_1, k_1^n) \wedge \cdots \wedge eq(x_m, k_m^n))\} \end{aligned} \quad (3.17)$$

[定義 13] M_{FD}を以下のプログラム P_Tの最小エルブランモデル [45] とする.

- 等号公理: 式 (3.12)–(3.15)
- 組公理: 式 3.16)
- 関数従属性公理: 式 (3.9) で示される関数従属性の集合

これらの論理式は確定プログラム節 (definite program clause) を構成するため，最小エルブランモデルを持つ [45] ことに注意されたい.

3.2.2 仮想データベースの無矛盾性

Reiter[70] によってすべての関係理論は無矛盾であることが示されている．しかしながら，本研究では問合せ処理で関数従属性を用いるため，関係理論 Tと関数従属性 FDを組み合わせた $T \cup FD$が無矛盾であることを示す必要がある．そこでまず，$T \cup FD$が無矛盾であるための必要十分条件を示す.

[定理 1] 関係理論 T，関数従属性の集合を FDとすると M_{FD}が $eq(c_i, c_j), (i \neq j)$ を含まないとき，またそのときに限り $T \cup FD$は無矛盾である.

(証明) $(\rightarrow)$ $eq(c_i, c_j) \notin M_{FD}$であるとき，$M_{FD}$が $T \cup FD$のモデルであることを証明する．これは M_{FD}が等号公理，組公理，関数従属性公理からなるプログラムのモデルであり，これらが M_{FD}を満足することから自明である．Tのドメインは $C \cup \Omega$であり，等号公理の反射律から閉領

域公理が満足されることがわかる．仮定より，唯一名公理は解釈により明らかに満足される．補完公理については，一般性を失わずに一つの述語 p を考えることができる．E_pを M_{FD}中の p の基底アトムの集合とする．$p(d_1, d_2, \cdots, d_m) \in E_p$が組公理の中にあれば補完公理の右辺は明らかに満足される．$p(d_1, d_2, \cdots, d_m)$ が E_p中に存在しない場合，M_{FD}の定義よりある基底アトムが存在し，$p(c_1, c_2, \cdots, c_m), eq(c_1, d_1) \wedge eq(c_2, d_2) \wedge \cdots \wedge eq(c_m, d_m)$ となる．これにより補完公理は満足される．

($\leftarrow$) M_{FD}が $eq(c_i, c_j)(i \neq j)$ を含むとする．このとき $T \cup FD \vdash eq(c_i, c_j)$ である．ところが，唯一名公理より，$T \cup FD \nvdash eq(c_i, c_j)$ である．故に $T \cup FD$が無矛盾であれば M_{FD}は任意の $c_i, c_j \in C(i \neq j)$ について $eq(c_i, c_j)$ を含まない． □

残念ながら式 (3.3) による VFD の定義でも，形式的には定理1で示された必要十分条件を常に満足するわけではない．しかしながら，定理1で示された，M_{FD}が $eq(c_i, c_j), (i \neq j)$ を含む場合とは，属性名の写像あるいはスキーマの構造の写像の問題などで，要素データベース間でデータが矛盾している場合であり，ここで考えている対象外の問題である．つまり，ここでは M_{FD}が $eq(c_i, c_j), (i \neq j)$ を含むことはなく，$T \cup FD$は無矛盾であると考えて差し支えない．

3.2.3 問合せと正解

まず，問合せとそれに対する解の定義を行う．

[**定義 14**] Tを関係理論，τをドメイン，Wを問合せの論理式とする．このとき問合せを式 (3.18) のように定義する．

$$Q = \langle x_1/\tau_1, \cdots, x_n/\tau_n | W(x_1, \cdots, x_n) \rangle \tag{3.18}$$

[**定義 15**] タプル $(c_1, c_2, \cdots, c_n)$ は式 (3.19) の条件を満たすとき Q の解であるとする [70].

$$T \vdash \tau_i(c_i) \quad i = 1, 2, \cdots, n \ \text{かつ} \ T \cup FD \vdash W(c_1, \cdots, c_n) \tag{3.19}$$

[**定義 16**] 関係理論 Tと関数従属性の集合 FDに対して，FDによって拡張された関係理論 T_{FD}は以下のように定義される．

- 閉領域公理:

$$\{eq(x, d) \mid eq(x, d) \in M_{FD}\} \tag{3.20}$$

- 唯一名公理:式 (3.11)
- 等号公理: 式 (3.12)–(3.15)
- 組公理: 等価関係を表す述語 eqを除いた任意の述語 p に対して，

$$\{p(k_1, \cdots, k_m) \mid p(k_1, \cdots, k_m) \in M_{FD}\} \tag{3.21}$$

- 補完公理:式 (3.17)

[定理 2] Tを関係理論，FDを関数従属性の集合とする．$T \cup FD$が無矛盾ならば$T \cup FD$とT_{FD}は等価である．

(証明) M_{FD}の定義より式 (3.20), (3.21) は明らかに$T \cup FD$の論理的帰結である．

T_{FD}のモデルでありかつ$T \cup FD$のモデルではない解釈 I が存在すると仮定する．するとある関数従属性 $X \rightarrow A \in FD$が存在し，これはIを満足しない．なぜならばTはT_{FD}の部分集合であるためである．これは少なくとも組 t_1, t_2が存在し，$eq(t_1[X], t_2[X])$ かつ$\neg eq(t_1[A], t_2[A])$ ということである．しかしながら，もし M_{FD}が $eq(t_1[X], t_2[X])$ を満足する組 t_1, t_2を含むのであれば関数従属性公理より当然 $eq(t_1[A], t_2[A])$ も M_{FD}に含まれるので仮定は矛盾する．従って，$T_{FD} \vdash T \cup FD$が成り立つ． □

T_{FD}はReiter[71] によって定義された空値を含んだ関係理論を構成する．Reiter [71] によれば，関係理論が正リテラルのみからなる場合には式 (3.18) で定義された問合せに対して，式 (3.19) で定義された解を求める正しくかつ完全なアルゴリズムが存在する．

3.3 分散したデータベースへの問合せ処理

3.3.1 問合せ処理

TとFDからT_{FD}をすべて求めれば，問合せ処理にReiterのアルゴリズムを適用することにより正しく，かつ完全な解集合を求めることができる [71]．しかしながら要素データベースのスキーマ統合の過程ではT_{FD}は大きくなる可能性があるため，T_{FD}を求める方法は現実的でない．よってここではすべてのT_{FD}を求めずに解を求める方法を考える．

要素データベース上の属性と仮想データベース上の属性との関係は直観的には要素データベース上の各属性は属性をノード，VFD を有向リンクとして結合され，VDB 上に写像されていると考えることができる．式 (3.3) の VFD の定義より，要素データベースの交わった部分についてのリンクに矛盾がないことは明らかである．

論理的には，完全な解を求めるということはVDB 上で再帰的に VFD のリンクをたどりながらタプルに相当する基底アトムを計算していく必要があり，要素データベース間で推移閉包 [92] を求めることに相当する高価な処理を行う必要があるが，問合せによって参照されるノードに関してVFD が閉路を作らない場合は，分散環境下においてはVFD のパス毎に各要素データベースで並行処理を行うことによって計算時間を短縮することができる．

3.3.2 VFD が閉路を作らない場合のアルゴリズム

関数従属性の左辺および右辺に現れる属性をそれぞれ $lhs(f), rhs(f)$ とする．関数従属性の推移律より，2 つの関数従属性 f, gについて，$rhs(f) \in lhs(g)$ ならば$lhs(f) \cup lhs(g) - \{rhs(f)\}$ か

algorithm GetTheory(VDB, VFD, $\langle x_1/\tau_1, \cdots, x_n/\tau_n | W(x_1, \cdots, x_n)\rangle$);

1. **begin**
2. $T := \{\}$;
3. **for each** 属性$\tau_1, \cdots, \tau_n$を含む要素データベース DB_i **do begin**
4. **for each** Wに現れ，DB_iに含まれない属性 A_j **do begin**
5. **for each** DB_iと A_jに対する *VFD-path* $(f_1 \circ \cdots \circ f_k)$ **do begin**
6. $s_j = (\bigcup_{l_1}\{\sigma_{A_j=W}\pi_{attr(f_1)}R_{l_1} \mid attr(f_1) \subseteq U_{l_1}\}) \bowtie \cdots$
$\bowtie (\bigcup_{l_k}\{\pi_{attr(f_k)}R_{l_k} \mid attr(f_k) \subseteq U_{l_k}\})$;
(従属性 $f_1 \circ \cdots \circ f_k$に現れる属性から構成される関係 s_jを求める．)
7. **end**;
8. **end**;
9. $T := T \cup \pi_{\tau_1,\cdots,\tau_n}(R_i \bowtie s_1 \bowtie \cdots \bowtie s_j)$;
10. **end**;
11. **return** T;
12. **end**;

図 3.4: VFD が閉路を作らない場合のアルゴリズム

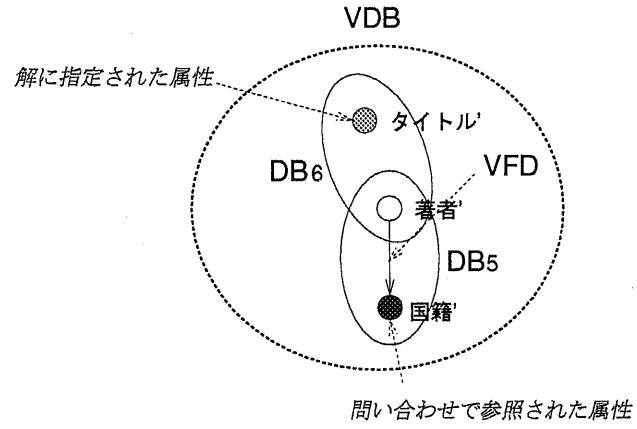
ら $rhs(g)$ への関数従属性が導出される．f, gから推移律によって導出される関数従属性を $f \circ g$で表すことにする．

[定義 17] 各要素データベース D_iと D_i中に現れない属性 A に対して，D_iと A に対する *VFD-path* とは，以下の条件を満たすVFD 中の関数従属性の列 $(f_1, f_2, \cdots, f_n)$ である．

1. $rhs(f_1) = A$
2. 各 i について，$rhs(f_i) \in lhs(f_1 \circ f_2 \circ \cdots \circ f_{i-1})$
3. $lhs(f_1 \circ f_2 \circ \cdots \circ f_n)$ は D_iの属性集合に含まれる．

図 3.4にアルゴリズムを示す．3.1.3節で述べたように，U_{l_k}は要素データベース DB_{l_k}上の属性集合，定義 4で述べたように，$attr(f_l)$ は関数従属性 f_lの両辺に現れる属性の集合，定義 6で述べたように R_iは要素データベース DB_iのスキーマである．また，$\sigma, \pi, \bowtie$ はそれぞれ選択演算，射影演算，結合演算を表す．また，図 3.4の第 6 ステップにおける選択演算$\sigma_{A_j=W}$は，問合せ Wで示された条件から得られた解を推移的に適用していくことを表している．本アルゴリズム中ではVFD を用いており，VFD は式 (3.3) で与えられる．

(3.3) には関数従属性の閉包が含まれるが，アルゴリズム中ではある関数従属性 $f : x \rightarrow y$が FD_i^+に属するかどうかの判定を行えば十分なため，FD_i^+を求める必要はない．この計算は FD_i



DB5: 著者–国籍データベース　DB6: 著者–タイトルデータベース

著者	国籍
Smith	米国
Thomas	英国
Robert	米国

著者	タイトル
Smith	データベース
Robert	論理

図 3.5: VFD が閉路を作らない場合

に関する x^+の計算が従属性の長さの和に比例するように実現することができること [92] を用いると $f : x \to y \in FD_i^+$ の判定は関数従属性の数 N_{FD}，属性の数 N_{AT}に対して $O(N_{FD} \cdot N_{AT})$ で多項式時間での計算が可能である．

ここで問合せ処理の簡単な例を考える．図 3.5は要素データベース DB_5，および DB_6から構成される VDB を表している．この例における VFD は“著者′ → 国籍′”である．ここで「米国籍の著者の文献のタイトルを求めよ」という問合せを考えてみる．このとき図 3.4の第 3 ステップにおける DB_iは DB_6となり，第 4 ステップにおける A は「国籍′」となる．よって，*VFD-path* は“著者′ → 国籍′”となり，第 6 ステップにおいて DB_5から関係 s_j として (著者，国籍) ={(Smith, 米国), (Robert, 米国)} が求まり，第 9 ステップから“データベース”と“論理”が解として得られる．この場合，もしも要素データベース DB_5 と DB_6に独立に問合せを発行したとすると，解は得られない．

3.3.3 VFD が閉路を作る場合

図 3.6のような場合を考える．この例の場合，VFD は $BD \to A$，$AC \to D$，$EF \to B$という超グラフ (hypergraph) であり，閉路を構成する [92]．式 (3.3) の VFD の定義では，一見このような VFD は定義されないように思われるかもしれないが，これは A～F の全ての属性を含む要素データベース DB_{10}が存在し，その上にこのような VFD が存在するという特殊な場合である．このよ

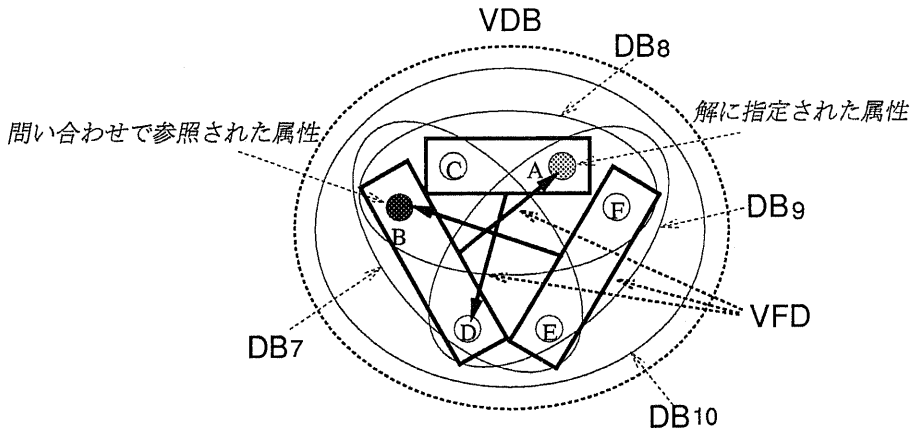


図 3.6: VFD が閉路を作る場合

うな場合に完全解を求めるには VFD をチェックしながらすべての基底アトムを再帰的に求める必要がある．例えば図 3.6において,「B = 1 の場合の A を求めよ」という問合せを考える．3.3.2 節で説明したアルゴリズムを適用することを考えると，図 3.4の第 5 ステップにおける *VFD-path* が閉路を構成するため，その閉路で第 6 ステップにおいて解が増えなくなるまでループする必要がある．これは要素データベース間に渡って，推移的にタプルに相当する基底アトムを計算することであり，推移閉包を求めることに相当するため，現在のところ，これを効率的に計算することは難しいが，VFD の閉路は検索を行う前に検知することができるため，この場合には要素データベースの規模を考慮して完全解を求めるかどうかという戦略を決定すればよい．

3.4 まとめ

3.2節における議論より，統合的アクセスを行おうとする要素データベース間でデータの矛盾がない場合には本問合せ処理で得られる解は正しくかつ完全であることがわかる．つまり，ユーザは一回の問合せでその問合せから得られる正解を全て求めることができるため，これ以上の解があるかどうかを気にする必要はない．

VFD の計算は検索を行うデータベース上の関数従属性の集合から動的に計算される．これは検索を行おうとするデータベースの組合せは膨大な数となる可能性があるため，各データベースの全ての組合せについて計算しておくのは現実的な方法ではないと考えられるためである．さらに，3.3節における議論より，VFD が閉路を構成しない場合には解を効率的に計算することが可能であることがわかる．

この VFD が閉路を作るかどうかは実際に検索処理を行う前に検知が可能であるため，ユーザは問合せを行う目的によって高価な処理を行うかどうかを選択することができる．実際に複数のデータベースを取り扱う際にはその数や規模について考慮する必要があるが，これについては 6 章で議論する．

第 4 章

提案した問合せ処理手法の有効性

3章で述べた問合せ処理方法の有効性について論じる.

4.1 インスタンスベースの統合手法との比較

本手法の特徴である空値を導入したスキーマ統合手法では，スキーマレベルでのデータの一貫性を考えている．一般に，複数データベースの統合プロセスでは，データの規模は大きくなると考えられるが，[40] で提案されているインスタンスルールベースでの統合手法では，データの規模の増大に伴い，インスタンス間の関係を記述するルールも膨大なものとなり，データの蓄積効率が悪くなる上に，処理時間も大きくなる．さらに，本手法では各要素データベース上のスキーマの上で成り立つ関係からシステムが仮想データベース上で成り立つ関係を導き出し，矛盾のないスキーマ統合が行われるのに対して，インスタンスルールレベルの手法では基本的に人間がインスタンス間のルールをインスタンス毎に与えてやる必要があり，この点からもデータの規模が大きくなった場合には実用性に欠ける．

次に，3章で述べたスキーマ統合手法と問合せ処理手法によって得られる解の大きさについて調べる．本問合せ処理手法を用いるということは，仮想データベース上で仮想関数従属性を用いて解を求めるプロセスをルールによる解の演繹と見ることができることからも明らかなように，個々のデータベースに対して問合せを発行する場合よりも解が大きくなることは明らかである．これは3.1.2節の例によっても明らかであるが，これらの論理的に正しい解がどの程度増えるかということも，スキーマ統合および問合せ処理方法の一つの重要な評価基準となる．よって，この解の大きさについて，シミュレーションにより検証を行った．

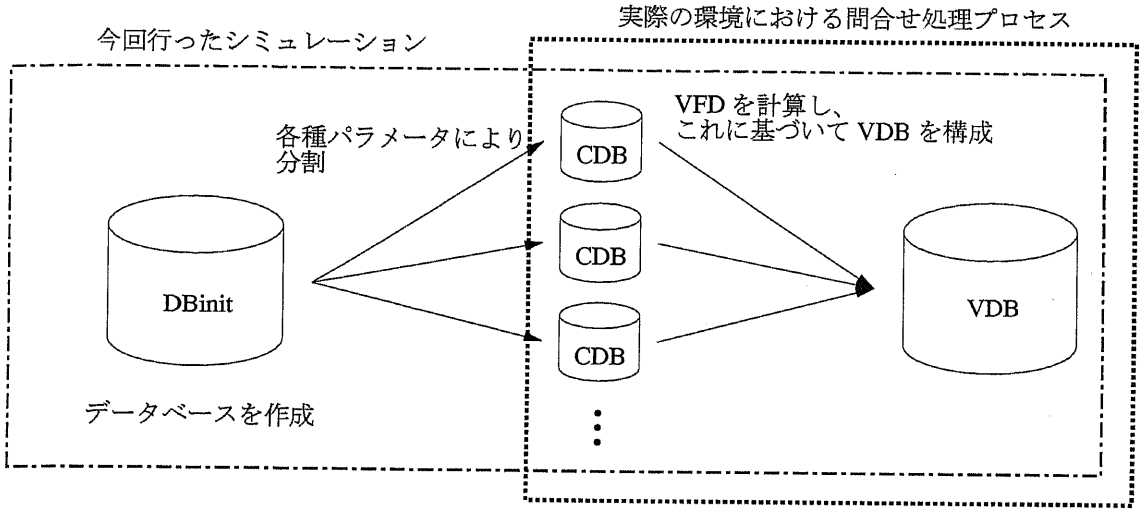
4.2 シミュレーションによる評価

実際には問合せ処理は実存する複数のデータベースに対して行われるものであるが，3章で述べた問合せ処理方法の有効性を検証するためにこれらの実存のデータベースを用いるには以下のような問題点がある．

1. 実際のデータベースを用いた場合にはVFDの数などのいくつかのパラメータの設定が自由に制御できない．これは各データベースにおいてそのスキーマはデータベースの作成時にあらかじめ定められており，そのスキーマにそう形でデータも作成されていることに起因する．

2. 3章で述べたように，本論文では属性値の一致によって実体の一致を規定している．実際のデータベースを使用した場合には表記の揺れ，属性のマッチング等の問題が現れるが，これは3章で述べた問合せ処理の有効性を議論する際には除外したい問題である．

これらのことを考慮して，本章で行う実験では実存のデータベースではなく，実験用に作成したデータベースに対してシミュレーションを行うという方法をとった．



CDB : 要素データベース (component database)

VDB : 仮想データベース (virtual database)

VFD : 仮想関数従属性 (virtual functional dependency)

DBinit : 最初に構成したデータベース

図 4.1: 実験の概要

4.3 実験の概要

シミュレーションを行うにあたり，3章で述べた問合せ処理方法が実際の分散環境におけるデータベースで有効であることを示す必要がある．つまり，シミュレーションの環境をなるべく現実に近いように設定する必要がある．実際の問合せ処理では統合的にアクセスを行おうとする各要素データベースの情報を用いることによって仮想データベース (以下 VDB) を計算することになるが，このシミュレーションでは簡単のため最初にあるデータベース (以下 DB_{init}) を構成し，それをいろいろなパラメータに応じて要素データベース (以下 CDB) に分割し，さらに仮想関数従属性 (以下 VFD) を用いて VDB の再構成を行うという方法をとっている (図 4.1参照)．

実際の環境に近付けるためには DB_{init} の分割の際にデータベースの垂直分割および水平分割を考慮する必要がある．つまりドメインの分割を行うと共に，何らかの方法でタプルを間引く必要がある．単純に考えると DB_{init} を VDB とみなし，分割を行った結果できたデータベースを CDB とみなせばよいように思われるが，こうした場合には DB_{init} から CDB 上のスキーマへの射影によって，CDB 上のタプルでは DB_{init} 上のタプルのいくつかの情報が欠落してしまうことになり，問合せ処理によって得られる解の大きさの比較は無意味なものとなってしまう可能性がある．そのため，本シミュレーションでは DB_{init} の分割によって構成された複数の CDB から新たに VDB を構成し，この VDB と CDB の集合を比較することにより，分割による情報の欠落がシミュレーション結果へ及ぼす影響を取り除いている．また，分割の際に DB_{init} 上の関数従属性を保存する

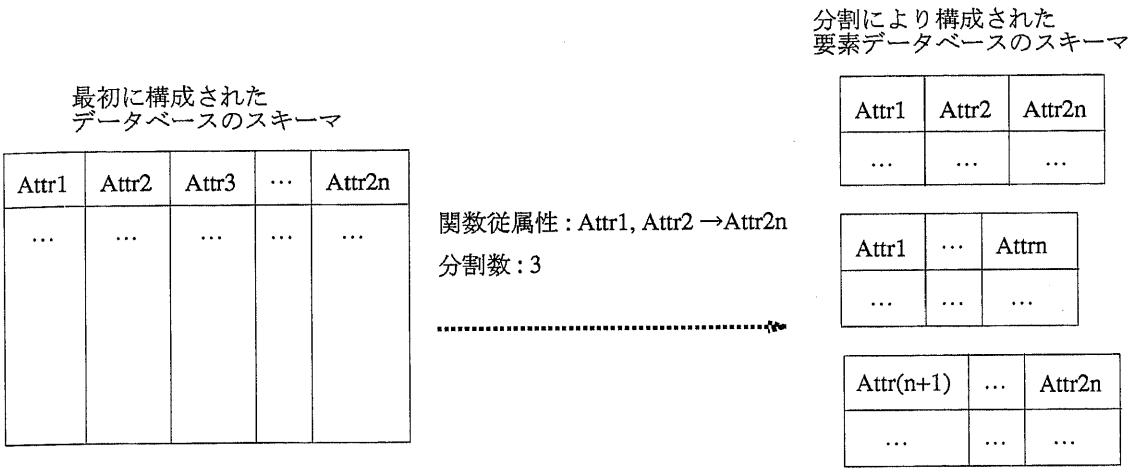


図 4.2: 分割の戦略

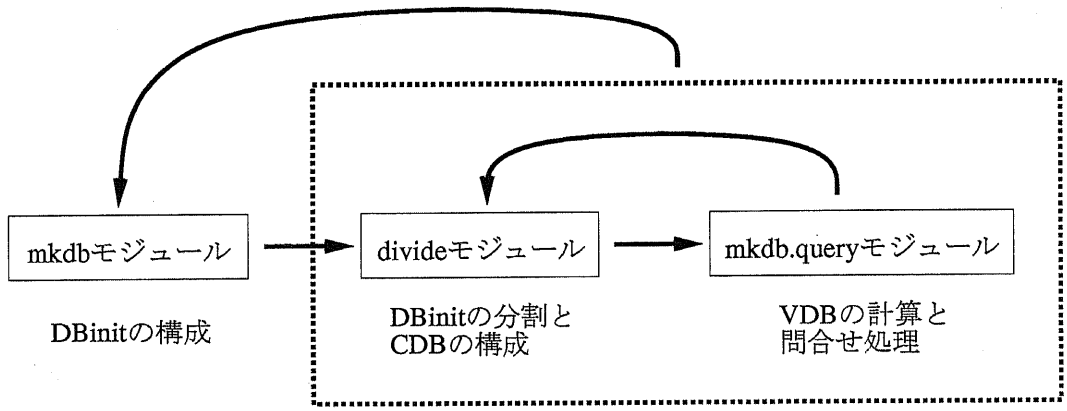


図 4.3: シミュレーションプログラムのブロック図

ために，以下のような分割の戦略を採用している．(図 4.2参照)

1. DB_{init}上の関数従属性の両辺に現れる属性のみを含む CDB をまず作る．

2. 次に何個の CDB に分割するかに応じてDB_{init}を等分割して CDB を作る．

この結果，分割の詳細は表 4.1のようになっている．

シミュレーションプログラムは約 8000 行の C 言語で書かれており，これを SUN SPARC center 2000 上で動作させた．シミュレーションプログラムのブロック図を図 4.3に示す．mkdb モジュールでは先に述べたようにDB_{init}を構成する．この構成の際の留意点については 4.4節で述べる．divide モジュールでは分割の戦略でも述べたようにDB_{init}上の関数従属性を保存するようにまずドメインを分割した後，パラメータとなるある確率に従ってタプルを間引く処理を行う．このようにし

表 4.1: VDB の分割

VFD の数	分割数	VFD を保存する分割	等分割
6	4	3FD*2	2
6	6	2FD*3	3
6	10	1FD*4+2FD*1	5
9	4	5FD*1+4FD*1	2
9	6	3FD*3	3
9	10	2FD*4+1FD*1	5
15	4	8FD*1+7FD*1	2
15	6	5FD*3	3
15	10	3FD*5	5

て構成された複数のデータベースから重複するタプルを削除しCDB とする．次に mkdb.query モジュールでは VDB の計算と問合せ処理を行う．実際の分散環境においては VDB は問合せ処理の際に動的に計算するが，本シミュレーションにおいては処理を高速化するために各パラメータごとに VDB を予め計算しておくという手法をとっている．

4.4 シミュレーションのパラメータ

DB_{init}の全属性数は 30 で，実際のデータベースの条件に少しでも近付けるために，INSPEC, COMPENDIX 等の実際に使用されているいくつかの文献データベースのスキーマを調べ，30 個の属性のうち，2 値の値をとるものを 1 個，残りはとり得る値の幅がそれぞれ 100, 200 種となる属性をほぼ同数用意し，この条件を満たすDB_{init}上のタプルを 10000 個準備した．また他のパラメータとして，分割される CDB の数を 6，VFD の数を 3, 6, 9, 12, 15，各 CDB の持つ属性数を 16, 20, 25, 28，問合せの条件で参照する属性数を 1, 2, 3 と変化させた．

また，CDB 上の属性の分布だけでなく，タプルデータそのものの分布に対する本手法の有効性を示すために Copy Rate というパラメータを導入した．Copy Rate は以下の式 (4.1) で与えられる．ここで，πは射影演算，$|DB|$ は DB 中のタプル数を表す．(R_iについては定義 6を参照)

$$CopyRate = \frac{|DB_i|}{|DB_{init}|} \left(= \frac{|\pi_{R_i} DB_{init}|}{|DB_{init}|} \right) \tag{4.1}$$

これは直観的には各 CDB 間におけるデータの重複度を表す．例えば，DB_1の Copy Rate が 20%であるとはDB_{init}のタプル中の 20%をDB_1が保持していることを示す．本シミュレーションにおいては Copy Rate は 20, 40, 60, 80, 100%で変化させ，タプルの分配は乱数によって行うこととした．以上の各パラメータを表 4.2にまとめる．

表 4.2: シミュレーションにおけるパラメータ

パラメータ	とり得る値
要素データベース数	6
要素データベースの持つ属性数	16, 20, 25, 28
DB_{init} のタプル数	10000
仮想関数従属性の数	3, 6, 9, 12, 15
問合せで参照する属性数	1, 2, 3
Copy Rate	20, 40, 60, 80, 100 %
ドメインのとり得る値の大きさ	(100,200)

問合せは参照する属性，その値を乱数によって設定し，90000 回発行した．例えば，VFD の数を 15，CDB の数を 6，それらの持つ属性数を 16 とすると，DB_{init} 上には 30 個の属性があり，その属性間には 15 個の相異なる関数従属性が存在する．これらの属性のうちそれぞれ乱数によって選ばれた 16 個の属性が 6 個の CDB 上に射影されるということである．

4.5 実験結果

図 4.4〜4.7に示すグラフの縦軸はユーザの与えた問合せを各要素データベースに個別に問合せを発行した結果の総体から得られる解の個数に対する，VFD を適用して得た解の個数の比を表している．つまり個々の CDB に対して問合せを発行して得られた場合の解の大きさを 1 とした場合に，3章で述べた問合せ処理方法で得られる解の大きさを表している．まず表 4.3に図 4.4〜4.7に示す各シミュレーションにおいて変化させたパラメータ，固定したパラメータをまとめる．

図 4.4は問合せで参照する属性数を変化させた場合の解の比率を表している．問合せで参照する属性が多くなるに従い，その問合せで参照する全ての属性が，ある CDB 上に存在する確率は小さくなるため，3章で述べた手法の効果が大きく現れることがわかる．次に図 4.5は CDB 上の属性数を変化させて場合であり，データの垂直分散に対する問合せ処理手法の有効性を調べるために行った実験．本実験においては VDB 上の全属性数は 30 であるが，3章で述べた手法は CDB 上の属性がある程度の重なりをもって分布している場合に効果が大きいことがわかる．図 4.6は式 (4.1) で示した Copy Rate を変化させた場合で，これはデータの水平分散に対する問合せ処理の有効性を調べるために行った実験である．本実験の環境では Copy Rate が 40〜60%程度で 3章で述べた手法の効果が最もよく現れている．本問合せ処理手法では問合せ処理の際に VFD が値を演繹するルールの役割を果たすため，VFD の数が多くなるほど得られる解が大きくなるのはほぼ自明であるが，図 4.7ではこれが検証されている．これらの実験から実際に問合せ処理において取り扱うと考えられる情報資源の状態を比較的よく再現していると思われる環境において，個々

表 4.3: 各グラフにおけるパラメータ

	変化させたパラメータ	固定したパラメータ
図 4.4	問合せで参照する属性数 1, 2, 3 Copy Rate 40, 60, 80	VFD 数 15 CDB 上の属性数 20
図 4.5	CDB 上の属性数 16, 20, 24, 28 Copy Rate 40, 60, 80	VFD 数 15 問合せで参照する属性数 3
図 4.6	Copy Rate 20, 40, 60, 80, 100 問合せで参照する属性数 1, 2, 3	VFD 数 15 CDB 上の属性数 20
図 4.7	VFD 数 3, 6, 9, 12, 15 Copy Rate 40, 60, 80	問合せで参照する属性数 3 CDB 上の属性数 20

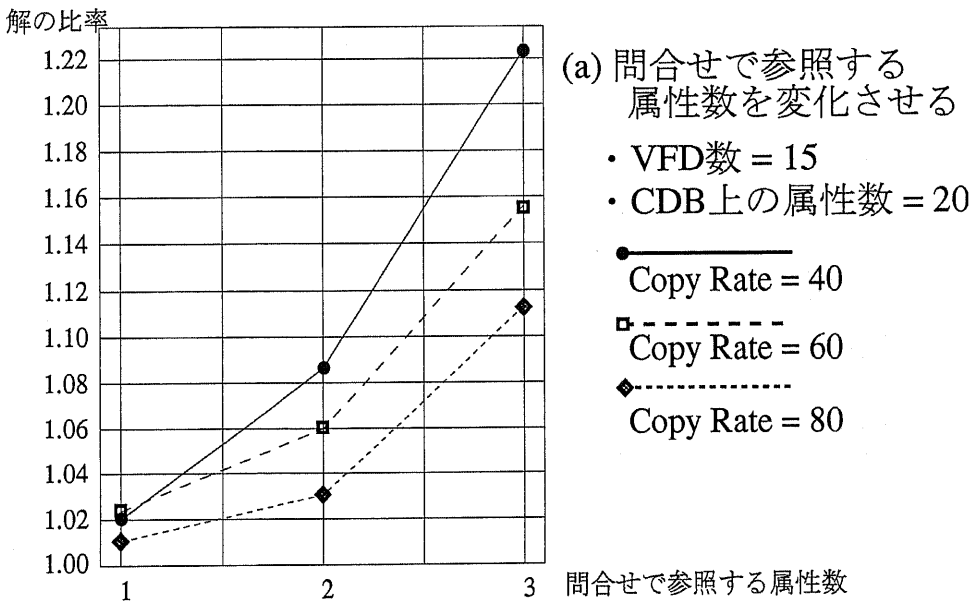


図 4.4: 問合せで参照する属性数を変化させた場合

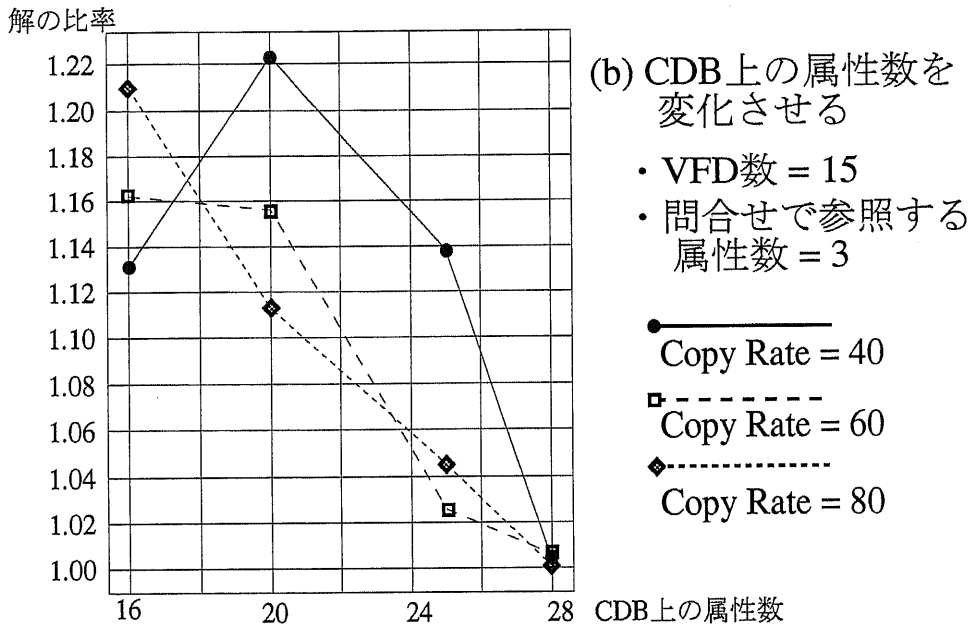


図 4.5: 要素データベースの持つ属性数を変化させた場合

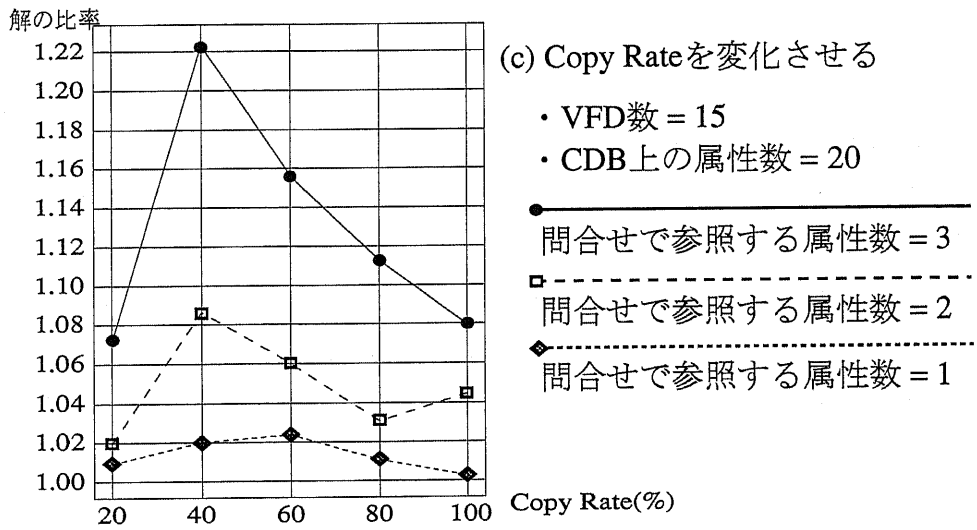


図 4.6: Copy Rate を変化させた場合

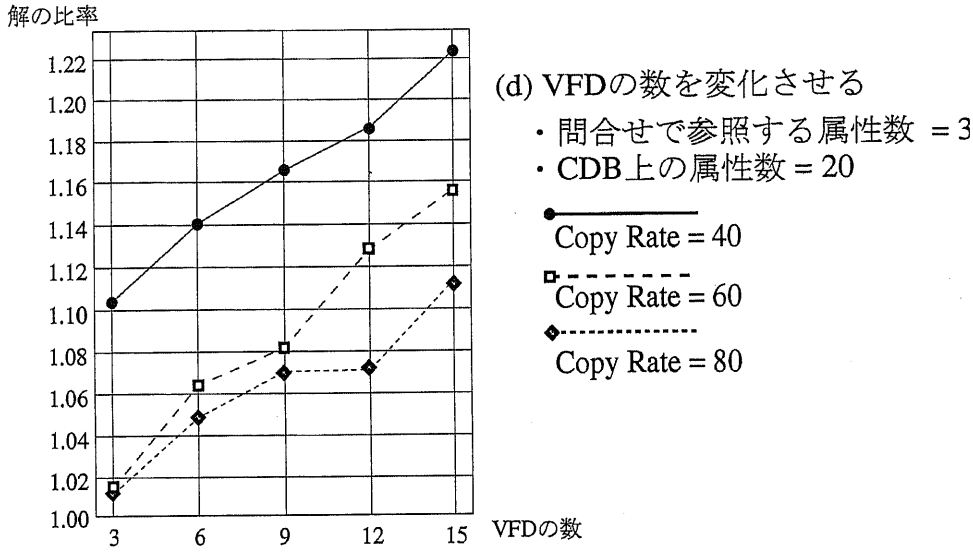


図 4.7: VFD の数を変化させた場合

の CDB に対する通常の問合せ処理と比較して最高 20%程度大きな解集合を得ることができることが示された.

4.6 まとめ

3章で述べた，属性が異なる複数のデータベースに対する統合的アクセスを実現するためのスキーマ統合手法および問合せ処理手法に関する有効性を検証する意味で，まずインスタンスベースのスキーマ統合手法との定性的な比較を行い，さらに得られる解の大きさに関するシミュレーションを行った．本手法では VDB を構成することにより，ユーザは属性の不整合を考慮せずに問合せを記述することができる．VDB に対して発行された問合せは VFD を用いることによって各 CDB に対して発行され，ユーザの検索の論理的意味を損ねることなく，個別に問合せ処理を行う場合よりもより多くの解を求めることができる．特に VDB 上で VFD が閉路を作らない場合には，分散処理によって正しくかつ完全な解を効率的に得ることができる．

4.2節で述べたように，問合せ処理は実際には実存するデータベースに対して行われる．シミュレーションで設定した各条件はできるだけ実際のデータベースの状態を表現するようにつとめたが，以下の点で実際のデータベースとは異なっていると考えられる．

- 属性名，属性値のマッピングが行われていること
- データの分布

マッピングの問題に関しては，問合せ処理の有効性を議論する際にあえて除外した問題であり，本研究ではこの問題は問合せ処理の外で取り扱う問題ととらえる．この点については 6 章で，全

体のシステム構成について述べる際に議論する．次にデータの分布に関しては，本シミュレーションにおいては属性のタイプごとにその値の幅を定めた一様分布となっている．一般的にデータの分布はその取り扱おうとする対象に応じて様々であり，一概に議論できるものではない．例えば対象として名前などを想定したとすると，この場合にはZipf分布[1]が測定値と近い分布を示すと報告されているが[25]，日本語ではこれをそのまま適用することはできず，その他にも決定的な分布が提案されているわけではないこともあり，今回のシミュレーションにおいては各属性の値の分布は定義域の大きさを変化させた一様分布とした．

4.5節でも述べたように，シミュレーションの結果，VFDを用いた問合せ処理では，各要素データベースに個別に問合せを発行した場合よりも，より大きな解集合を得られることが確かめられた．3章での議論より明らかなように本問合せ処理で得られる解は要素データベース間でデータの矛盾がない場合には論理的には正しい．ユーザは自分の既知の情報から更に多くの情報を得ることができるが，この問合せ処理は完全であることから，論理的に正しい情報がこれ以上存在するかどうかを心配する必要はない．

[1]英語の単語の出現頻度に関する性質で，単語を出現頻度の大きい順に並べ順位番号を1, 2, ･･･とつけると，順位番号×出現頻度 = 一定という関係が成り立つという経験則[59].

第5章

問合せ処理の応用

3章で述べた問合せ処理方法の応用について考える.

5.1 応用の対象に関する検討

分散する情報の統合利用というと漠然としているが，3.1 節で述べたように，本研究では属性が異なる複数のデータベースが存在し，これにユーザが統合的にアクセスを行うという環境を想定している．このために3章で問合せ処理手法を提案したが，本章ではこの応用対象について議論する．そこでまず，応用対象が満たす要件として以下のような条件を考慮した．

1. 実際に広く用いられていること．

2. 統合利用する価値があること．

3. 属性の対応表が与えられること．

1, 2については工学的な立場からの要件であり，この 2 つの要件を満たさないような対象を選んでも実際に役には立たないと思われるため，まずこの 2 点を検討した．

次に3について考える．3章で述べた問合せ処理手法では，3.1.1節で述べたように,「スキーマの構造および属性の意味 (semantics) に関しては対応表を与え，属性値の一致によってその同一性を規定するものとする.」という仮定をおいている．つまり，同じ意味 (semanctics) の属性は同じ名前に写像されている必要がある．実際に複数の異種データベースを取り扱うシステムの構成を考える際には，この対応表の作成方法，管理の仕方などについての議論が不可欠であるが，ここではこの対応表があらかじめ与えられるような対象，つまり同じ意味の属性は同じ名前に写像されているような複数のデータベースが存在すれば都合がよい．

これらの要件を考慮した結果，個人的に作成する文献データベースの一種である BIBTEX データベースを対象とすることにした．BIBTEX は広く一般に用いられている文書組版システムである LATEX[38] により利用される参考文献情報を整理しておくためのシステムであり，広く一般に用いられている．しかもこれは文献データベースの一種と考えられることから，統合利用する価値は非常に高いと考えられる．これらのことから前述の要件の 1, 2を十分に満足している．また，5.2節で詳しく説明するが，BIBTEX では複数の文献の型に対して，その属性の集合が定められており，論理的には属性名の写像が行われてた複数の文献データベースと考えてよい．つまり，これは3の条件を満たしていることを意味する．

5.2 BIBTEX とは何か

BIBTEX とは個人ベースで作成する文献データベースであり，図 5.1に示すような形式で記述される．通常はこのような形式のエントリを持つ複数のファイルに文献情報を蓄積しておき，LATEX で文書を作成し参考文献を参照するとファイル内から対応する文献に関する情報を探しだし，必要な情報を自分の指定した参考文献形式に整形して出力するというもので，論文および文書の作成に LATEX を使っている研究者，学生の間で広く用いられている．

```
% bibtex item file
% classification.bib 分類を取り扱う論文
% 10/18/1995
% C^2ICM は以前に著者らが発表した C^3M という手法に変わるものではなく、
% 文書が増えていく過程での reclustering に対応するための手法。
% 11/02/1995
% ABSTRACT from NACSIS-IR:
% Clustering of very large document databases is useful for both
% searching and browsing. The periodic updating of clusters is required
% due to the dynamic nature of databases. An algorithm for incremental
% clustering is introduced. The complexity and cost analysis of the
% algorithm together with an investigation of its expected behavior are
% presented. Through empirical testing it is shown that the algorithm
% achieves cost effectiveness and generates statistically valid clusters
% that are compatible with those of reclustering. The experimental
% evidence shows that the algorithm creates an effective and efficient
% retrieval environment. (Author abstract) 31 Refs.
@Article{Can.1993,
  author =       "F. Can",
  title =        "Incremental Clustering for Dynamic Information
                  Processing",
  journal =      "ACM Transactions on Information Systems",
  year =         1993,
  volume =       "11",
  number =       "2",
  pages =        "143-164",
  month =        apr,
  note =         "ISSN:1046-8188"
}
```

図 5.1: BIBTEX ファイルの記述例

BIBTEX では文献情報を以下の 14 個の型 (type) に分類する [49]. この分類は使用者の自由裁量であるため, 人によって同じ文献の表現形態が異なることがあり得る. 例えば図 5.1において, Can.1993 というタグを付けられた論文は **Article** という型に分類されている. 文献の型と, その型の文献が持つ属性集合[1]が定められており, これをまとめて表 5.1に示す. 属性には表 5.2に示す 24 個のエントリが存在し [48], plain, alpha, abbrv, unsrt などの BIBTEX の各スタイルに応じて, annote を除いた項目がタイプセットの際に使用される. 各型の文献と属性の間の関係を表 5.2に示す. ただし, スペースの関係で属性は表 5.3で示した略語を使って表現されており, 表中の o はその属性が「必須」であることを示し, x はその属性が「任意」であることを示す. 添字である数字が同じ属性は同一文献情報中で OR の関係であることを示す. ここで, 属性が「必須」であるとか「任意」であるとかの意味は以下のようになっている.

必須 これが欠落していると警告メッセージが出力され, まれにではあるが文献リストが変な形に出力される. もし必須な情報が意味を持たないのであれば, 間違った型を指定しているのである. 必須な情報が意味を持つのであるが, 他の属性にそれが記述されている場合には警告メッセージを無視すればよい.

任意 この属性は, もしあれば使われるが, 何の問題もなしに省略できる. もしこれらの属性の情報が読者を助けるのであれば, これを省略しない方がよい.

5.3 何ができるか

5.2節でも述べたように, BIBTEX は LATEX を用いて論文を作成している研究者の間で広く利用されている文献データベースである. このデータベースは基本的に個人単位で管理されているため, その規模は比較的小さい. また, BIBTEX では人によって, 文献を異なる文献の型として管理している場合があり, その型に応じた情報の入力を行うため, 文献情報が欠落している可能性がある.

ここで構成しようとするシステムは3章で述べた問合せ処理方法を応用することによって, 分散して存在する小さなデータベースをあたかも一つの大きな文献データベースであるかのように利用しようというものであり, ユーザは本システムを介して, これらの文献データベースをネットワークを介して統合的に利用することができる. さらに前述の欠落した情報を VFD を使うことによって「追跡」することができる. ここで注意することは意味的には各文献の型である, **Article**, **Book** 等が一つのデータベースとみなされるのであって, BIBTEX のファイルが一つのデータベースとみなされるのではないということである. BIBTEX への応用を考えるために, 以下のように簡単化を行う.

[1] BIBTEX に関する文書 [48, 49] では "author", "title" などを「フィールド」と呼んでいるが, ここでは「属性」と呼ぶものとする.

表 5.2: 文献の型と属性の関係

	属性																							
Document category	Ad	An	Au	B	Ch	Cr	Ed	Et	H	I	J	K	M	No	Nu	O	Pa	Pu	Sc	Se	Ti	Ty	V	Y
Article			o								o		x	x	x		x				o		x	o
Book	x		o1				x	o1					x	x	x1			o		x	o		x1	o
Booklet	x		x						x				x	x							o			x
Conference	x		o	o				x					x	x	x1	x	x	x		x	o		x1	o
Inbook	x		o1		o		x	o1					x	x	x1		o	o		x	o	x	x1	o
Incollection	x		o	o	x		x	x					x	x	x1		x	o		x	o	x	x1	o
Inproceedings	x		o	o				x					x	x	x1	x	x	x		x	o		x1	o
Manual	x		x				x						x	x		x					o			x
Mastersthesis	x		o										x	x					o		o	x		o
Misc			x						x				x	x							x			x
Phdthesis	x		o										x	x					o		o	x		o
Proceedings	x							x					x	x	x1	x		x		x	o		x1	o
Techreport	x		o							o			x	x	x						o	x		o
Unpublished			o										x	o							o			x

表 5.1: 文献の型

文献の型	説　　明
Article	論文誌など発表された論文
Book	出版社の明示された本
Booklet	印刷，製本されているが出版主体が不明なもの
Conference	inproceedings と同じ (Scribe との互換性のため)
Inbook	書物の一部 (章，節，文など何でも)
Incollection	それ自身の表題を持つ，本の一部分
Inproceedings	会議録中の論文
Manual	マニュアル
Mastersthesis	修士論文
Misc	他のどれにも当てはまらない時に使う
Phdthesis	博士論文
Proceedings	会議録
Techreport	テクニカルレポート
Unpublished	正式には出版されていないもの

1. 属性の意味を考慮して，Annote, Crossref, Key の 3 つの属性は除く.

2. 属性の必須と任意の区別についてはとりあえず考慮しない．必須の属性の値は空値になり得ないと考えるのが妥当である.

これらを考慮して表 5.2を簡略化すると，表 5.3のようになる．表 5.3は仮想データベースのスキーマに相当する．問合せ処理では仮想関数従属性 (以下 VFD) を利用するわけだが，この VFD を構成するためには各型の文献の FD のセットが必要となる．BibTeX の 14 個のカテゴリの文献ついては属性間の意味を考えて FD のセットを与えた[2]．全体については付録 Aに示すが，表 5.4 にサンプルとして **Article** および **Mastersthesis** の 2 つの型の文献についての FD のセットを示す.

5.4 実装をにらんだ検討

実際にシステムを作成するにあたり，BibTeX の以下のような特徴を考慮する必要がある.

1. BibTeX ファイル (`*.bib`) では一つのファイルに複数の異なる文献の型のエントリが存在する.

[2]実際のデータベースではこのセットはあらかじめ与えられていることに注意

表 5.2: 属性の説明

属性名	説　　明
address	出版主体の住所
annote	注釈付きのスタイルで使われる
author	著者名
booktitle	本の名前
chapter	章，節などの番号
crossref	相互参照する文献のデータベースのキー
edition	本の版 (日本語以外では順序数で最初は大文字にする．)
editor	編集者
howpublished	どのようにしてこの奇妙なものが発行されたか (最初は大文字)
institution	テクニカルレポートの発行主体
journal	論文誌名
key	著者名がない時に相互引用，ソーティング，ラベル作成などに使われる
month	発行月，または書かれた月
note	読者に役立つ付加情報 (最初は大文字)
number	論文誌などの番号
organization	会議を主催した機関名あるいはマニュアルの出版主体
pages	ページ (範囲)
publisher	主版社 (者) 名
school	修士，博士論文が書かれたあるいは提出された大学名
series	シリーズ，あるいは複数巻の本の名前
title	表題
type	テクニカルレポートの種類
volume	論文誌，複数巻の本の巻
year	発行年，または書かれた年
yomi	JBibTeX 特有のもの．日本語の author のアルファベット順のソートなどのために利用.

表 5.3: 属性名の略語

正式名	略語	正式名	略語	正式名	略語	正式名	略語
address	Ad	edition	Ed	month	M	school	Sc
annote	An	editor	Et	note	No	series	Se
author	Au	howpublished	H	number	Nu	title	Ti
booktitle	B	institution	I	organization	O	type	Ty
chapter	Ch	journal	J	pages	Pa	volume	V
crossref	Cr	key	K	publisher	Pu	year	Y

表 5.4: 文献の型毎の FD のセット (抜粋)

Document category	FDs
Article	author → yomi
	author, title → journal
	author, title → year
	author, title → month
	author, title → note
	author, title → number
	author, title → pages
	author, title → volume
	journal, year → volume
	journal, volume, pages → author
	journal, volume, pages → title
Mastersthesis	author → yomi
	author, title → year
	author, title → month
	author, title → type
	author → school
	school → address

表 5.3: 簡略化された文献の型と属性の関係

	属性																			
Document category	Ad	Au	B	Ch	Ed	H	I	J	M	No	Nu	O	Pa	Pu	Sc	Se	Ti	Ty	V	Y
Article		o						o	x	x	x		x				o		x	o
Book	x	o			x				x	x	x			o		x	o		x	o
Booklet	x	x				x			x	x							o			x
Conference	x	o	o						x	x	x	x	x	x		x	o		x	o
Inbook	x	o		o	x				x	x	x		o	o		x	o	x	x	o
Incollection	x	o	o	x	x				x	x	x		x	o		x	o	x	x	o
Inproceedings	x	o	o						x	x	x	x	x	x		x	o		x	o
Manual	x	x			x				x	x		x					o			x
Mastersthesis	x	o							x	x					o		o	x		o
Misc		x				x			x	x							x			x
Phdthesis	x	o							x	x					o		o	x		o
Proceedings	x								x	x	x	x		x		x	o		x	o
Techreport	x	o					o		x	x	x						o	x		o
Unpublished		o							x	o							o			x

2. 各人が入力する文献情報のフォーマットはさまざまである.

3. 雑誌名にはBIBTEXで定められた省略形が存在する.

1は各bibファイル中に**Article,Book**などの異なるエントリが存在するということである. 前に述べたように, 論理的にはファイルを一つのデータベースとみなすのではなく, **Article, Book**などの各文献の型を一つのデータベースとみなしているため, まずbibファイルを解析して, 各エントリを取り出す必要がある. 次に2は例えば著者の記述法が異なるとか, 値が "{", "}", "``", "''"で括られているとかの問題である. 後者についてはシステムの説明の際に述べるが, ファイルを読み込む際に単純な処理を行うことによって対応している. 前者はマッチングとの関係もあり, 解決しなければならない問題であり, 現在は理論的な検討を行っている最中である. システムとして完全一致と部分文字列一致を実装することによって対応している. 3についてはBIBTEXで与えられる省略形のマッピングをシステムに与えてやればよい.

5.5 作成したシステム

5.5.1 システムの概要

図5.4に作成したシステムの構成図を示す. システムはクライアント-サーバ形式で構成されており, BIBTEXファイルを保持する複数のサーバに対してユーザ側のVDBを利用するクライアント側がアクセスするという形をとっている. プログラムはC言語で書かれており, SUN SPARC station 20上のSolaris 2.4上で動作を確認した. クライアントとサーバの間の通信通信部分はRPC (Remote Procedure Call) [8, 88]を用い, ユーザが入力した問合せおよびBIBTEXファイルのパーシングにはlex [39]およびyacc[31]を用いた. 現時点でのプログラムの規模は約14000行である. 以下サーバ側とクライアント側, 通信部分についての詳細な説明を行う.

5.5.2 サーバ側

まずBIBTEXファイルを読み込んで字句解析および構文解析を行い, 各属性とその値を抽出し, データベースを構成する. これは5.4節で述べたように, データベースの単位をファイルではなく文献の型にしているためである. 現在データは学術情報センター研究開発部の4名のものを用いており, 件数は約700件である. BIBTEXファイルの解析処理はlex, yaccを使用して記述しているが, この際5.4節で述べた記法の問題で, BIBTEXファイルの性質を考慮して,

- 文字列の最初と最後がそれぞれ"{", "}"だった場合はそれを除いたものを値とする.
- 文字列の最初と最後が最初と最後に "``", "''"がある場合はそれを除いたものを値とする.
- 文字列中の連続する改行, 空白文字を一つのスペースに置き換える.

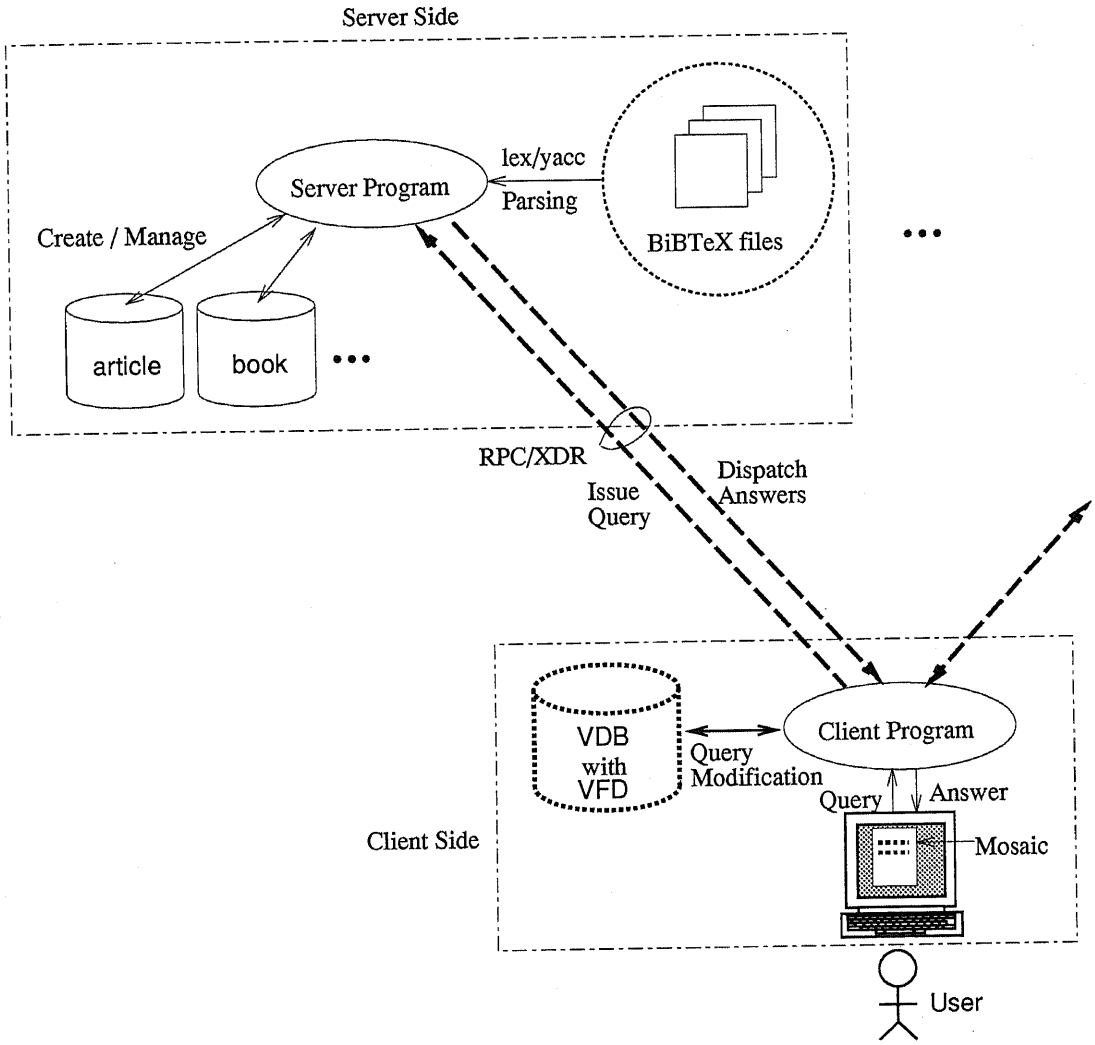


図 5.4: System 構成

などの処理をbibファイルの解析の際にlexを用いて実行している．文字列としての日本語はEUCコードのみに対応しており，文法のチェック機構はyaccを用いて実装している．ファイル中の文法誤りつまりBIBTEXファイルの記述誤りについては間違ったエントリのみを読みとばすという仕様とした．

本システムではサーバとクライアント間の通信にはRPCを用いている．サーバ側はBIBTEXファイルを読み込み，解析してデータベースを構成した後はバックグラウンドで動作し，クライアントからの接続を待つ．クライアントからの接続要求が到着すると，サーバは新たにプロセスをforkする．新たにforkされた子プロセスはクライアントから送られた問合せと構成したデータベースのマッチングを行い結果を返す．マッチングに関しては現在完全一致と部分文字列一致を実装している．

5.5.3 クライアント側

クライアント側ではNCSA Mosaic[56]を利用して，図5.5に示すようなユーザインタフェースを作成した．ユーザは左上のウィンドウで検索を行う文献の型を選択することができ，問合せを属性に対する条件の形，あるいは検索式の形で書くことができる．Fieldtypeボタンをクリックすると全部の属性が表示され，その中から自分の検索を行いたい属性を選択することができるようになっている．検索式にはORを示す'+'および'|'，ANDを示す'*'および'&'，優先順位をコントロールする'('を書くことができる．ここで，Fieldtypeに対する条件入力の場合も，複雑な検索が可能であるように，入力項目を2つ準備し，その間のAND演算，OR演算をサポートした．

Mosaicインタフェースに対してのユーザの入力はsubmitボタンを押すことによって，クライアントプログラムに入力値が渡される．このクライアントプログラムはCGI (Common Gateway Interface) [14]プログラムであり，Mosaicインタフェースに入力された文字列を受け取り，選択された文献の型のFDのセットからVFDを構成する．また，インタフェースから渡された問合せをlex, yaccを用いて解析し，その問合せを選言標準形に変形し，その個々の要素をRPCを用いてサーバに問合せを発行する．選言標準形への変形はサーバにおけるマッチングモジュールの構成を簡単なものとするために行っている．サーバから解が返されると，その解とVFDを用いて問合せを再構成し，もし必要であればもう一度サーバに問合せを発行する[3]．これらのセッションの後，クライアントプログラムはMosaicインタフェースに整形した解を転送し，これはユーザに対して図5.6のように表示される．

通信部分をsocketインタフェースを用いて記述する場合，

1. 問合せおよび解をストリングにエンコード・デコードする
2. 送信側でそのポインタの型と実体を取得して展開してソケットに書き込み，受信側でソケットから読み込んだ型に応じたポインタにメモリを割り当て，さらにポインタの指す実体部分をソケットから読み込んで，割り当てたメモリに書き込む

[3] このサーバは最初に問合せを発行したサーバと異なる場合もある

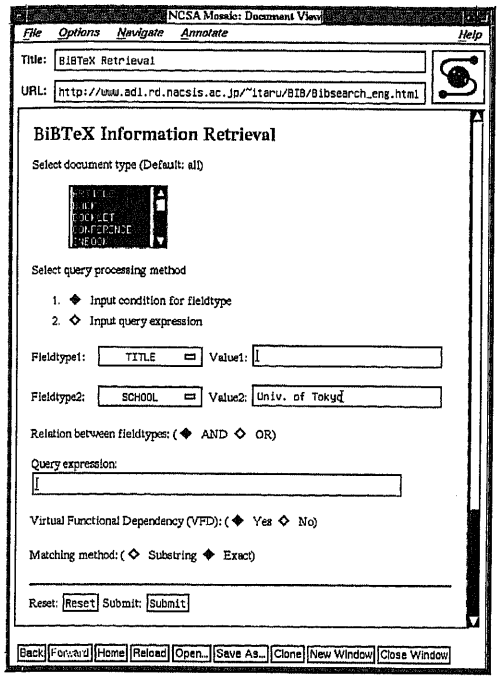


図 5.5: BIBTEX を取り扱うシステムのユーザインタフェース

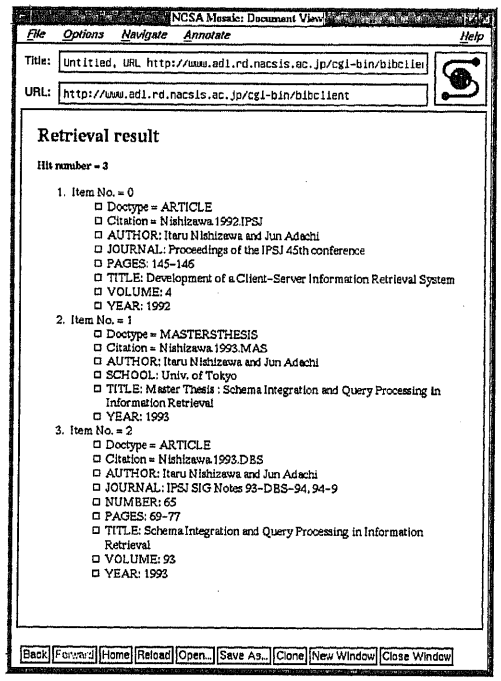


図 5.6: 図 5.5の問合せに対する検索結果

などの処理を行うルーチンを書く必要があるが，ここではRPC を用いることにより，ポインタを自動的に展開することができる．またRPC はXDR (External Data Representation) [87] を使うことによって実現されており，これによりネットワークバイトオーダの問題も解決される．

図 5.7はシステムが通信を行う際に用いるデータ構造を定義している．クライアントは問合せを選言標準形に変換し，条件の連言結合の一つ一つをConjQ 構造体にセットする．DisjQ 構造体はこのConjQ へのポインタを配列として保持し，サーバにはこのDisjQ 構造体が転送される．サーバはDisjQ 構造体を受け取り，条件にあった解を返す．この解はDB 構造体で表現される．信ルーチンはRPC 用記述ファイルからrpcgen(1) コマンドを使って生成され，サーバ側，クライアント側では実際の処理モジュールにこの通信ルーチンを組み込めばプログラムが完成することになる．

5.6 システムの挙動

ここでシステムに簡単な問合せを発行し，動作を確認してみる．発行する問合せは

```
SCHOOL = "Univ.  of Tokyo"
```

である．この問合せの意味は「東京大学出身あるいは在学中の著者の文献情報を求めよ」ということになる．まずこの問合せをVFD を用いない設定でシステムに対して発行すると (クライアント側のインタフェースでVFD method をNo use にすると)，図5.8(a) に示すように**Masterstheis**

```
/*
 * exchange_query.x 問い合わせおよび解の送受信のための仕様。
 * 03/28/1995
 */

/*
 * プログラムの定義
 */
program QUERY_PROCESS_PROG {
    version QUERY_PROCESS_VERS {
        /*
         * 最初のバージョンでは解の個数を返すことにする。
         * そのため、返り値は int であるとする。
         */
        int     PROCESS_QUERY(DISJQ) = 1;       /* 手続き番号 1 */
        DB      GET_ANS_REC(DISJQ) = 2;         /* 手続き番号 2 */
    } = 1;                          /* バージョン番号 */
} = 0x30012438;                     /* プログラム番号 */
```

図 5.7: RPC 記述ファイルのプログラム定義部分

型に属する文献が一つだけ得られた．次にVFDを用いる設定でこの問合せをシステムに対して発行すると(クライアント側のインタフェースでVFD methodをuseにすると)図5.8(b)に示すように先ほど得られた**Mastersthesis**型に属する文献が一つと**Article**型に属する文献が二つ得られた．

これは表5.4に示したFDのセットからVFDを計算した場合にVFD中にAUTHOR → SCHOOLという関数従属性が存在するためで，システムはこの関数従属性と最初に得られた解であるAUTHOR = "Itaru Nishizawa and Jun Adachi"から図5.8(b)のID = 1,3の**Article**型に属する二つの文献を解としてデータベースから取り出してきたためである．この解は論理的には正しく，ユーザにとってはこのような解も得られる方がより便利であると考えられる．

5.7 まとめ

3章で述べた問合せ処理方法の応用として，BIBTEXによって構築された文献データベースを対象としたプロトタイプシステムを設計し，実装した．本システムを用いることにより，ユーザは個人的に作成された小さなデータベースを，あたかも一つの大きな文献データベースであるかのように利用することができる．実際の文献データに対してプロトタイプシステムを動作させた結果，その有効性が確認できた．

Answer ID. Document Category	
Attribute	Value
1. **Mastersthesis**	
AUTHOR	"Itaru Nishizawa and Jun Adachi"
SCHOOL	"Univ. of Tokyo"
TITLE	"Master Thesis: Schema Integration and Query Processing in Information Retrieval"
YEAR	"1993"

(a) Without VFD method

Answer ID. Document Category	
Attribute	Value
1. **Article**	
AUTHOR	"Itaru Nishizawa and Jun Adachi"
JOURNAL	"Proceedings of the IPSJ 45th conference"
PAGES	"145–146"
TITLE	"Development of a Client-Server Information Retrieval System"
VOLUME	"4"
YEAR	"1992"
2. **Mastersthesis**	
AUTHOR	"Itaru Nishizawa and Jun Adachi"
SCHOOL	"Univ. of Tokyo"
TITLE	"Master Thesis: Schema Integration and Query Processing in Information Retrieval"
YEAR	"1993"
3. **Article**	
AUTHOR	"Itaru Nishizawa and Jun Adachi"
JOURNAL	"IPSJ SIG Notes 93–DBS–94, 94–9"
NUMBER	"65"
PAGES	"69–77"
TITLE	"Schema Integration and Query Processing in Information Retrieval"
VOLUME	"93"
YEAR	"1993"

(b) With VFD method

図 5.8: 検索結果

第 6 章

複数の情報資源を取り扱うシステムの提案

本章では分散する複数の情報資源を統合的に取り扱う際の問題点について議論した後，複数の情報資源を統合的に取り扱うシステム DIRECTOR の提案を行う．

6.1 複数の情報資源を取り扱う際の問題点

まず分散する複数の情報資源を統合的に取り扱う際の問題点について議論する．2章の関連研究でも述べたように，複数の情報資源の統合利用を実現する際には解決しなければならない問題を三つあげることができる．以下がその問題である．

1. 情報資源の異種性 (heterogeneity) の吸収方法
2. 情報資源の発見方法
3. 情報資源が構成する情報空間の組織化の方法

以下の節でこれらの問題について議論し，これらの問題を解決し情報資源への統合アクセスを実現するシステムについての検討を行う．まず最初に 6.2節で 1の情報資源の異種性について議論し，次に 6.4節では 2および 3の問題について考察を行う．最後に 6.5節で，これらの問題を解決し，情報資源への統合アクセスを実現するシステムである DIRECTOR のシステム構成およびその構成要素についての検討を行う．

6.2 情報の異種性の問題

2.3節で述べたように，複数の情報資源を取り扱う際にはさまざまなレベルの異種性を取り扱う必要がある．ここでは，文献情報を取り扱う際の例をあげながら，もう一度解決しなければならない異種性を整理する．

1. ある情報源では「筆者」，ある情報源では「著者」というように属性の表現が異なるという属性の表現の異種性の問題
2. 著者名の表現において，ある情報源では Itaru Nishizawa，ある情報源では I. Nishizawa となっているというように記法が異なっているという記法の異種性の問題
3. 同一の文献つまり同じ実体 (entity) を表現するための属性集合が，情報源によって異なっているという問題

2.3節で述べた，意味の異種性 における「意味 (semantics)」という言葉には明確な定義が無く議論の多い所である．実際には実体の同定の問題はデータベースの作成者の世界観までも考慮する必要があり，例えばオブジェクト指向データベースのアプローチにおいては字面の処理では実体の同定は難しいという立場から，ID を用いることによって実体の同定を行っている例もある．しかしながら本論文で取り扱っている複数のデータベースの統合問題においては，各要素データベース間に渡って実体に ID をつけるということは現実的には不可能であるため，本研究においては属性表現の写像，記法の写像により，意味の相違は吸収が可能であるという立場をとる．こ

れは言い替えれば，属性表現および記法の写像で対応できない「意味の異種性」については，本研究では取り扱わないものとするということを意味する．

3章における問合せ処理手法では主に3の「同一の文献つまり同じ実体を表現するための属性集合が，情報源によって異なっているという問題」について議論した．2の記法の問題はスペルミスの問題や，省略の問題，順番の変化の問題等の様々な問題があり，文字型の配列を用いて語を符合化する部分文字列法 [95] や文字のヒット数に着目して語の照合を行う IBIS[91] など多くの研究がなされているが，5章における BIBTEX を取り扱うシステムでは2の記法の問題についてはシステムに完全一致と部分文字列一致を準備するという方法で対応を行った．

次に1の属性の表現の問題について考える．5章で説明したシステムでは属性間の写像があらかじめ与えられている対象を考えていた．ところが，さらに一般的な対象を考える場合には，最終的には人間の判断を仰ぐとしても，これらの属性の間の写像を与える際にシステムの支援が望まれる．また，情報の組織化の問題にも関連するが，ユーザが探したい考えている情報を保持している情報資源を膨大な情報空間から選択する方法についても考える必要がある．そこで，6.3節では上記の2つの問題を解決する目的で，データベースのクラスタリングを行うことに関する検討を行う．

6.3 データベースのクラスタリングについての検討

6.3.1 動機と目的

まず，なぜここでクラスタリングの検討を行うかについて改めて整理しておく．3章で述べた問合せ処理手法を用いて複数の情報資源に対するアクセスを行おうとする際には，最初にデータベースの選択を行う必要がある．対象となるデータベースの集合が決定された後は，3章で述べた問合せ処理法によって自動的に解が求められる．ところが，実際にシステム構成を考える際には対象とするデータベースはどのようなもので，規模としてはどの程度のものを想定しており，これらの対象となるデータベースは誰が作るのかなどの問題を考慮する必要がある．

まず，対象となるデータベースとしては3.1節で述べたように関係データベースを考える．ここで，関係データベースという制限については，問合せ処理において関数従属性という情報を使用するということに起因している．もちろんもっと下のレベルでどのようなデータモデルに基づいたデータベースシステムが使われているかは問題ではなく，システムから見た場合にデータベースのスキーマとして，属性集合，タプル集合，関数従属性の集合で表すことができればよく，これらの情報から仮想データベースが構成される．次に対象となるデータベースであるが，各要素データベースの管理者が作成したものを，属性の写像行った後に利用するということになる．つまりシステムのユーザがこれらの作業を行う必要はない．

次にデータベースの規模の問題について考える．ここでいう規模という言葉には，

1. 各データベースの大きさ

2. データベースの数

という二つの意味がある．3章で述べた問合せ処理手法では，ユーザから問合せで参照された属性と条件が与えられた属性に注目して仮想データベースを構成し，その上での関係である仮想関数従属性 (以下 VFD) を用いるが，この VFD が閉路を構成する場合にはデータベース間にわたって閉包計算に相当する高価な処理を行うことになるため，取り扱うインスタンスの数[1]が問題となる．取り扱うインスタンスの数を問題にするだけであれば，1, 2とも同じ問題としてとらえられるように思えるが，1のデータベースの規模について考えるときはデータベース間で転送する情報が大きくなることに対する，情報の転送の戦略についての検討が必要となる．また2については，例えばインターネットでアクセス可能な全てのデータベースを対象とするというのは実際には不可能であり，どのデータベースを対象とするかを選択するような機構を準備する必要がある．本研究では個人的に作成するような比較的小さなデータベースを主な対象としているため，特に2の問題に焦点をあて，数多くのデータベースから対象とするデータベースを選択するという問題について考察する．この情報資源の選択の問題を考察するにあたり，以下の二点を考慮する．

1. データベースの分類

2. データベースの有用性の指針

まず，2の有用性の問題は検索対象とするデータベースが多過ぎてそれら全てに検索を行うのが現実的に難しい場合に，その中から対象となるデータベースを選択する指針を与えようとするものである．また，1の分類の問題には関連の少ないデータベースを統合利用するよりも関連のあるデータベースを組み合わせて利用するための指針を与えようというものである．例えば3章における問合せ処理手法においては，統合利用を行おうとするデータベース上の属性間に重なりが全くなく，しかも VFD も存在しない場合には通常の問合せ処理方法と同じ結果しか返さない．

さらに，データベースの選択の問題とは別に，6.3.1節で述べた属性の写像の問題がある．この写像の問題はそれぞれの要素データベースの作成者の「世界観」を考慮する必要があるため，大変難しい問題であり，現在のところこれを自動化する研究は成功していない．しかしながら完全な自動化は不可能だとしてもシステムとしては何らかの支援が望まれる．よって，この属性間の写像の付与を支援する手法にについても検討する．

6.3.2 情報の取り出し方についての検討

関係データベースのクラスタリングを行うにあたり，考えうる最も単純な方法は，図 6.1に示すように関係データベースの各属性値をその境界を無視して一つの文献ととらえることにするというものである．このようにすれば，データベースのクラスタリングの問題は2.2節において紹介した文献のクラスタリングの問題に帰着することができ，そこで紹介したさまざまな手法を適用す

[1]ここではタプル数に相当する．

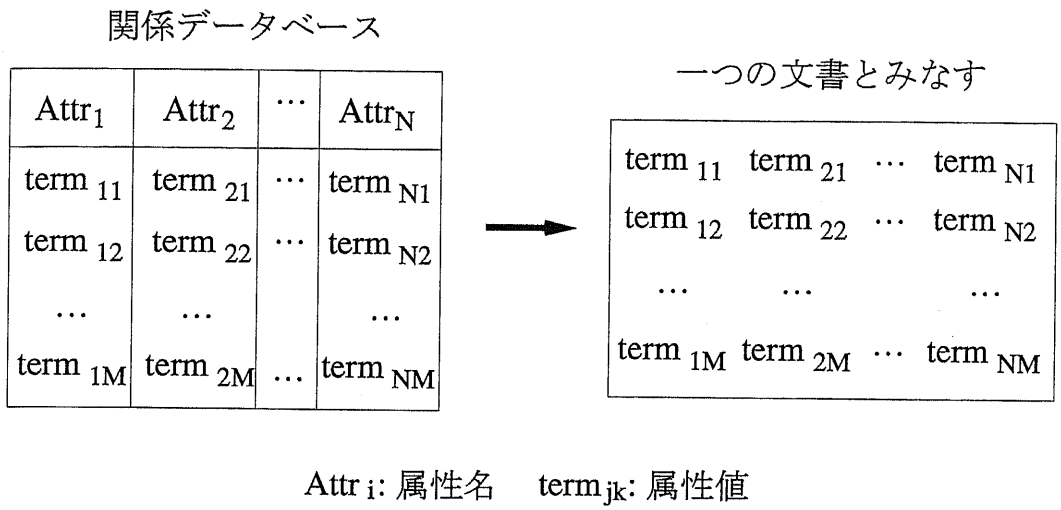


図 6.1: 関係データベース全体を一つの文書ととらえる手法

ることができる．また，有用性の問題についても2.1.2節で説明した全文データベースを対象としたGlOSSの手法を用いることも可能となろう．しかしながら，単純にこのような手法をとった場合には，

- 関係データベースの持っている属性という構造情報を捨ててしまうことになること
- 6.3.1節で述べた属性のマッピングへの支援とは全く関係が無くなってしまうこと

などの問題点をあげることができる．そこでこれらの問題点に対応するため，ここではデータベース全体を一つの文献ととらえるのではなく，ある属性に属する値の集合つまり非形式的にいうと，データベースにその属性によって射影演算を施した値の集合を一つの文献とみなす手法が有力であると考えられる (図6.2参照)．

6.3.3 語の選び方とクラスタリング手法に関する検討

文献のクラスタリングにおいて式 (6.1) で示される Term-Document Matrix Dがさかんに用いられていることを関連研究で述べたが，この行列中に現れる語 T_iの選び方についての議論は少なく，索引付けされた語を用いる，あるいは人間があらかじめ与える等の方法がとられているようである．しかしながら対象となるデータベースが変化する場合にはこれをあらかじめ与えておくことは不可能であるので，ここでは式 (6.1) における T_iの選び方についてなんらかの指針を定める必要がある．

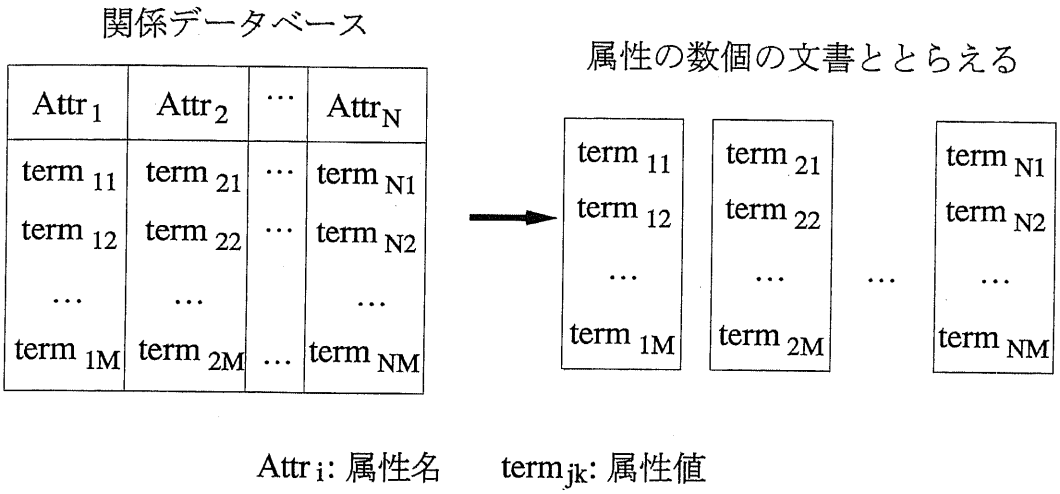


図 6.2: 各属性ごとの値の集合を一つの文書ととらえる手法

$$D = \begin{array}{c} \\ D_1 \\ D_2 \\ \\ D_N \end{array} \begin{array}{c} \begin{array}{ccc} T_1 & T_2 & T_t \end{array} \\ \begin{bmatrix} d_{11} & d_{12} & d_{1t} \\ d_{21} & d_{22} & d_{2t} \\ & & \\ d_{N1} & d_{N2} & d_{Nt} \end{bmatrix} \end{array} \tag{6.1}$$

6.3.2節でも述べたように，ここでは属性を単位としたクラスタリングを行おうとしているので，まず最初に属性の定義域の大きさに注目する．つまり，属性値から切り出された単語を全て式(6.1)におけるTとして取り扱うと，Tの大きさが問題になるため，これに対処するために属性の定義域に注目する．まず各属性ごとにデータベースを(語，頻度)という形に整理し，この属性に現れる属性値の定義域を求め，この定義域の大きさに注目してまずクラスタリングを行い，その後に近いクラスタ同士の属性に関して語の内容を考慮したクラスタリングを行うことにより，Tの大きさの爆発を押えることができる．

6.4 情報資源の組織化と発見方法に関する問題

6.1節の問題点3の「情報空間の組織化の方法」を考えた場合，2.1節で述べたように現在の情報システムはその情報資源へのアクセスの仕方，つまり情報空間の管理の仕方という観点から索引型，航行型に分類することができる．各々の特徴と問題点をまとめると以下のようになる．

索引型のシステム もし，全ての情報が統合管理されているならば，例えば全ての情報にユニークなIDを付加し，それを管理するサーバを作っておけばよい．これは現在稼働中の情報検索システムでとられている主要なアプローチであり，システム構成が簡単になるなどの利点が

あるが，ネットワークの巨大化によって情報量が激増している今日では，検索時のサーバへのアクセスの集中や，管理する情報のメンテナンスなど，情報量の大きさに起因する問題は無視できないものとなっている．

航行型のシステム WWW 等の広域情報システムはハイパーテキストの技法を用いてネットワークを通じて航行型の情報検索を行うシステムであり，情報空間を有向グラフとして取り扱う．これらのシステムではユーザからみると索引は見えず，情報検索というよりも，むしろリンクを辿りながら情報を発見するという情報発見システムと呼ぶ方が適切である．このようなシステムでは索引型のシステムで述べた，検索時のサーバへのアクセスの集中の問題，管理する情報のメンテナンスなどの問題は回避できるが，現在のシステムでは情報発見という色彩が強く，実用的な情報検索を行うのは難しい．

このように索引型・航行型のシステムには各々に長所と短所が存在する．また，6.1節の問題点2の「情報資源の発見方法」にも関連して，1章で述べたように，膨大な情報からいかにして有用な情報を得るかということが大きな関心事となっている．よって，有用な情報の選択と6.2節で述べた属性のマッピングを支援するような仕組みを考える．

6.5 DIRECTOR の提案

6.5.1 求められる機構

5章では BibTeX を対象としたシステムを構築した．その際には，「スキーマの構造および属性の意味 (semantics) に関しては対応表を与え，属性値の一致によってその同一性を規定するものとする.」という仮定をおいていた．しかしながら，5.1節でも述べたように，さらに一般的な対象を考えた場合には対応表の作成方法，管理の仕方などについての議論が不可欠である．そこで，本節では6.2節および6.4節で述べた問題点を解決し，分散する複数の情報源を統合的に利用するためのシステムである DIRECTOR について述べる．DIRECTOR のシステム構成図を図 6.3に示す．DIRECTOR は情報検索システムが持つべき特徴であると考えられる以下のような機能の実現を目標として設計されている．

- 情報資源の位置透明性を確立し，ユーザが情報資源の位置を陽に指定する必要がないこと
- ユーザが各情報資源上での属性の違いを意識せずに，有用な情報を体系的に得られること
- ある特定のシステム構成要素にアクセスが集中することを避けること

以下の節ではこれらの機能を実現するためのシステム構成およびシステムの構成要素について議論していく．

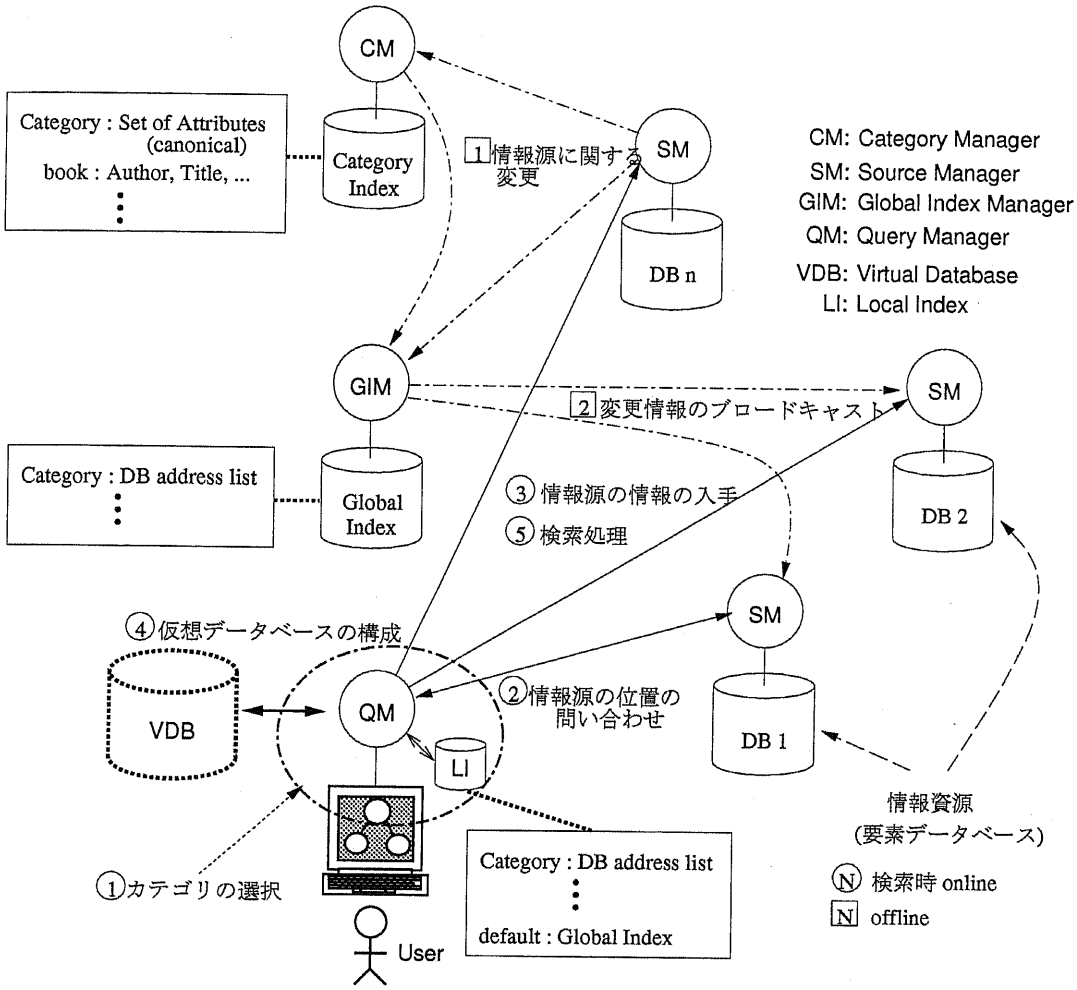


図 6.3: DIRECTOR のシステム構成

6.5.2 DIRECTOR の構成要素

システムの構成要素は Global Index Manager, Category Manager, Source Manager, Query Manager であり，これらは自律的に動作する．以下，これらの構成要素について説明する．

- **Category Manager (CM)**

  DIRECTOR では，情報の区分・管理に際し，実体 (entity) という概念に注目する．この実体によって情報を区分し，この区分をカテゴリ (category) と呼ぶ．CM はカテゴリとそのカテゴリ中の情報の実体の性質を表現する標準属性集合を管理する．例えばカテゴリとして「文献」を考えた場合，この標準属性集合は「著者」,「タイトル」などを考えることができ，その初期値はシステム設計者が与えるものとする．CM はさらに，各属性に対する記法の規則を管理する．CM に変更が加えられた場合には，その変更は GIM に伝えられる．

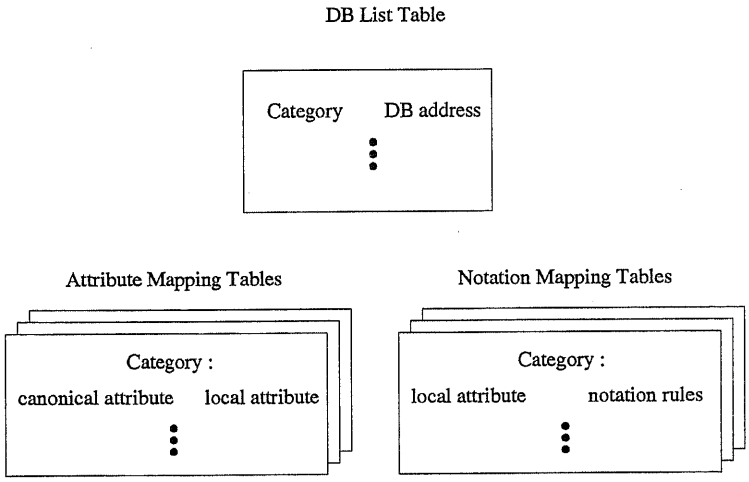


図 6.4: SM が管理する情報

- **Global Index Manager (GIM)**
  GIM は各情報のカテゴリと，その情報を保持する情報源の名前のリストを管理する．ある情報源を新たにシステムに追加・削除あるいは変更するなどの処理は GIM に対して行われる．GIM はその変更を該当するカテゴリの情報源に対してブロードキャストする．この処理の頻度は通常の情報検索処理よりも遥かに小さいと考えられる．

- **Source Manager (SM)**
  SM の一つの役割は検索エンジンであり，Query Manager からの問合せに対して，各情報源の情報に応じた検索を行う．この検索の際には，属性名のマッピングと，属性に応じた記法のマッピングを用いる．新しい情報源を DIRECTOR システムに加える場合，その情報源の管理者は SM にその情報源における属性と CM で管理される属性のマッピングを行う．この操作を行う際には 6.3節で説明した属性を単位としたクラスタリング手法を用いることにより，マッピングの支援を行うことができる．CM に対応する属性が存在しない場合には，これまでに存在する属性とは異なる新しい名前の属性を CM に追加する．記法についての規則についても同様である．SM は図 6.4に示すように，Attribute Mapping Table, Notation Mapping Table, DB List Table を管理する．6.5.2で述べた GIM は各カテゴリの情報資源のリストを保持しているが，

  1. 検索の際の GIM へのアクセスの集中を避ける
  2. GIM がアクセス不能の場合も情報資源へのアクセスを可能とする

  などの理由から，各 SM は同カテゴリの情報資源の名前のリストを保持している．このリストは GIM のコピーであり，GIM に変更があった場合には変更があったカテゴリの情報資源にブロードキャストによって変更が伝えられる．

	索引型	航行型	DIRECTOR
体系的な検索	○	×	○
サーバへのアクセスの集中	×	○	○
管理情報のメンテナンス	×	○	○
システム構成の単純さ	○	○	×

図 6.5: 各情報システムの利害得失の比較

- **Query Manager (QM)**
  QM は Local Index (LI) を管理する．LI はカテゴリとそのカテゴリの情報を持つ情報源のリストを管理しており，GIM のキャッシュとみなすことができる．ユーザが選択したカテゴリに関する情報が LI にある場合は QM はその LI 中のいずれかにアクセスし，現在の当該カテゴリのデータベースリストを得て，LI に記述する．もし LI 中に当該のカテゴリに関する情報がない場合は GIM にアクセスし，LI に得られた情報資源のリストを記述する．

6.5.3 情報検索時のシステムの挙動

DIRECTOR の情報検索時の挙動について説明する (図 6.3参照)．DIRECTOR は情報資源の選択と，その情報資源に対する仮想データベースを利用した検索という 2 つのフェーズからなる．まず，情報資源の選択のフェーズでは，ユーザは必要な情報をカテゴリという形でシステムに入力する．この情報から QM はまず LI の情報を用いて，そのカテゴリの情報を保持する情報源を選択し，その情報源の SM とのインタラクションによって，現在の情報源の集合を得る．

情報源のリストが得られたら，QM はそれらにアクセスし，仮想データベースを構成するために必要な情報を得て仮想データベースを構成し，問合せ処理を行う．ある情報源にカテゴリの属性とマッピングできない属性が存在した場合は，仮想データベース上で空値を補うことによってユーザの検索意図を損なわずに検索を行うことができる．

6.5.4 従来の分散情報システムとの比較検討

従来の索引型システム，および航行型システムと DIRECTOR の比較を図 6.5にまとめる．DIRECTOR は，カテゴリ分けされた情報空間でユーザから情報源の位置透明性を確立し，SM によるマッピング機能などによって，体系的な情報検索が可能となるように設計されている．また，情報を検索するフェーズでは仮想データベースを構成することによって各情報資源に個別に検索を行う場合と比較して解集合を大きくすることができる．索引型のシステムで問題となる検索時のサーバへのアクセスの集中や，管理する情報のメンテナンスなど，情報量の大きさに起因する問題については SM における同カテゴリの情報源のリストの管理，および LI における情報のキャッシュを用いることにより，GIM に対するアクセスの集中を避けることができる．

6.6 まとめ

分散する複数の情報資源を統合的に取り扱う際の問題点は以下のようにまとめることができる.

1. 情報資源の異種性 (heterogeneity) の吸収方法
2. 情報資源の発見方法
3. 情報資源が構成する情報空間の組織化の方法

本章では異種性の中で属性の表現の異種性の問題を解決し，統合利用を行うという観点から重要と思われる情報の分類の問題に取り組むために，データベース単位のクラスタリングという概念を検討し，クラスタリングを行うための手法を検討した．さらに，情報資源の発見・管理の手法についても従来の情報システムの問題点を考慮し，複数の情報資源への統合的なアクセスを実現するシステムである DIRECTOR を提案し，そのシステム構成と構成要素についての検討を行った.

第7章

考察

本研究の成果について考察を行い，今後の課題について述べる．

7.1 本研究の成果

本研究の成果を箇条書きの形でまとめると以下のようになる.

- **統合的アクセスを実現するための問合せ処理手法の提案**
  実体を表現する属性集合が異なるデータベースに対して，ユーザの問合せに対して論理的に正しくかつ完全な解を得るための問合せ処理手法を提案した．具体的には以下のような手順による.
  - データベースの形式化を行い，仮想データベース，仮想関数従属性の定義を行った．この仮想データベースおよび仮想関数従属性は，問合せを行う対象となるデータベースが変化するに伴って動的に再構成される.
  - 仮想データベースの無矛盾性を示した．また，仮想データベースに対して問合せを行った時に正しく，かつ完全な解を得るためのアルゴリズムが存在することを示し，実際にそのアルゴリズムを示した．このアルゴリズムを用いることにより，仮想関数従属性が仮想データベース上で閉路を構成しない場合には正しくかつ完全な解を多項式時間で計算することができる.

- **問合せ処理方式の提案と有効性の検証**
  3節で提案した問合せ処理方式の有効性の検証を行った．データベース統合プロセスにおいては取り扱うデータ量が多くなるという観点からインスタンスベースでルールを与える手法との定性的な比較を行い，さらに個々の要素データベースに対して別々に問合せ処理を行った場合と比較して，どの程度解が大きくなるかを検証するためにシミュレーションを行った．その結果，シミュレーションで設定した条件の下では最高 20%程度解が大きくなることがわかった.

- **BibTeX 形式の文献データベースを対象とするシステムの開発**
  問合せ処理方法の応用対象として BibTeX 形式で構成される文献データベースを考え，システムを構築した．本システムは複数の人が個人的に作成した BibTeX ファイルを，ネットワークを介してあたかも一つの大きな文献データベースのように利用することができる.

- **属性を単位としたクラスタリング手法と，分散する情報を統合利用するシステム DIRECTOR の提案**
  分散する情報資源を取り扱う際の問題点を整理し，異種性の問題の一つである属性の表現の問題を解決するために属性を単位としたクラスタリング手法の検討を行った．これは3節で提案した問合せ処理法が属性間の写像が与えられていることを要求するためである．また，情報空間の組織化を行うために情報を統合利用するシステム DIRECTOR を提案し，そのシステム構成とシステム要素の検討を行った．5 節で紹介した BibTeX 形式の文献データベースを対象とするシステムはこの DIRECTOR のサブセットと位置付けることができる.

以上のように本研究では分散する情報資源の統合利用のための検討がその中心である．分散する情報資源の利用という目的のためにはレベルの異なる複数の異種性の吸収と，情報資源を取り扱う情報空間の組織化が二つの大きな問題となる．本研究ではこの二つの問題をさらにいくつかの部分問題に分割し，それらの問題を考察することにより，分散する情報資源への統合的アクセスのための基礎となる部分の検討を行うことができた．

7.2 他の研究との関連

本節では本研究と関連研究の相違点について論じる．

7.2.1 未知の値を求める方法について

最近はやりの研究であるデータベースからの知識発見 (Knowledge discovery in database)[26, 47] で取り上げられている問題のひとつに missing value の問題がある．データベースからの知識発見とは「潜在的で自明でなく有用な情報をデータから抽出すること」であり，以下のように特徴付けることができる [26].

- 発見された知識は人間にとって理解可能な高レベル言語 (High-Level Language) である必要があること．
- 発見された知識はデータベースの内容を的確に表現するものであること．あるいはその確からしさの指標を示すことができること．
- 発見された知識は新奇な有用なものであること．
- 知識発見の過程は効率的なものであること．

この知識発見の研究においては，

1. データベースから知識を得る
2. 得られた知識をルールとして用いることによって，未知の値を求める

という処理が行われる．2の「ルールを用いてわからない値を求める」という処理は仮想データベース上の関係である仮想関数従属性を用いて仮想データベース上の空値に相当する値を求めていくことと関連すると思われる．確かにルールを用いて未知の値を求めるという点では本研究と missing value の問題は類似点があると考えることもできるが，本手法では対応するデータベースの変化に伴って仮想関数従属性，仮想データベースを動的に計算する．これは「閉包計算を用いた動的なルールの合成」という概念ととらえることができ，この点が得られたルールセットが固定される missing value の研究との相違点と考えられる．

7.2.2 情報資源の組織化，発見方法について

索引型のシステムと航行型のシステムとの比較の概論については6.5.4節で述べたので，ここでは研究段階の各システムとの比較検討を行う．

GlOSS[28] は集中化索引を作らないとしているが，それはあくまでも検索の際に用いる索引の意味であり，どのデータベースにはどのような情報が存在するかについては集中的に人手で管理を行う必要があるようである．また，SIDLP[86] はその目的が，「個人的に管理されているような異種 (heterogeneous) 情報を統合しようとする試み」ということで，本研究と目標は似た所を狙っているが，SIDLP がアクセスの統合を目指しているのに対して，本研究における手法は情報自体の統合を行おうとしている点が異なっている．MINERVA[90], Indie[18] は共に，自律的に動作するシステム要素を備えて情報資源への統合的なアクセスを実現するが，それらの構成要素から得られた解の整合性に関する議論が行われていない．

7.2.3 異種性の取扱いについて

2.3節で述べたように，連邦型データベースシステムが要素データベースの統合に重点をおき，各要素データベースのスキーマを統合した大域的な連邦スキーマを作成して，ユーザからの問合せを処理するのに対して，マルチデータベースシステムでは大域的なスキーマを作成せず，ユーザからはデータベースは一つには見えない．マルチデータベースシステムは，ユーザが全域的なスキーマを用いないという所に最大の特徴がある．この結果，複数のデータベースには意味の違い，矛盾が存在することになるが，これらは自律性の一つの側面ととらえられる．本研究においては対象とするデータベースが変化するという立場から大域的なスキーマを構成せず，しかもユーザからの問合せに対して矛盾のない解を返すことのできるような仕組みを目指しており，この点が連邦型データベースシステムおよびマルチデータベースシステムとは異なる．

[40] は自律して動作する異種データベースを統合する際のタプルが同じ実体 (entity) を表現しているかどうかを決定するためにインスタンスレベルのルールである ILFD (instance level functional dependency) を与えているが，統合プロセスでは ILFD の数は膨大になる可能性がある．これに対して本論文の手法ではタプルの同一性問題については各要素データベース上のスキーマレベルでの一貫性を考えるというより実用的なアプローチをとっている．IRDS [67], PCTE[2] はともに分散する情報の利用のための規格であるが，両方とも CASE を対象とし，プログラム (システム) 構築を補助するという目的で作成され，プログラムのマッピングをどのようにして与えるかということについて議論しており，取り扱う対象が異なっている．また，[20, 21, 79, 81] は全て 6.2節で述べた異種性のある一つのレベルを取り扱っているだけで，3 つの異種性に関して総合的に議論されてはいない．

7.3 未解決の問題への検討と今後の展望

本節では未解決の問題についてまとめ，今後の展望について述べる．

- **属性値のマッチングの問題についての検討**

  検索に対する各情報源におけるマッチングに対する検討が必要である．これは6.2節で述べた記法の異種性に関連する問題であり，6.2節で述べたようにさまざまなアルゴリズムが存在する．近年，これまでに蓄積された文書をOCR等を利用して文字情報に変換し，計算機による操作が容易な形式に変換する文書画像データベースの研究が盛んになりつつあり[46, 58]，この関連で盛んになりつつある不完全なキーワードマッチング等の研究成果が期待される．実際にこの記法の異種性を吸収するモジュールはDIRECTORの構成要素であるSource Managerに組み込まれることになるが，対象に応じた記法に関する規則を個別に作成するというのではなく，SUN MicrosystemsのHotJava[89]で用いられているような，動的な処理プログラムの転送機構を用いることによる，ルールの違いによって処理を動的に変更するシステムの実装について検討している．

- **従属性の拡張，確実度の導入についての検討**

  3.2.2節で述べたが，本論文で提案した問合せ処理手法は全ての対象に適用できるわけではない．つまり，問合せ処理の正しさを保証する関係理論Tと関数従属性公理FDの組合せである$T \cup FD$がいつも無矛盾であるとは限らないからである．このような問題は例えば次のような二重国籍の問題を取り扱おうとした際に問題となる．今簡単のために，統合の対象となるデータベースは二つだけであるとし，これらのデータベースは著者と国籍に関する情報を蓄積しているものとする．一方のデータベースにおいては，著者「西澤 格」の国籍は「日本」であり，もう一方のデータベースにおいては，著者「西澤 格」の国籍は「アメリカ」であったとし，さらに個々のデータベース上で著者から国籍に関数従属性があったとする．この場合，提案した統合手法では矛盾が起きるのというものである．このような問題は，論文中でも述べているように，統合を行おうとするデータベース間でデータが矛盾している場合であり，3で説明した問合せ処理法では対象外の問題としていた．

  もしこの問題に対応しようとすると，従属性を拡張する方法，あるいは確実度(certainty)を導入する方法などを考えることができる．従属性の拡張については多値従属性[23]，包含従属性[12, 13]等を候補としてあげることができ，これらについて検討を行ったが，これたの従属性は関数従属性ほど一般的に用いられていないことと，実際に関数従属性をこれらの従属性に変更すると仮想データベース上で成り立つ従属性の定義と公理の構成がうまくいかないこと[1]などから従属性の拡張については単純に拡張することはできないという結論に達した．

[1]例えば多値従属性を考えた場合，関数従属性では成り立つ推移律がそのままでは成り立たない．具体的には，属性集合X, Y, Zに対して，関数従属性で成り立つ推移律$X \rightarrow Y$かつ$Y \rightarrow Z$ならば$X \rightarrow Z$が多値従属性においては成立しないということである．

次に確実度を入れる方法について考えると，これは2.2.4節で紹介した，ルール(ここでは関数従属性に相当する)に確率を入れて拡張する方法と同じ手法と考えることができる．この方法は先ほどの問題についての有効なアプローチと考えることができるが，確実度に関する情報を取り出すための統計的な手法の検討と，確実度を入れたルールの集合から構成される公理系についての意味づけに関する検討が必要となる．

- **クラスタリング手法の検討および検証実験**

  6.3.1節でデータベースのクラスタリングを行うことについての検討を行った．6.3.1節では属性を単位としたクラスタリングの手法の提案を行っただけであるので，実際のデータを用いて実験を行い，このクラスタリング手法の有効性を検証する必要がある．また，この属性のクラスタリングを行った結果を用いて，実際のデータベースのクラスタリングを行うわけであるが，属性の間に成り立つ関係とその集合として構成されるデータベースの間に成り立つ関係の相関についての考察を行う必要がある．

- **データベースの有用性についての検討**

  6.3.1節で述べたデータベースの有用性について理論的な検討が必要である．6.3.1節ではGlOSSの手法を用いることが可能であることを述べたが，GlOSSは各データベースで，ある語が現れる期待値を計算しているとみなすことができ，これは各データベースごとの有用性を示す尺度であるといえる．これに対して，本研究では情報の統合利用を目標としており，複数の情報資源を利用することによってより多くの情報を得ることを目指している．このため，各データベース個々の有用性のみならず，これらの情報を組合せて用いた際の有用性の尺度が定義できれば興味深い．2.2.4節では関係の大きさに注目する研究を紹介した．これらの関係式を用いて複数のデータベースを組み合わせて利用する場合の関係の大きさの期待値を見積もることにより，データベース選択の基準とする方法を検討している．

第 8 章

結論

本研究では，分散する情報資源を統合的に取り扱うための手法の確立を目標としている．分散する情報資源を取り扱う際には，情報の異種性の問題と，情報資源の管理方法の問題，そして氾濫する膨大な情報の中から有用な情報をいかにして見つけ出すかという問題を解決する必要がある．

まず異種性の問題については異種性のレベルに応じた手法を適用する必要があるが，実体を表現する属性集合が異なる場合に対応する問合せ処理では仮想データベースという概念を導入することによって，ユーザは複数の情報資源を意識すること無く，しかもそれぞれの情報資源に対して個別に検索を行う場合よりも多くの情報を得ることが可能となった．本論文ではこの仮想データベースを利用する問い合わせ処理方法については理論的な検討と，有効性を評価するためのシミュレーションを行い，その有効性を確認した．また，実際に通信機構，仮想データベースを用いた問い合わせ処理機構を実装したプロトタイプシステムを作成し，その評価を行った．

さらに一般的な対象について考えた場合には，前述の情報資源の管理方法と有用な情報の発見方法についての考察が必要となる．さらに情報の統合利用のために不可欠な情報資源間の情報の写像の支援を行うべく，属性を単位とするクラスタリング手法について検討し，実際にこれらを構成要素として分散する情報を統合利用するシステム DIRECTOR の提案と，そのシステム構成およびシステム構成要素の検討を行った．

計算機とネットワークの普及によって，利用できる情報はますます増えつつあり，さまざまな情報を個人が作り，発信するということが盛んに行われるようになりつつある．ネットワークを介してこれらのデータベースに対する統合的なアクセスを実現することは，疎結合の情報資源の活用に対する一つの可能性を探るものである．本研究が分散資源のより有効な活用のための一つの手助けとなることを期待して本論文の結びとする．

謝辞

本研究を進めるにあたり，貴重な時間を割いてご指導下さった安達淳教授に深く感謝致します．先生は現在電子図書館関係のお仕事で非常にご多忙でありながら，思案にくれる私に熱心に研究の進むべき道を示唆下さり，激励して下さいました．また，特に理論面において本研究に関して事あるごとに有益な助言・示唆を賜わりました，学術情報センターの高須淳宏助教授にも深く感謝致します．高須先生の御提案で学術情報センター内の有志で行った論理プログラムに関する本の輪講は，博士論文をまとめる上で大きな助けとなりました．

安達研究室の先輩であり，1995年現在学術情報センター助手の片山紀生先生には理論はもとより，実際のシステム作成やプログラミングへの多くの助言を賜わりました．片山先生にはお忙しい中でも私の質問等に丁寧な解説をいただきました．大変感謝しております．修士課程に進むまでワークステーションに触れたことさえなかった私が計算機の分野で論文をまとめるなどの仕事がこなせるようになりましたのも安達先生，そして安達研究室の先輩であるNECの倉島顕尚さん，学術情報センターの片山紀生先生を始めとする研究室の人たちの御指導の賜であります．どうもありがとうございました．

5年間を共に過ごした，学術情報センター内の安達研究室，浅野研究室，濱田研究室のみなさんには，公私両面にわたってお世話になりました．みなさんへの感謝の気持ちをここに記したいと思います．学術情報センター内の研究室は，計算機環境をはじめとして研究設備に大変恵まれており，ここで学ぶことができたのは大変幸せであったと感じています．また，博士課程の第2年度からは日本学術振興会新プログラムの特別研究員に選ばれることができ，金銭面でも研究の大きな助けとなりました．このようなプログラムを御提供下さいました日本学術振興会に感謝致します．

最後に大学院博士課程までの長きにわたって，何不自由無く学問を修める機会と援助を与えてくれた両親を始めとする家族と，特に精神的な面でいつも支えになってくれた妻に感謝の意を表したいと思います．

1995年12月20日

西澤 格

参考文献

[1] W. W. Armstrong: "Dependency Structures of Data Base Relationships." *Information Processing*, pp. 580–583 (1974) North-Holland Pub. Co., Amsterdam.

[2] European Computer Manufacturers Association: "Portable Common Tool Environment (Standard ECMA-149) Abstract Specification." (1993) URL: ftp://omg.org/pub/PCTE/ から入手可能.

[3] F. Bancilhon and R. Ramakrishnan: "An Amateur's Introduction to Recursive Query Processing Strategies." In *Proceedings of ACM-SIGMOD Conference on Management of Data.*, pp. 16–52 (1986).

[4] H. W. Beck, T. Anwar, and S. B. Navathe: "A Conceptual Clustering Algorithm for Database Schema Design." *IEEE Transactions on Knowledge and Data Engineering*, Vol. 6, No. 3, pp. 396–411 (June 1994) ISSN:1041-4347.

[5] T. Berners-Lee, et al.: "World-Wide Web: The Information Universe." *Electronic Networking: Research, Applications, and Policy*, Vol. 2, No. 1, pp. 52–58 (September 1992).

[6] T. Berners-Lee, L. Masinter, and M. McCahill: "Uniform Resource Locators (URL)." RFC 1738 (December 1994).

[7] P. A. Bernstein: "Synthesizing Third Normal Form Relations from Functional Dependencies." *ACM Transactions on Database Systems*, Vol. 1, No. 4, pp. 277–298 (1976).

[8] A. D. Birrell and B. J. Nelson: "Implementing Remote Procedure Calls." *ACM Transactions on Computer Systems*, Vol. 2, No. 1, pp. 39–59 (1984).

[9] F. Can: "Incremental Clustering for Dynamic Information Processing." *ACM Transactions on Information Systems*, Vol. 11, No. 2, pp. 143–164 (April 1993) ISSN:1046-8188.

[10] F. Can, E. A. Fox, C. D. Snavely, and R. K. Robert: "Incremental Clustering for Very Large Document Databases: Initial MARIAN Experience." *Information Sciences*, Vol. 84, No. 1-2, pp. 101–114 (May 1995) ISSN:0020-0255.

[11] F. Can and E. A. Ozkarahan: "Concepts and Effectiveness of the Cover-Coefficient-Based Clustering Methodology for Text Databases." *ACM Transactions on Database Systems*, Vol. 15, No. 4, pp. 483–517 (December 1990) ISSN:0362-5915.

[12] M. A. Casanova, R. Fagin, and C. H. Papadimitriou: "INCLUSION DEPENDENCIES AND THEIR INTERACTION WITH FUNCTIONAL DEPENDENCIES." *Journal of Computer and System Sciences*, Vol. 28, No. 1, pp. 29–59 (February 1984) ISSN:0022-0000.

[13] M. A. Casanova and V. M. P. Vidal: "TOWARDS A SOUND VIEW INTEGRATION METHODOLOGY." In *Proceedings of the Second ACM SIGACT-SIGMOD Symposium on Principles of Database Systems.*, pp. 36–47 (1983) ISBN:0-89791-097-4.

[14] "The Common Gateway Interface (CGI)." URL: http://hoohoo.ncsa.uiuc.edu/cgi/overview.html (1995).

[15] K. C. C. Chan and A. K. C. Wang: "A Statistical Technique for Extracting Classificatory Knowledge from Databases.", chapter 6, pp. 107–123 AAAI Press / The MIT Press (1991).

[16] E. F. Codd: "A Relational Model of Data for Large Shared Data Banks." *Communications of the ACM*, Vol. 26, No. 1 (1983).

[17] "Computer Science Technical Reports." URL: http://www.cnri.reston.va.us/home/cstr.html (1995).

[18] P. B. Danzig, S. Li, and K. Obraczka: "Distributed Indexing of Autonomous Internet Services." *Computing Systems*, Vol. 5, No. 4, pp. 1–12 (1992).

[19] U. Dayal and H. Hwang: "View Definition and generalization for database integration in a multidatabase system." *IEEE Transactions on Software Engineering*, Vol. 10, No. 6, pp. 628–644 (November 1984).

[20] L. G. Demichiel: "Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains." *IEEE Transactions on Knowledge and Data Engineering*, Vol. 1, No. 4, pp. 485–493 (December 1989).

[21] G. Dong and K. Ramamohanarao: "Representation and Translation of Queries in Heterogeneous Databases with Schematic Discrepancies." In *IFIP Transactions A-25, Interoperable Database Systems.*, pp. 177–189. North-Holland (1993).

[22] A. Emtage and P. Deutsch: "Archie – An Electronic Directory Service for the Internet." In *Usenix Winter Conference.*, pp. 93–110 (1992).

[23] R. Fagin: "Multivalued Dependencies and a New Normal Form for Relational Databases." *ACM Transactions on Database Systems*, Vol. 2, No. 3, pp. 262–278 (September 1977).

[24] R. Fagin: "Horn Clauses and Database Dependencies." *Journal of the ACM*, Vol. 29, No. 4, pp. 952–985 (October 1982).

[25] J. Fedorowicz: "THEORETICAL FOUNDATION OF ZIPF'S LAW AND ITS APPLICATION TO THE BIBLIOGRAPHIC DATABASE ENVIRONMENT." *Journal of the American Society for Information Science*, Vol. 33, No. 5, pp. 285–293 (September 1982) ISSN:0002-8231.

[26] W. J. Frawley: "Knowledge Discovery in Databases: An Overview.", chapter 1, pp. 1–26 AAAI Press / The MIT Press (1991).

[27] Hervé Gallaire, Jack Minker, and Jean-Marie Nicolas: "Logic and Databases: A Deductive Approach." *ACM Computing Surveys*, Vol. 16, No. 2, pp. 153–185 (1984).

[28] Luis Gravano, Hector Garcia-Molina, and Anthony Tomasic: "The Effectiveness of GlOSS for the Text-Database Discovery Problem." In *Proceedings of the SIGMOD'94.* (1994) http://gloss.stanford.edu/papers.html から入手可能.

[29] D. Heimbigner and D. Meleod: "A federated architecture for information management." *ACM Transactions on Office Information Systems*, Vol. 3, No. 3, pp. 253–278 (July 1985).

[30] 堀内 一: "ソフトウェア環境におけるリポジトリへの要求." 情報処理学会研究報告 94–SE–101, Vol. 94, No. 99, pp. 65–72 (November 1994).

[31] S. C. Johnson: "YACC – yet another compiler compiler." Technical Report 32, Bell Laboratories (1975).

[32] B. Kahle and A. Medlar: "An Information System for Corporate Users: Wide Area Information Servers." *Connexions – The Interoperability Report*, Vol. 5, No. 11, pp. 2–9 (November 1991).

[33] 加藤 哲朗: "統合型データベース UniSQL の概要." 情報処理学会研究報告 92–IM–7, Vol. 92, pp. 17–23 (July 1992).

[34] D. A. Keim, H. Kriegel, and A. Miethsam: "Integration of Relational Databases in a Multidatabase System based on Schema Enrichment." In *Proceedings of IEEE RIDE–IMS '93.*, pp. 96–104 (1993).

[35] Taha Khedro and Michael R. Genesereth: "Progressive Negotiation for Resolving Conflicts among Distributed Heterogeneous Cooperating Agents." In *Proceeding of AAAI-94.*, pp. 381–386 (1994).

[36] W. Kim, S. Gala, W. Kelly, and T. Reyes: "An Object-Oriented Approach to Defining a Multidatabase Schema." In *Proceedings of 2nd International Computer Science Conference.* (December 1992).

[37] 桑澤 嘉宏, 児西 清義: "ビジネス・アプリケーションを想定した異種分散マルチメディア DB の構築." 情報処理学会研究報告 93–DBS–96, Vol. 93, pp. 17–25 (October 1993).

[38] L. Lamport: "LaTeX: A Document Preparation System." Addison-Wesley (1986) (邦訳: 『文書処理システム LaTeX』 Edgar Cooke, 倉沢良一 監訳, 大野俊治, 小暮博道, 藤浦はる美 訳, アスキー, 1990 年).

[39] M. E. Lesk: "Lex – a lexical analyzer generator." Technical Report 39, Bell Laboratories (1975).

[40] E. Lim, J. Srivastava, S. Prebhakar, and J. Richardson: "Entity Identification in Database Integration." *IEEE Proceedings of the Ninth International Conference*, pp. 294–301 (1993).

[41] P. Lindner: "Internet Gopher User's Guide." University of Minnesota (1993) gopher://bash.cc.keio.ac.jp/11/info_gopher から入手 可能.

[42] W. Litwin: "An overview of the multidatabase system MRDSM." In *Proceedings of the Annual Conference of the Association for Computing Machinery.*, pp. 524–533 (1985) ISBN:0-89791-170-9.

[43] W. Litwin, L. Mark, and N. Roussopoulos: "Interoperability of Multiple Autonomous Databases." *ACM Computing Surveys*, Vol. 22, No. 3, pp. 267–293 (1990) ISSN:0010-4892.

[44] W. Litwin and A. Zeroual: "Advances in multidatabase systems." In *Research into Networks and Distributed Applications.*, pp. 1137–1151. North-Holland (1988).

[45] J. W. Lloyd: "Foundations of Logic Programming – Second, Extended Edition." Springer–Verlag, New York (1987).

[46] K. Marukawa, H. Fujisawa, and Y.Shima: "Evaluation of Information Retrieval Method based on 'non-deterministic text' of Character Recognition." In *Proceedings of RIAO 88.*, pp. 65–72 (1995).

[47] C. J. Matheus, P. K. Chan, and G. P. Shapiro: "Systems for Knowledge Discovery in Databases." *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, pp. 903–913 (December 1993).

[48] 松井 正一: "BIBTEX の使い方." jbibtex-0.31 のパッケージに添付のドキュメント (January 1991).

[49] 松井 正一: "日本語 BIBTEX:JBIBTEX." JBIBTEX C Version キットと共に配布されている文書 (ver.0.20 用, 1989 年) (1991).

[50] O. A. McBryan: "GENVL and WWWW: Tools for Taming the Web." In O. Nierstrasz, editor, *Proceedings of the First International World Wide Web Conference.* (May 1994) URL: http://www.cs.colorado.edu/home/ mcbryan/WWWW.html.

[51] M. McCahill: "The Internet Gopher: A Distributed Server Information System." *Connexions – The Interoperability Report*, Vol. 6, No. 7, pp. 10–14 (July 1992).

[52] M. Merzbacker and W. W. Chu: "Pattern-Based Clustering for Database Attribute Values." In *Knowledge Discovery in Databases Workshop Notes.*, pp. 291–298 (July 1993).

[53] D. Michie, D. J. Spiegelhalter, and C. C. Taylor: "Machine Learning, Newral and Statistical Classification." Ellis Horwood (1994) ISBN 0-13-106360-X.

[54] P. Mockapetris: "DOMAIN NAMES – CONCEPTS AND FACILITIES." RFC 1034 (November 1987).

[55] P. Mockapetris: "DOMAIN NAMES – IMPLEMENTATION AND SPECIFICATION." RFC 1035 (November 1987).

[56] "NCSA Mosaic Home Page." URL: http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Docs/help-about.html (1995).

[57] M. N. Murty and A. K. Jain: "KNOWLEDGE-BASED CLUSTERING SCHEME FOR COLLECTION MANAGEMENT AND RETRIEVAL OF LIBRARY BOOKS." *Pattern Recognition*, Vol. 28, No. 7, pp. 949–963 (July 1995) ISSN:0031-3203.

[58] A. Myka and Ulrich Günter: "Fuzzy Full-Text Searches in OCR Databases.", chapter 7 in *Advances in Digital Libraries*, pp. 87–100 SPRINGER-VERLAG (1995).

[59] 長尾 真, 石田 晴久, 稲垣 康善, 田中 英彦, 辻井 潤一, 所 真理雄, 中田 育男, 米澤明憲: "情報科学辞典." 岩波書店 (1990) ISBN 4-00-080074-4.

[60] B. C. Neuman: "Prospero: A Tool for Organizing Internet Resources." *Electronic Networking: Research, Applications, and Policy*, Vol. 2, No. 1, pp. 30–37 (September 1992).

[61] J. M. Nicolas: "First order logic formalization for functional, multivalued and mutual dependencies." In *Proceedings of ACM-SIGMOD Conference.*, pp. 40–46, Austin (1978).

[62] 西澤 格, 安達 淳: "情報検索におけるスキーマの統合と質問処理." 修士論文, 東京大学大学院工学系研究科電気工学専攻 (1993).

[63] 西澤 格, 高須 淳宏, 安達 淳: "情報検索におけるスキーマの統合と質問処理." 情報処理学会研究報告 93–DBS–94, 94–9, Vol. 93, No. 65, pp. 69–77 (July 1993).

[64] H. Oliver: "A Comparison of ISO-IRDS and PCTE." pp. 1–9 (August 1992) URL: ftp://omg.org/pub/PCTE/ から入手可能.

[65] National Information Standards Organization: "Information Retrieval Service Definition and Protocol Specifications for Library Applications." *American National Standard Z39.50–1988* (1988) URL: http://ds.internic.net/z3950/z3950.html に関連文書あり.

[66] M. Ozsu and P. Valduriez: "Principles of Distributed Database Systems." Prentice-Hall (1990).

[67] B. Parker: "Introducing ANSI-X3.138-1988: A standard for information resource dictionary system (IRDS)." In *Proceedings of the 1992 Symposium on Assessment of Quality Software Development Tools.*, pp. 90–99. ANSI Information Resource and Dictionary Technical Committee, IEEE (1992) ISBN:0-8186-2620-8.

[68] R. Reiter: "Deductive Question-Answering on Relational Data Bases." In H. Gallaire and J.Minker, editors, *Logic and Data Bases.*, pp. 149–177. Plenum Press, New York (1978).

[69] R. Reiter: "Equality and domain closure in first-order databases." *Journal of the ACM*, Vol. 27, No. 2, pp. 235–249 (1980).

[70] R. Reiter: "Towards a Logical Reconstruction of Relational Database Theory." In *On Conceptual Modelling: Perspectives from Artificial Intelligence, Databases and Programming Languages.*, pp. 191–233. Springer-Verlag, New York (1984).

[71] R. Reiter: "A Sound and Sometimes Complete Query Evaluation Algorithm for Relational Databases with Null Values." *Journal of the ACM*, Vol. 33, No. 2, pp. 349–370 (April 1986).

[72] Riecken, et al.: "Intelligent agents." *Communications of the ACM*, Vol. 37, No. 7, pp. 18–149 (July 1994).

[73] "List of Robots." URL: http://web.nexor.co.uk/mak/doc/robots/active.html (1994).

[74] M. Rusinkiewicz, R. Elmasri, B. Georakopoulous, D. Karabatis, G. Jamoussi, A. Loa, and Y. Li: "OMNIBASE: Design and implementation of a multidatabase system." In *Proceedings of the 1st Annual Symposium in Parallel and Distributed Processing.*, pp. 162–169 (May 1989).

[75] T. Saito: "A clustering method using the strength of citation." *Journal of Information Science*, Vol. 16, No. 3, pp. 175–181 (1990) ISSN:0165-5515.

[76] G. Salton: "Automatic Text Processing." Addison Wesley (1988).

[77] 佐藤 友康, 森保 健治: "管理主体の特性に応じた粒度で格納するリポジトリの構成法について." 情報処理学会研究報告 95–SE–103, Vol. 95, No. 25, pp. 201–208 (March 1995).

[78] "A Gentle Introduction to SGML." URL: http://etext.virginia.edu/bin/tei-tocs?div=DIV1&id=SG (1995).

[79] M. C. Shan: "Pegasus Architecture and Design Principles." In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data.*, Vol. 22, pp. 422–425 (1993).

[80] A. P. Sheth and J. A. Larson: "Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases." *ACM Computing Surveys*, Vol. 22, No. 3, pp. 183–236 (1990) ISSN:0010-4892.

[81] A. Sheth and G.Karabatis: "Multidatabase Interdependencies in Industry." In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data.*, Vol. 22, pp. 483–486 (1993).

[82] "Proceedings of IEEE 3rd International Conference Heterogeneous distributed database systems: Issues in integration.". IEEE Computer Society Press (1987).

[83] P. Smyth and R. M. Goodman: "Rule Induction Using Information Theory.", chapter 9, pp. 159–176 AAAI Press / The MIT Press (1991).

[84] P. H. A. Sneath, R. R. Sokal 著, 西田 英郎, 佐藤 嗣二 訳: "数理分類学 (NUMERICAL TAXONOMY)." 内田老鶴圃, 第 1 版 (February 1994) ISBN 4-7536-0117-X C3041.

[85] V. Sridhar and M. N. Murty: "CLUSTERING ALGORITHMS FOR LIBRARY COMPARISON." *Pattern Recognition*, Vol. 24, No. 9, pp. 815–823 (1991) ISSN:0031-3203.

[86] "Stanford Digital Libraries Project." URL: http://www-diglib.stanford.edu/diglib/pub/ (1995).

[87] Sun Microsystems: "XDR: External Data Representation." RFC 1014 (1987).

[88] Sun Microsystems: "RPC: Remote Procedure Call, Protocol Specification Version 2." RFC 1057 (1988).

[89] Sun Microsystems "The HotJava™ Browser: A White Paper." (1995) HotJava パッケージ中の文書.

[90] R. M. Tong and D. H. Holtzman: "Knowledge-Based Access to Heterogeneous Information Sources." *Proceedings of Digital Libraries*, pp. 61–66 (1994).

[91] 坪谷 寿一, 安達 淳: "知的情報検索のための文献同定システムの実現." 修士論文, 東京大学大学院工学系研究科電気工学専攻 (1992).

[92] J. D. Ullman: "Principles of Database and Knowledge-Base Systems Volume I and II." Computer Science Press (1988).

[93] 宇野 裕之, 茨木 俊秀: "演えきデータベースにおける質問処理コストの近似的評価法." 電子情報通信学会論文誌 D-I, Vol. J75-D-I, No. 9, pp. 855–863 (September 1992).

[94] "Gopher server at veronica.cc.keio.ac.jp." URL: gopher://veronica.cc.keio.ac.jp:2347/7-m200?veronica (1995).

[95] R. A. Wagner and M. J. Fischer: "The String-to-String Correction Problem." *Journal of the ACM*, Vol. 21, No. 1, pp. 168–178 (January 1974).

[96] N. Zhong and S. Ohsuga: "Discovering concept clusters by decomposing databases." *Data & Knowledge Engineering*, Vol. 12, No. 2, pp. 223–244 (March 1994) ISSN:0169-023X.

発表文献

1. 学会誌論文

   - 西澤 格, 高須 淳宏, 安達 淳: “属性集合が異なる複数のデータベースへの統合的問合せ処理手法.” 電子情報通信学会論文誌, Vol. J79-DI, No. 1, pp. 18–27 (January 1996).

2. 国際会議論文

   - I. Nishizawa, A. Takasu, and J. Adachi: “A Query Processing Method for Integrated Access to Multiple Databases.” *The 8th in the series of CAiSE Conferences which provide a forum for presentation and exchange or research results and practical experiences within the field of Information Systems Engineering (CAiSE '96) (submitted).*
   - I. Nishizawa, A. Takasu, and J. Adachi: “An Integrated Query Processing and its Validity in Multidatabase Environment.” *First IFCIS International Conference on Cooperative Information Systems (CoopIS '96) (submitted).*

3. 研究会論文

   - 西澤 格, 高須 淳宏, 安達 淳: “情報検索におけるスキーマの統合と質問処理.” 情報処理学会研究報告 93–DBS–94, 94–9, Vol. 93, No. 65, pp. 69–77 (July 1993).
   - 西澤 格, 高須 淳宏, 安達 淳: “文献情報を対象としたスキーマ統合の一手法とシステムの構築.” 情報処理学会研究報告 95–FI–39, Vol. 95, No. 87, pp. 73–80 (September 1995).

4. 大会論文

   - 西澤 格, 安達 淳: “クライアント-サーバモデルによる情報検索システムの提案.” 第 45 回情報処理学会全国大会講演論文集 (4), pp. 145–146 (October 1992).
   - 西澤 格, 高須 淳宏, 安達 淳: “異種スキーマをもつデータベースへの統合的なアクセス手法.” 第 46 回情報処理学会全国大会講演論文集 (4), pp. 91–92 (March 1993).
   - 西澤 格, 高須 淳宏, 安達 淳: “マルチデータベース環境における問い合わせ処理.” 第 49 回情報処理学会全国大会講演論文集 (4), pp. 235–236 (September 1994).

- 西澤 格, 高須 淳宏, 安達 淳: “分散環境におけるデータベースへの問い合わせ処理.” 電子情報通信学会 1994 年春季大会講演論文集 (6) 情報システム D-108, pp. 6–108 (March 1994).

付録 A

5章において仮定した関数従属性の集合

5章において構築した BibTeX を統合利用するためのシステムでは BibTeX の各型の文献上における属性間の関数従属性を仮定している．5章においてはスペースの関係で **Article** 型と **Masterthesis** 型の文献のみの関数従属性の集合を示したが，ここでは BibTeX で用いる全ての型の文献の関数従属性を示しておく．

Document category	FDs
Article	author → yomi
	author, title → journal
	author, title → year
	author, title → month
	author, title → note
	author, title → number
	author, title → pages
	author, title → volume
	journal, year → volume
	journal, volume, pages → author
	journal, volume, pages → title
Book	author → yomi
	author, title → publisher
	author, title → year
	author, title → month
	author, title → number

Document category	FDs
	author, title → series
	author, title → volume
	publisher → address
Booklet	author → yomi
	author, title → year
	author, title → month
Conference	author → yomi
	author, title → booktitle
	author, title → edition
	author, title → month
	author, title → number
	author, title → organization
	author, title → pages
	author, title → publisher
	author, title → series
	author, title → volume
	publisher → address
	booktitle, vol, pages → author
	booktitle, vol, pages → title
	booktitle, year → vol
Inbook	author → yomi
	author, title → chapter
	author, title → editor
	author, title → pages
	author, title → publisher
	author, title → year
	author, title → edition
	author, title → month
	author, title → number
	author, title → series
	author, title → type

Document category	FDs
	author, title → volume
	publisher → address
Incollection	author → yomi
	author, title → booktitle
	author, title → year
	author, title → edition
	author, title → editor
	author, title → month
	author, title → number
	author, title → pages
	author, title → series
	author, title → type
	author, title → volume
	publisher → address
	booktitle → publisher
	booktitle, year → vol
	booktitle, pages → title
	booktitle, pages → author
Inproceedings	author → yomi
	author, title → booktitle
	author, title → edition
	author, title → month
	author, title → number
	author, title → organization
	author, title → pages
	author, title → publisher
	author, title → series
	author, title → volume
	publisher → address
	booktitle, vol, pages → author
	booktitle, vol, pages → title

Document category	FDs
	booktitle, year → vol
Manual	author → yomi author, title → edition author, title → month author, title → year organization → address
Masterthesis	author → yomi author, title → year author, title → month author, title → type author → school school → address
Misc	author → yomi author, title → year author, title → month
Phdthesis	author → yomi author, title → year author, title → month author, title → type author → school school → address
Proceedings	editor, title → year editor, title → month editor, title → number editor, title → organization editor, title → publisher editor, title → series editor, title → volume publisher → address
Techreport	author → yomi author, title → institution

Document category	FDs
	author, title → year author, title → month author, title → number author, title → type institution → address
Unpublished	author → yomi

索 引