07－開－10

データベース構築促進及び技術開発に関する報告書

大規模データベースにおける構造化情報抽出方式の調査

平成８年３月

財団法人　データベース振興センター
委 託 先　株式会社日本総合研究所

KEIRIN 00

この事業は、競輪の補助金を受けて実施したものである。

序

データベースは、わが国の情報化の進展上、重要な役割を果たすものと期待されている。今後、データベースの普及により、わが国において健全な高度情報化社会の形成が期待されている。さらに海外に対して提供可能なデータベースの整備は、国際的な情報化への貢献および自由な情報流通の確保の観点からも必要である。しかしながら、現在わが国で流通しているデータベースの中でわが国独自のものは１／３にすぎないのが現状であり、わが国のデータベースサービスひいてはバランスある情報産業の健全な発展を図るためには、わが国独自のデータベースの構築およびデータベース関連技術の研究開発を強力に促進し、データベースの拡充を図る必要がある。

このような要請に応えるために、（財）データベース振興センターでは日本自転車振興会から機械工業振興資金の交付を受けて、データベースの構築および技術開発について民間企業、団体等に対して委託事業を実施している。

委託事業の内容は、社会的、経済的、国際的に重要で、また地域および産業の発展の促進に寄与すると考えてられいるデータベースの構築とデータベース作成の効率化、流通の促進、利用の円滑化・容易化などに関係したソフトウェア技術・ハードウェア技術である。

本事業の推進に当って、当財団に学識経験者の方々で構成されるデータベース構築・技術開発促進委員会（委員長 前山梨学院大学教授 蓼沼良一氏）を設置している。

この「大規模データベースにおける構造化情報抽出方式の調査」は平成７年度のデータベースの構築促進および技術開発促進事業として、当財団が株式会社日本総合研究所に対して委託実施した課題の一つである。

この成果が、データベースに興味をお持ちの方々や諸分野の皆様方のお役に立てば幸いである。

なお、平成７年度データベースの構築促進および技術開発促進事業で実施した課題は次表のとおりである。

平成８年３月

財団法人 データベース振興センター

平成7年度　データベース構築・技術開発促進委託課題一覧

分　野	課　題　名	委　託　先
社　会	1　法的データベースにおける多分野データベースの統合一体的管理とオフ・オンラインの融合化に関する調査、研究	㈱日本法律情報センター
	2　新聞記事分類キーワードの標準モデル構築と自動付与に関する調査研究	㈱エレクトロニック・ライブラリー
中小企業振興 地域活性化	3　パソコンを用いた地図データベースの基礎構築	㈲朝日データサービス
	4　景観シミュレーション用樹木のデータベース構築	㈱ストゥディオサカイ
	5　包装機械データベースの構築	（社）日本包装機械工業会
	6　新規事業創出支援のためのデータベース構築に向けた基礎調査	㈱日本インテリジェントトラスト
	7　高効率化先進材料ファクトデータベースのパッケージ化	（財）次世代金属・複合材料研究開発協会
	8　阪神・淡路大震災の情報デジタル・アーカイブ	神戸マルチメディア・インターネット協議会
技　術	9　Ｍｏｓａｉｃの利用によるマルチメディアデータベース検索システムの構築	日本電子開発㈱
	10　大規模データベースにおける構造化情報抽出方式の調査	㈱日本総合研究所
	11　モーバイルデータベースシステムに関する調査研究	㈱イフ・アドバタイジング
	12　情報収集ロボットによるInternetでのWWW所在検索データベースの構築	日外アソシエーツ㈱
	13　インハウスデータベース用ＣＧＩ作成の調査研究	日本電子計算㈱

目 次

要約

インターネットの普及、テキスト処理技術やハードウェア性能の向上に伴い、我々がアクセス可能な情報資源が拡大している。それら大規模化した情報資源を管理し、ユーザビリティを高める技術の開発が求められている。

本報告書では、情報管理ツールの現状を大規模化克服の観点からサーベイするとともに、情報資源の拡大要因を、

- 量的拡大
- 範囲の拡大
- 表現方式の拡大

の３つの観点で捉え、それぞれ構造化情報を抽出する方式の現状と展望について述べている。

大規模化の各要因に対応する技術として、それぞれ以下の技術をとりあげている。

1. KDD（Knowledge Discovery from Database）を用いて一定データ群から情報構造を抽出する。
2. 分散人工知能技術を用いて、ネットワーク上の複数の情報ソースから構造化情報を抽出する。
3. マルチモダリティデータベースから構造化情報抽出に適したモダリティを選択し、構造化情報を抽出する。

1 はじめに

インターネットや高性能なデータベース処理機の普及に伴い、ユーザが利用可能な情報資源は大幅に拡大している。また、データベースを利用目的も多様化し、画像データ（静止画、動画）を中心とするマルチメディアデータ処理に対する要請も高まってきている。

広大な情報空間から、ユーザの希望するレコードを検索する技術の確立が求められている。これまでに、情報検索を援助する手段としてインデックスや書誌情報などの文献の情報構造に関する技術開発が行われてきた。

データの巨大化にともない、情報抽出の高機能化、構造抽出の自動化などに関する技術開発が求められている。さらに、データ表現形式の多様化に伴いデータの形態も考慮した構造情報抽出技術開発が求められている。

本課題では、機械学習、協調分散型人工知能技術を用いてより高度な情報の構造化をはかると同時に、ビデオニュース情報のようなマルチモダリティデータベースについてその適用可能性を検討する。

情報資源の拡大要因は以下の３つの観点で捉えられる。

- 量的拡大
- 範囲の拡大
- 表現方式の拡大

上記の大規模化の各要因に対応する技術として、それぞれ以下の技術をとりあげる。

1. KDD（Knowledge Discovery from Database）を用いて一定データ群から情報構造を抽出する。
2. 分散人工知能技術を用いて、ネットワーク上の複数の情報ソースから構造化情報を抽出する。
3. マルチモダリティデータベースから構造化情報抽出に適したモダリティを選択し、構造化情報を抽出する。

これらの方式のいずれもが、その有用性について多数の指摘がなされてきたが、提案されている方式や概念に関するコンセンサスが得られていない。さらに、大規模データベースへの適用を前提とした研究や方式の提案も少ない。

2 情報検索システムの現状

本章ではまず現状の情報検索/管理システムに関するサーベイを行なう。現状のシステムの利点、問題点を明らかにするとともに、大規模化がユーザと管理者の双方に与える影響について考察する。

2.1 これまでの情報管理技術

過去数十年間にわたって情報管理システムそして文献検索システムの研究や開発が行なわれてきた。これらのシステムを支える基盤技術を網羅的にあげると以下のようになる。

- 検索方式
  - 一次情報に対する検索
  - 二次情報を用いた検索
- 蓄積方式
  - フラットファイル
  - 各アプリケーションフォーマット
  - 文書構造化
    - SGML
    - その他の構造化則
  - 検索ツール
    - シソーラス（同義語辞書）
    - 自動テキスト処理ツール
    - サーチエンジン

これらの中で最も基本となるのはフラットテキストを媒体とした技術であり、頻繁に使用される手法である。しかし、フラットテキストの検索で良く用いられるフリーターム検索は約 500,000 ページを越えるとパフォーマンスが非線形的に低下するといわれ、大規模化に対応する上で問題がある。さらに、検索精度に関する問題も指摘されており、他の検索方式と組み合わせた方式を導入する必要がある。

検索プロセスを支援する技法として以下のような技法が提案、開発されている。

- **ブール演算子**
  検索条件は “AND”、“OR”、“NOT” などの論理演算子で結合して指定できる。ブール演算演算子による検索には、次のような問題点がある。
  - 検索文が長く、複雑になる。
  - ブール演算演算子で表現される文の持つセマンティックスと、ユーザが要求するセマンティクスが一致しないことがある。
  - 関連性の順位を表現することが難しい。

- **ストップワード**
  ストップワードリストは検索に用いるべきではない語のリストである。たとえば英語の場合、“a” や “the” といった語は検索語として用いるべきではない。ストップワードはフリーターム検索やインデックス作成において効果を持つ。しかし言語に依存する部分が多く、構文解析と同時に意味解析が必要な日本語においては、その実現が特に難しい。

- **インデックス**
  インデックスはテキスト検索で最も多く用いられるツールである。インデックス化される対象も日常的な語から専門用語、またシソーラスのようにインデックが意味的構造化されているものからそのような構造を持たないものまでと、さまざまなレベルのインデックスが提供されている。一方で、インデックスの作成やメンテナンスの作成は、ほとんど人力に依存しており情報量の増大に追従していない。

- **近接サーチ**
  単純な検索ではなく文や段落といったドキュメントが持つ構造を考慮した検索である。例えば「“人工” という語と “知能” という語が同一文の範囲で出現するもの」といった検索を行なう。海外の検索システムでは近接サーチをサポートしたシステムが多いが、わが国の検索システムで近接サーチを行なえるシステムは少数である。近接サーチはターゲットとなる言語に依存する部分があり、今後日本語に対応した方式の開発が望まれる。

- **インバーテッドファイル**
  インバーテッドファイルはキーワードがオリジナルテキストで出現する位置や頻度に関する情報を集めたファイルである。この技法によりインデックス処理を高速化できる。基本的な処理が自動化でき、容量の大きなデータが含まれいる場合この技法は有効である。

2.1.1 ハイパーテキスト

ハイパーテキストは 1940 年代後半に Bush[2] によって提案されたテキストアクセスの技法である。ハイパーテキストはテキストの部分と他の部分あるいは他のテキスト

をハイパーリンク、あるいは単にリンクと呼ばれるポインターで結合したテキストである。ユーザはテキストをシーケンシャルに読むだけではなく、リンクを辿って非線形的なアクセスを行なう。

ハイパーテキストの適用例はUNIXにおけるInfoシステムなど、ごく小数のシステムに限定されていた。しかし、1980年代に入りhttp（Hyertext transfer protocol）が提案、実装され、インターネットにおけるサービスの拡大に伴いネットワーク上での有力な情報提供手段となっている。

ハイパーテキストにおけるリンクは、検索の知識、対象とする概念の構造などを表現したものといえる。ハイパーテキストでは、ユーザがドキュメントに知識を手軽に埋め込める半面、リンクはプログラムにおけるgoto文のネストによるメンテナビリティの低下をまねく可能性ある。

現在、この点に対応するためにネットワーク上のリンクやドキュメントそしてリソースを検索するWWWロボットサービスが行なわれている。

2.2 情報検索における問題点

大量の情報源からユーザが求める情報を的確に検索するために、ナビゲーションツールや情報管理ツールを“知能化”し、その高度化を計る必要がある。まず、検索情報関連する問題点についてユーザそして管理者、それぞれの視点からの考察を行なう。

2.2.1 ユーザサイドからの問題点

ユーザが検索を行なうとき、その要求は曖昧なことが多い。また、ユーザのシステムモデルと検索システムが持つ検索モデルとが一致しない場合が多い。ユーザのが検索システムに対する要求を網羅的に列挙すると以下のようになる。

1. キーワードの綴がわからない。
2. 何をキーワードに使えば良いのかわからない。
3. キーワードで順々にしぼって行く時、前にもどりたい。
4. キーワードを“AND”、“OR”などの論理演算子でつないで指定したい。
5. 今どんな指定をしているか見たい。
6. 検索スピード。
7. 件数のめやすが出る（統計情報）。
8. アウトプットメディア（オンライン、紙、フロッピー、CDなど）。
9. 課金の仕組み。
10. タイトルや属性のブラウズを迅速にしたい。

11. タイトルや属性のブラウズを見やすくしたい。

これらの問題の大半はユーザの習熟度に起因するものであるが、情報検索ユーザの増大そして情報量の拡大により、今後さらに増大するものと予想できる。情報検索の困難性を増大させる要因として、以下の点があげられる。

- キーワードやインデックスなどの検索情報に関連する問題。
- ユーザインタフェースに関する問題。

これまでのインデックス技法は概念的なカテゴリや関連に関する知識を考慮していない。そのため知識の負荷を直接ユーザに負わせることになる。また、大半のアプリケーションではドキュメントの集まりの中から詳細かつ正確な情報を呼び出すためのに有効な問い合わせを公式化するような支援を与えていない。

グループウェアの普及やワープロの高機能化にともなって、ユーザのディスクトップでインデックシングやドキュメント間の関連づけ、あるいは検索などが頻繁に行なわれるようになって来た。このようなデータの発生点での処理はデータの分散化、大規模化に有効であろう。しかし、現在ワープロに附属しているのオンラインシソーラスは類義語処理が主体となっており、概念カテゴリの構築やハイパーテキストなどの機能がサポートされていない。すなわち、通常のワープロでは意味処理を含んだ検索支援は不十分である。

2.2.2 管理者サイドからの問題点

テキストで作成された情報を管理する最大の目的は、エンドユーザが検索して各自の情報ニーズに当てはまるドキュメントやドキュメントの一部を取り出すことである。管理者側からの問題点として、このようなインデックシング作業の困難さがあげられる。

インデックシングは対象とする領域に関する専門的な知識が要求さる。この作業の大半は専門的知識を持ったエキスパートがすべて人手で行なっており、バックログの増大が指摘されてきた。研究開発の学際化が進むにつれて関連する領域も増加し、さらに研究開発のサイクルが短くなり、そこから生まれて来る新しい概念の量も飛躍的に増大している。

また、著作や出版の電子化により雑誌や論文誌の発行件数が年々増加しているが、これらの間の書式は統一されておらず、配布媒体も様々媒体である。このような“表現”の多様性は、単に電子的な媒体やそのフォーマットに限らず表記方法にも見られる。たとえば、日本語では同一の用語を表すために以下のような多様な表現が可能である。

- 漢字
- カタカナ（半角と全角）
- ひらがな
- アルファベット（半角と全角）

- 送りがな
- 強制改行（日本語にはハイフォネーションが無い）

これら表現のうち、ユーザの検索を阻害するような多様な表記方法は一定の是正されるべきである。さらに、このような文字が混在しているはドキュメントの機械的処理を著しく阻害する可能性がある。

電子化によって処理すべき情報量が増大しただけではなく、作業項目あるいは作業量もかえって増大しているのが現状である。このような状況はインデックス作成の作業を著しく増大させ、バックログをさらに増大させることもなっている。

2.3 情報管理システム

これまでにJICSTによる科学技術文献速報やOPACに代表されるような図書館、大規模情報プロバイダーによる情報検索サービスが提供されてきた。そして、それらをサポートするのツールやシステムの開発が行なわれてきた。これらのシステムの大半はセンター運用を前提としており、管理者とユーザの境界が明確な環境用のシステムがほとんどであった。

近年のダウンサイジング化そしてネットワーキングの進展により比較的小規模なコンピュータ環境でも情報管理システムの構築が行なわれている。システムのユーザや管理者も専門家から、エンドユーザであること増加している。このような状況を背景として、小規模なシステムにも対応できるパーソナルコンピュータやワークステーションを用いた情報管理システムが製品化されている。

情報管理システムを選択する際に以下のような考慮点が指摘されている[11]。

- ブール関数
- 近接サーチ
- ワイルドカード
- 自然言語

- 検索時間
- 検索の記述タイプ
- インデックス併用の有無
- シソーラス機能
- 検索の最適化とストレージ圧縮の有無
- 再現率と適合率
- マーキング機能の有無

- ドキュメント間のリンクの必要性
- インデックス作成機能
- グラフィック情報の有無
- フォーマットコンバージョン機能の有無
- 習熟に必要な期間
- ネットワーク対応

1990年代以降製品化された情報管理システムの大半が、クライアント/サーバモデルを前提としている。また、多くの製品がリレーショナルデータベースを中核にし、システムにおけるすべてのデータアクセスとデータ変更はすべてデータベース管理システムで行なわれる。

また、ハイパーテキストを基本モデルとした情報管理ツールも製品化されており、テキスト、イメージ、データ、グラフィックス、音声など多様なメディアに対するシームレスなアクセスを提供しているツールも多い。

このような情報管理ツールは以下のようなサブシステムから構成されている。

- データベース管理機能
- 検索機能
- インタフェース構築ツール
- 非テキストデータ格納機能

2.3.1 データベース管理システム

データベース管理システムはテキストデータベースの変更、操作のモニタリング、ログの採取などを行なうモジュールである。製品の多くが以下のような多様なフォーマットデータに対応している。

- ワープロファイル
- DECのDDIF[1]やIBMのDCA/MODCA[2]などのメーカ独自フォーマット
- SGMLドキュメント

ドキュメントはシステム内部では各製品独自な物理フォーマットで格納されている。また、格納されたドキュメントの論理構造は以下のようなのタイプに分類される。

[1]DDIFは米国DEC社の登録商標である。
[2]DCA/MODCAは米国IBM社の登録商標である。

- フラットテキスト
- ドキュメント本体とそのドキュメントの属性で記述されたデータから構成される。例えばドキュメントがあるメールである場合、この属性は記入者/日付/テーマ/宛先/発信者などである。
- 章、節といった文書構造を表現するために、文書構造を表す属性を持つタイプ。ある１つのドキュメントは複数のパラグラフから構成され、あるパラグラフは複数の文から構成されるといった階層的構造でドキュメントを表現する。これらの各構造単位に構造を表現する、あるいは各構造の内容を表象したラベルが付けられる。このラベルはドキュメントやデータベースを管理したり、検索処理を行なう時に使用される。

さらに、蓄積されたドキュメントに関するデータディクショナリ機能をサポートしているシステムも多い。データディクショナリとドキュメント構造に関する属性を併せて使うことにより、ある用語が含まれるドキュメントや章や節そして文などをトレースできる。

2.3.2 検索機能

ほとんどのプロダクトがブール関数、近接サーチ、ワイルドカードといった標準的なサーチをサポートしている。さらに、インデックス、シソーラス（類義語、概念階層）などの二次情報を使用した検索をサポートしている。

これらの一般的な検索機能の他に次のような機構を使って検索を行なえるシステムもある。

- ハイパーテキストによるリンク
- 知識ベースによる検索
- ワードベクターあるいは概念空間による検索

ワードベクターあるいは概念空間に基づいたシステムでは各概念を定義する機能と、その概念を用いて検索を行なう機能が提供されている。この概念は一つの単語あるいは複合語だけではなく、階層的構造として表現される場合もある。このような形式で表現された概念は、他の概念となりうる下位概念に分解できる。システムはデータベース中にある各ドキュメントを評価し、単語頻度やテキスト頻度などの関連性基準からドキュメントの内容と各概念との関連性を決定する。この概念木の各枝に重み付け係数を付けることにより、ユーザのある関心事項を表現することも可能である。ユーザはこの重みを使ってある概念を構成する用語のそれぞれの重要度を定義する。

ユーザが与えた検索条件に含まれる概念と、各ドキュメントに含まれるドキュメントとの間の関連性評価として検索は表される。検索条件として与えられた概念との適合度が高いドキュメントは関連ランクが高くなり、検索が完了した後で作成されるドキュメントリストの上位に表示される。また、このような概念木を使って、データベー

ス中の各ドキュメントを一定の概念空間上に配置したり、あるドキュメント間の類似性判定を行なうことも可能である。

ワードベクターベースを使った検索を行なうために、専用の演算子を提供しているシステムもある。たとえば、ある演算子はドキュメントに含まれている重み付けられた全用語を集めて、そのドキュメントのスコアを計算する。この演算子で得られた結果は、その値が高いドキュメントほど与えられた概念との関連度が高いと解釈できる。

2.3.3 インタフェース構築ツール

検索や検索結果に対するインタフェースを提供する他に、検索結果数の表示、検索中に使用した用語から用語への動的なリンク機能などをサポートしている。

既存の各製品が提供しているインタフェースを以下のように列挙できる。

- メニュー方式のインタフェース
- キャラクターインタフェース
- マルチウインドウベースのインタフェース
- コマンドベースのインタフェース
  - SQL ライクの検索コマンド
  - 製品独自コマンド

インタフェースの好み各人多様であり、ウィンドウベースのインタフェースをサポートすれば“良い”インタフェースが提供できる訳ではない。多様なインタフェースが提供され、各ユーザが自分に合った形態を選択できるインタフェースこそが最良のインタフェースといえる。

2.3.4 非テキストデータ格納機能

イメージデータや音声といった非テキストデータを格納/管理するためのサブシステムである。非テキストデータのフォーマットはテキストデータ以上に多岐に渡っている。テキストデータの格納や検索結果の表示などにおいては、このような多様なフォーマットを他のフォーマットに変換したり、また表示/再製するドライバーが重要な機能となる。

このような非テキストベデータースに格納されたデータにコメントを付けたり、他のドキュメント構成要素とともに表示したりできるが、一度格納されると編集しにくくなるシステムが多い。

2.4 情報管理ツールの評価

ドキュメントデータベースの利用の観点からみれば、ドキュメントはそのコンテンツによって管理されるべきであろう。コンテンツとはそのドキュメントが伝えようとして用いている言葉ではなく、伝えようとしている事柄である。コンテンツ検索は、内容によって分類されたドキュメントでデータベースが構成され、あるテーマでの絞り込みが可能なドキュメント集合に適している。インデックスやシソーラスのような意味主体の構造化や意味的構造化情報は、ある概念に関する知識量が少ないユーザによる検索を支援する強力な情報になる。

構造化情報の作成やそれらに基づいたドキュメントの構造化は、各ドキュメントをデータベースに登録する際、専門家によって評価され処理される。先にも述べたように、この作業の大半は人力にたよっており大規模化に対応していけない。また、通常の情報管理ツールの大半がこの問題をブレークスルーするような有力な技術を提供していない。

意味情報の構造化と並んで、章や節などの単位でドキュメントの構造化（以下、文書構造化という）が行なわれている。文書構造化は近接サーチのようなコンテンツベース検索に近い検索を行なう時に、有力な情報源となる。文書構造化情報をドキュメントに埋め込む有力な技術としてSGMLがあり、近年の“CALS過熱”現象に伴ってSGMLをサポートするツールが製品化されている。しかし、DTDに対するコンセンサス作りの立ち遅れから互換性のないSGML化が行なわれようとしている。このような互換性のないドキュメントの氾濫は管理者の作業やユーザの困惑を増加させることにもなっている。

3 大規模化の克服

本章ではドキュメントデータベースにおける構造化情報および、それらと検索/管理機構との関連について述べる。最後に、"浅い" 処理といった観点を導入し検索処理における意義について考察する。

膨大な情報からの検索処理において、テキスト検索が重要な機能となることは言うまでもない。また、非テキスト情報を検索したり処理したりする上でも、テキストを非テキストデータの意味的記述子として使用することが多い。前節で述べたように、フリーターム検索やキーワードといったドキュメントを検索するための技術開発が行われて来た。また、テキスト情報の処理を容易にするために文書構造化に関する技術開発も行われている。

前者のようなテキストの意味内容を対象とする技術は文書間構造化を行なう技術であり、後者の技術は文書内の構造化を行なう技術とも言える。これらの技術は相互排他的なものではなく、互いに関連し合うものである。

これまでの研究ではあるキーワードが適切であるか否かを判定する基準として、以下のような数値的基準を用いてきた。

- 再現率

$$R = \frac{\text{検索された関連するアイテム}}{\text{全体の中で関連するアイテムのトータル}}$$

- 適合率

$$P = \frac{\text{検索された関連するアイテム}}{\text{検索されたアイテムのトータル}}$$

再現率とは、そのキーワードが検索語として指定された時、目的の文献がどの程度効率良く検索されるかを表す基準である。一方、適合率とはそのキーワードが想起され実際に検索ワードとして使用される可能性を表す基準である。効率性基準と想起性基準の双方から見て高い水準にある単語が検索語として採用されてきた。しかし、この方式には以下のような問題点もある。

- 単語の集合ような自然の距離の定義されない集合上の関数の最適化に、通常の局所的な最適化手法を用いることは不適切である。
- 検索者の置かれている状況には、検索者の言語知識や検索目標などにかかわる個人的要因、検索対象となる文献データベースにかかわる背景などの多種多様な要因が含まれる。これらの要因を全て考慮して、単語の使用頻度を求めることは非常に困難である。

- 情報とは、あるデータに対してユーザーが意味づけをしたものである。この意味づけには、個々のユーザーがおかれている状況と関連する。同じデータであっても、状況によっては情報となったりたり、単なるノイズであったりする。

3.1 テキスト検索

基本的な検索プロセスにおけるユーザの行動には以下のような特徴がある。

- テキスト情報のアクセスを行なう場合、我々はブラウズと呼ばれる流し読みを頻繁に行なう。電子化された情報と紙ベースの情報との違いは、このブラウズのしやすさの違いにある。ブラウズが頻繁に行なわれる行動でありながら、その認知的なメカニズムはあまり解明されていない。

- ブラウズが電子化された情報と紙ベースの情報とで異なるように、データとテキスト間でも異なる。データを検索する場合、ユーザは「部品番号が 0023 の部品」といったように検索のターゲットとすべき値を知っている。ユーザが明確な目的を持たずにデータをブラウズすることはほとんどない。

- ユーザがある情報を検索する場合、まず一次情報をブラウズしたり一定の検索語を指定して検索を行なう。直接一次情報から情報を検索できない場合、二次情報を使った検索を行なう。また、一次情報の量が膨大であることがあらかじめ予想される場合、最初から二次情報を用いた検索を行なう。

現状の情報検索システムでは、以下のような機能がほとんどの環境でサポートされている。電子化された情報に対するアクセスはこれらの機能を使って行なう。

1. 一次情報
2. 二次情報
3. ユーザインタフェース

検索プロセスは一般に探査プロセスであり、試行錯誤的な過程を多く含む。また、効率的な検索を行なうためには、

1. ツールに関する知識
2. 対象領域の知識
3. 各データベース固有の方式

などに関する知識が求められる。ユーザがエキスパートではない場合、所望の情報にたどり着くためにはかなりの時間が必要となる。

3.2 構造化情報

ユーザからの不特定多数の検索要求に備えて情報を構造化する手段として、これまでにも多くのツールや技法の開発されてきた。主なものを列挙すると以下のようになる。

- 電子化目録
- 書誌情報の作成/管理
- 自動インデックス作成
- シソーラス
- 文書構造化技法

これらの情報構造化技術の中でもっとも研究され、かつ体系的な整理が行なわれてきたのがシソーラスである。

3.2.1 シソーラス

インデックスやシソーラス対象領域の概念の意味構造を表現しており、特に専門用語を構造化した辞書はシソーラスと呼ばれる。シソーラスは古くから情報検索や情報の意味的な構造化に頻繁に使われてきたツールであり、CDMARCS（米国国会図書館）や JICST のシソーラスなどが知られている。

ある専門用語の間の関係を表現する２つの関係がある。

1. 名詞間の関係
2. 名詞と動詞の関係

さらに、これらの関係は以下のカテゴリに分類できる。

1. 先行する用語
2. 広義の用語
3. 狭義の用語
4. 同等関係にある用語
5. 使用可能語
6. 関連用語
7. 反意語
8. 複合語

9. 派生語

10. その他の関係

    - 部分–全体関係
    - 順序関係
    - 因果関係
    - 論理関係

DESCRIPTOR	WS
UF	work station
	workstation
NT	office WS
	engineering WS
	graphic WS
	LISP machine
BT	small size computer
TT	computer
RT	intelligent terminal
	X-window terminal
	computer network
	network
	⋮

Figure 3.1: シソーラスの定義例

シソーラスにおける、ある専門用語定義の例を図 3.1に示めす [23]。シソーラスは検索条件や文献で使用される用語の意味関係を記述することにより、その用語群が表現する概念の構造を表わしている。シソーラスは大規模な意味構造情報であると同時に、“よく” 構造化された知識ベースである。シソーラスは未知の概念を検索する場合、強力なナビゲーションツールとなる。

シソーラスの作成やメンテナンスを自動化しようとするシステムの開発は、過去長期間にわたって行なわれて来た。しかし、完全に実用化されたシステムは少なく、シソーラスの作成やメンテナンス作業の大半は人力に頼っている。このことから、メンテナンスされるまでのタイムラグや必要な作業量の点で問題が多い。

3.3 情報抽出における知識処理応用アプローチ

知識処理を応用したアプローチの適用対象として以下のようなアプリケーションが考えられる。

- **サーチ文字列生成**
  ある概念を検索したい場合どのような検索語をどのような条件で指定したらよいか、また、条件の組合せを行いたいどのように記述したらよいかなど。

- **概念ベースクエリー**
  対象とする文書を解析して得られた概念木とシソーラスなどの構造化情報における概念木の照合による主題推定、または構造化情報の自己組織化など。

- **自然言語インタフェース**
  これまでに主に行なわれてきた研究は、ある検索にどの用語を使うべきかということが主要テーマであった。今後はインテリジェントエージェントと関連した研究やシステム開発が主流となっていくであろう。また、ユーザの検索パターンに現れる同義語を学習し、用語の精緻化プロセスに使用することも考えられる。

- **知識に基づくアプリケーション**
  FAQ（Frequently Asked Question）やCBR（Case-Based Reasoning）を応用したヘルプディスシステムなどは元テキストに多量の知識が埋めこめれたシステムといえる。このようなシステムを使ったユーザ支援過程におけるデータの再編性は、そのまま知識の構造化を行なっているといえよう。

- **検索方略**
  ある検索を行なう場合、どの検索方略をとるか、またどのように検索方略を混合するかといった問題に知識処理を応用する。あるエキスパートの判断を知識ベース化する。ここで知識ベース化される知識とは、

  - データベースの選択方法、
  - 主題とフィールドとの関係
  - 検索語の系列と条件の指定方法
  - 主題に対する用語の重みづけ

  などが考えられる。

3.4 ニューラルネット応用システム

ニューラルネットは特定の入力によって自己の状態を変える並列したノード情報を相互結合することによって、パターンを認識するものである。ニューラルネットは前提知識を持たない状態からスタートし、各ノード間の相互結合の重みづけを変更することにより学習を行なう。

ニューラルネットはパターンに関連する認識処理のために使われる。たとえば、ある音のパターン認識やテキスト中の同一パターンを発見することである。重み付けプロセスは他の分類方法に比べれば、一部データの欠損に対して強靭であので、ニューラルネットは特に入力あるいはプローブデータが不完全なパターンを認識することに役立つとされている。

3.4.1 ニューロンのモデル

ニューロンを情報処理素子とみれば、n 個の入力信号を受け取り、それをもとに計算して出力 zを答えとして出す素子である。入力信号 $x_1, x_2, \cdots, x_n$とし、その強さは $0 \leq x_n \leq 1$ である実数とする。

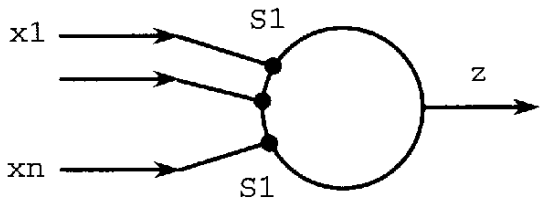

Figure 3.2: ニューロの数理モデル

いま $s_1, s_2, \cdots, s_n$を入力シナプス間の効率（重み）とする。このとき入力信号全体の影響は重み付きで総和されて、

$$\sum_{i=1}^{n} s_i x_i$$

となる。いま h をしきい値としてこれを差し引いたものを、

$$u = \sum_{i=1}^{n} s_i x_i - h \tag{3.1}$$

とする。

この u に対して出力関数、

$$sgn(u) = \begin{cases} 1, & u > 0 \text{ のとき} \\ -1, & u \leq 0 \text{ のとき} \end{cases} \tag{3.2}$$

を定義すると、各ニューロに対する入出力関係は以下のように定義できる。

$$z = f(u) = f(\sum s_i x_i - h) \tag{3.3}$$

実際のニューラルネットはこのようなニューロが相互接続されたネットワークである。もっとも単純なニューラルネトは図3.3に示すように、m 個のニューロがならんでいて、ここに n 個のニューロから共通の入力、$x = (x_1, x_2, \cdots, x_n)$ が入ってくるものである。また、この入力に対する出力は、$x = (z_1, z_2, \cdots, z_n)$ で定義できる。

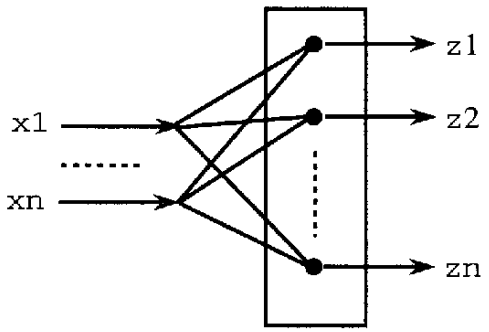

Figure 3.3: 層状変換神経回路網

いま、入力 x_iが第 j番目のニューロと結合するときの結合効率を s_{ji}とする。すると、第 j番目のニューロの出力 z_jは、

$$z_i = f(\sum_{i=1}^{n} s_{ij}x_i - h)$$

と定義できる。これを行列表現して、$z = f(Sx - h)$ と表現する。

もし、x 以外にも外部からニューラルネットに入力があるときは、第 j番目のニューロに入る外部刺激の総和を a_jと

3.4.2 学習方式とバックプロパゲーション

ニューラルネットにおける学習とは、各ニューロ間の結合が変化していくことである。ある一つのニューロの入力信号 $x_1, \cdots, x_n$の結合効率 $s_1, \cdots, s_n$の変化は、

$$\tau' \frac{ds_i}{dt} = -s_i + crx_i \tag{3.4}$$

と表現できる。ここで、rはシナプス効率を変えるタイミングを指定する条件信号であり、ニューロの入出力関係は、

$$z = f(\sum s_i x_i - h)$$

と表現される。

この rの決め方には種々なモデルがあるが、古典的モデルとして各ニューロの出力信号 zを s_iとするヘッブの学習則が知られている。このような特性を持つニューロが相互接続された回路網においては一つのニューロの学習結果が他のニューロに影響を与える。

競合学習条件下において、ニューロが特定の信号 x に対しては興奮し、他の信号に対しては興奮しないような信号選択規則が提案されている。例えば、マルスブルグ [56] によって次のような信号選択則が提案されている。

$$\sum_{i=1}^{n} s_i = \text{一定}$$

相互接続したニューロが層状回路を構成したとき、学習については以下のようにモデル化される。例えば、ある信号 x に対して誤りの大きさを定義する関数を以下のように定義する。

$$\frac{1}{2}\sum_{x}|s\cdot x-y_d(x)|^2 \tag{3.5}$$

ここで、y_dは期待される出力信号を表す。上式は実際の出力と期待される出力信号の差を表し、学習はこの差を最少化するように行なわれる。

層状のニューラルネットの場合、中間層のニューロはそれぞれ部分情報しか計算していないことから、どのニューロを改善したらよいかを判定することは難しのでニューロネット全体でこれを判定する。

学習したいと思うパラメータをすべてまとめて、$\theta=(\theta_1,\theta_2,\cdots)$ とする。すると、入出力関係はパラメータがθのネットワークでは、

$$z=z(a;\theta)$$

となる。3層の神経回路網の場合これは、

$$z=f(\sum_i s_i f(\sum_j w_{ij}a_j))$$

となる。a という入力をパラメータθのネットワークで処理したときの損失を $l(a,\theta)$ とすると、

$$l(a,\theta)=|z(a,\theta)-y_d(a)|\,k(s\cdot x)$$

となる。ここでθを、

$$\theta\to\theta-c\frac{\partial l(a,\theta)}{\partial\theta}$$

と変えていけば L が減少する。

一般にこうした学習方法には欠点がある。それは $L(\theta)$ には極小点が一つとは限らず、局所的極小点に停留してしまう可能性があることである。現在、ボルツンマンマシンあるいは“焼きなまし法”がこの問題点を克服する唯一の方法である。

ネットワークを3層のネットワークとする場合、各ニューロがアナログ型出力関数 fをもつとし、出力ニューロは複数あると仮定する。このとき、入出力関係は

$$z_i=f(\sum s_i x_i),x_i=f(\sum w_{ij},a_i)$$

と表せる。一方、ある信号 a に対して望ましい出力信号をベクトル $y_d(a)$ とし、損失関数を

$$l(a,\theta)=\frac{1}{2}|z-y_d(a)|^2 \tag{3.6}$$

とする。ここで可変パラメータを s_iと w_{ij}とし、上記の学習方程式を適用すると、

$$\frac{\partial l}{\partial s_i}=(z_1-y_{di}(a))f'(\sum s_j x_j)x_i \tag{3.7}$$

と書けるので、

$$r_i = (z_i - y_{di}) f'(s \cdot x)$$

となる。これを学習信号とし、

$$s_i \to s_i - c r_i x_i$$

のように s_i を変える。中間層の学習 w_{ij} については、

$$\begin{aligned}
&\frac{\partial l}{\partial w_{ij}} \\
&= \frac{\partial l}{\partial x_i} \frac{\partial x_i}{\partial w_{ij}} \\
&= \sum_k (z_k - y_{dk}) \frac{\partial z_k}{\partial x_i} \frac{\partial x_i}{\partial w_{ij}} \\
&= (\sum_k r_k s_k) f'(\sum_m w_{im} a_m) a_j \\
&= \tilde{r}_i a_j
\end{aligned}$$

となる。ここで、中間層の第 i 番目のニューロの学習信号 $\tilde{r}_i$ は、$r_k S_k - f'$ である。こうして、$w_{ij} \to w_{ij} - c\tilde{r}_i a_j$ とすればよい。

信号は入力層→中間層→出力層の順で流れるが、学習を行なう際には誤差、

$$e_i = z_i - y_{di}$$

がまず最終層にあたえられ、ここで学習信号

$$r_i = e_1 f'(s \cdot x)$$

が作られる。次に中間層の学習信号はこの r_i が出力層のニューロから逆に伝わり、その時逆にシナップス結合の値 s_i を掛けられて伝わっていく。このようなモデルに基づいた回路網の学習法をバックプロパゲーションという。

3.4.3 情報検索への応用

ニューラルネットを応用した情報検索システムは情報検索に関する明示的な知識を利用しないにも関わらず、大量の元データを検索できる環境を提供する。この機能によりユーザの明示的な知識獲得を支援する。

大量であるがゆえに未整理で曖昧さをもったデータを対象とした情報検索を実現するためには以下のような機能が要求される [12]。

1. 不完全な問い合わせの処理
2. ユーザの意図の理解
3. 質問や検索結果の一般化

4. ユーザの要求の変化に対するサポート

5. 動的で適切なフィードバック

6. ブラウジングの支援

7. 状況依存性の付加

このような機能を実現するためにニューラルネット技術が注目されている [37]。ニューラルネット技術を用いた情報検索法は以下のように分類できる。

- **相互活性型情報検索**
  Mozer[35] によって提案された方式で、ニューラルネット中のノードが特定のキーワードや文献を表現するという意味で局所表現を用いた相互活性型ネットワークにより実現された情報検索方式である。

- **自己組織化マップ型情報検索**
  中間層のない2層型の教師なし競合学習の一種である Kohonen[27] の自己組織化マップを検索に利用した方式である。

- **分散表現型情報検索**
  特定の概念や文献がニューラルネット内の特定のノードに対応せず、複数のノードで表現される分散表現型ニューラルネットを情報検索に応用した方式である。分散型ニューラルネットは情報表現に必要な根源的要素を発見する能力や伸概念の獲得能力、汎化学習能力に優れていると言われている。

- **その他**
  教師なし競合学習を行なうニューラルネットと自己組織化マップを組み合わせた方式 [54]、連想をより正確におこなうために統計的手法や情報量に基づいた方法 [55] などが提案されている。

3.5 構造化情報の利用

ドキュメントや文章のもつ意味的構造情報や文書構造が明らかになるにつれて、それらを積極的に利用したシステムの研究開発や、製品化が行なわれている。本節以降でそのようなシステムの概要を実際のシステムに基づいて説明する。

まず、シソーラスやインデックスといった情報構造を自己組織化に使用したシステムの例として、情報ベースシステム IBS/SORITES[14] における帰納的推理機能 [22, 23] の概要を述べる。ここでの帰納的推理は、シソーラスを背景知識として一般化調整を行なっている。

さらに、ドキュメント構造を処理するシステムについて、その概略を示す。

3.5.1 情報ベースシステムとシソーラス

情報はドキュメント、テキスト、データベース、グラフィックといった様々なフォーマットで保持されている。情報ベースシステムはデータベース技術、文書処理技術、知識処理技術などを組み合わせたシステムであり、その概観を図3.4に示めす。

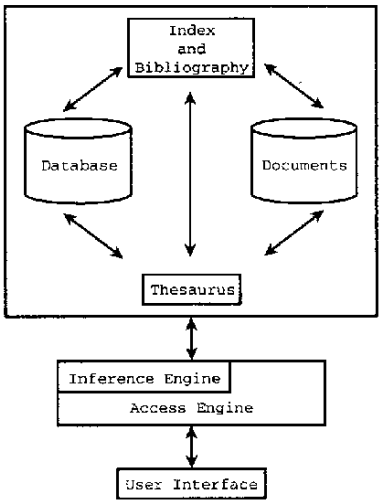

Figure 3.4: 情報ベースシステムの概要

情報ベースシステム上の情報は次の3タイプに分類される。

1. 文書構造情報
   (a) 書誌情報
   (b) インデックス
   (c) シソーラス
2. データベース
3. ドキュメント

シソーラスは、良く構造化された知識ベースの一種として考えられる。専門領域における概念は、シソーラスにおけるディスクリプターとして表現され、上位-下位関係(BT-NT 関係)は、知識ベースにおける'isa' 関係に対応する。また、部分-全体関係(BTP-NTP 関係)は知識ベースにおける'part-of' 関係あるいはエンティティ-属性関係に対応する。

3.5.2 帰納的推理と知識発見

ある事例群から、それらの事例群に共通な概念を得る推論を帰納的推理と呼ぶ。帰納的推論によって導出される概念は、ある集合を特徴付けるルールや定義である。

この機械学習のコンテクストにおける帰納的推理のフレームワークを以下ように表現できる。

$\mathcal{L}_E$は例を記述する言語、$\mathcal{L}_K$を背景知識を記述する言語、そして$\mathcal{L}_H$を仮説を記述する言語とする。$\mathcal{E}^+ \subseteq \mathcal{L}_E$を正の例の集合、$\mathcal{E}^- \subseteq \mathcal{L}_E$を負の例の集合、そしてと$\mathcal{K} \subseteq \mathcal{L}_K$を背景知識の集合とする。帰納推理により、$\mathcal{K} \cup H \vdash \mathcal{E}^+, \mathcal{K} \cup \mathcal{H} \nvdash \mathcal{E}^-$を満足する仮説$\mathcal{H} \subseteq \mathcal{L}_H$は与えられる。

一般化の概念は、帰納推理で有効な役割をはたし、非形式的には以下のように定義される。もし概念S_1が概念S_2より多くの概念を帰納するならば、S_1はS_2よりも一般的である。この定義で記述された一般化は内包関係に基づいている。もし$S_1 \subseteq S_2$ならばS_1がS_2によって含まれると言う。

帰納的推論に関する機械学習のコンテクストに基づいた研究は、これまでにも多数行なわれてきた。また、ID3[45]、EBL[10] といった著名なシステムの研究開発が行なわれている。

一方、近年データベースの巨大化やデータベースの高度処理に対するニーズの高まりなどを背景にして、データベースから知識発見（KDD）に関する研究が行なわれている。与えられる事例群に関する条件から、KDD と機械学習的帰納的学習ではその推論方法において大きな差異がある。機械学習のコンテクストにおける帰納的推論、例えば説明に基づく学習 [10] では 2 つの制約、完全性制約と一貫性制約を帰納的推論に使用する。

一貫性制約は、正およびの負の事例から構成された事例集合からの帰納的推論の基礎となる。一般的に、データベースには正の事例群だけがストアされ、負の事例群は無いと仮定するべきであろう。ゆえに、データベースからの知識発見や帰納的推論に一貫性制約を適用できない。このことは、KDD において事例や知識の一般化のための合理的な制約がないことを意味している。正の事例だけを用いて帰納的推論を行なうために、知識の過一般化を回避するための新たなの制約を加する必要がある。

3.5.3 情報ベース上の帰納推論

IBS/SORITES における帰納的推論は、以下の手続きから構成される。

1. 事例として与えられたタプル間の論理積を求める。

2. 帰納的推論の背景知識としてシソーラスを使用する。

シソーラスは、ψ-term[1] の表現で表され、推論システムの入力/出力もψ-term で表現される。この項表現は以下の形式で表現される。

定義 3.5.1 *[項表現]*

$$< term >$$

$$::= \ '('< string > \{< string > \ldots\}')'|$$
$$'('< string > \rightarrow' ('< term >')'$$

たとえば、シソーラス上のディスクリプタは以下のように表現される。

("WS" (
BT → ("small size computer"),
TT → ("computer"),
NT → ("office WS",
"engineering WS",
"graphic WS",
"LISP machine",
"PC"))).

この項表現で、トップレベルの "car" 部（たとえば、"WS"）はディスクリプターであり、概念名と呼ぶ。各要素の"cdr"部（たとえば"BT → ("small size computer")"）をフィールドと呼ぶ。フィールドの第1要素を属性名と呼び、そして他の要素を属性値値と呼ぶ。この表現によるデータベースのタプル表現は以下のようになる。

("PC" (
name → ("IBM AT"),
CPU → ("i8086"),
memory → ("256k"))).

IBS/SORITES における帰納推論は、López[28] による認知的競合解消方略に基づいた競合解消方略使用する。

典型性： もし、ある属性下のある値がその関係の中で一般的であるならばその値を採用する。

単調性： もし、ある属性下のある値がその事例間で一般的であるならばその値を採用する。

多様性 もし、ある属性下のある値がその "is-a" 階層における同じレベルの関係で一般的であるならばその値を採用する。

同質性： もし、ある値が同一の BT ツリー下にあるならば、その値を採用する。

非単調性： もし、ある値が同一の BT ツリー下に無いならば、その値を採用する。

本研究における帰納的推論では、典型性方略と単調性方略だけを使用している。典型性方略と単調性方略を適用するために、一般性に関する尺度を次のように定義する。

定義 3.5.2 *[一般性]* あるの条件を満足する値の数が閾値を越えるとき、その値は一般的であると言い、値の一般性は、次式で計算される。

$$\frac{\text{ターゲットとする値を含んでいるタプル数}}{\text{タプルの総数}} \geq 0.8 \qquad (3.8)$$

IBS/SORITES 上の帰納的推理は、与えられた例の論理積を得ることによるインクリメンタルな帰納的推理であり、レコード間の類似性からフィールド間の類似パターンを得る。

帰納的推理によって生成された項は、与えられた事例からフィールド間の論理関係を表すものである。システムに入力として与えられる事例群は、ユーザによってリある関係から選択されたタプル群である。基本的な帰納的な推論の手続きは、以下のようになる。

定義 3.5.3 *[帰納推論アルゴリズム 1]*

1. 各タプルの属性の積を得る。
2. 各属性値の積を得る。
3. ステップ 2 の結果が空の場合、それを表す記号によってそれを置き換える。

この手続きによって生成される帰納的概念のほとんどは、過一般化され過ぎている。この過一般化は各タプルの値の多様性に起因するものであり、これを避けることはむづかしい。そこで前章で示した競合解消方略を適用して、以下に示すように手続きを変更する。

定義 3.5.4 *[帰納推論アルゴリズム 2]*

1. 各タプルの属性の積を得る。
2. 各属性値の積を得る。
3. 典型性方略を適用する。
4. 単調性方略を適用する。
5. ステップ 2 の結果が空の場合、それを表す記号によってそれを置き換える。

帰納推論アルゴリズム 2 によっても、過一般化を回避できない場合がある。この過一般化を防止するために、シソーラス上の BT–NT 関係を帰納推論に適用し、値を概念木上のより広い概念で置き換える。以下に示す手続きは、シソーラスを使用した帰納推理の手続きである。

1. ある属性の値がシソーラス上にあるならば、その "BT" 値で置き換え、その値を新しい値とする。このサーチプロセスは、シソーラス上の各ツリーの最上位概念（TT）で終了させる。

name	CPU	memory	price
PC/AT	i8086	256k	1000
Compaq	i80386	16M	15000
PS/2	i80386	16M	20000
⋮	⋮	⋮	⋮

Table 3.1: 関係：パーソナルコンピュータ

name	CPU	memory	Network	price
Sun4	SPARC	16M	Token Ring	16000
P Station	SPARC	28M	Ethernet	25000
N 3000	R3000	16M	Ethernet	20000
⋮	⋮	⋮	⋮	

Table 3.2: 関係：オフィスワークステーション

2. 各事例の属性の積とる。

3. その属性の値の積をえる。

4. ステップ 2 の結果が空の場合、それを表す記号によってそれを置き換える。

3.5.4 実行例

データベースは関係 “オフィスワークステーション”、関係 “パーソナルコンピュータ”、そして関係 “エンジニアリングワークステーション” から構成されているとする。

関係 “パーソナルコンピュータ” および “オフィスワークステーション’ に次の SQL コマンドを発行する。

```
SELECT * FROM relation-name
    WHERE PRICE > 1000;
```

この問合せの結果を帰納推論の入力とし、帰納推論アルゴリズム 1 をトレースする。

1. 各関係の共通属性名は、“name, CPU, memory,price” である。

2. 例に示されているように、どの属性の共通値もヌル値である。

name	CPU	memory	Network	price
S Station10	SPARC	16M	Token Ring	16000
HP9000	R3000	16M	Ethernet	20000
S Station20	SPARC	28M	Ethernet	25000
⋮	⋮	⋮		

Table 3.3: 関係：エンジニアリングワークステーション

3. 問合せの結果に典型性方略を適用する。属性 “memory” の値がほとんどの結果において 16M バイトより大きいならば、この値を採用する。属性 “price” の値が、同じ方法で取り扱われる。

4. 単調性方略を適用する。もし、属性 “CPU” の値が関係上のほとんどタプルで “i80386” もしくは “SPARC” ならば、これらの値を採用する。

最後に次の結果を得る。

(”result001”
(CPU → (”i80386”,”SPARC”))
(memory → (”> 16M”))
(price → (”> 10000”)))

この例は、帰納的推理の結果としてパーソナルコンピュータとオフィスワークステーションの共通特性を表現している。

次にシソーラスを用いた帰納推論実行例を示す。この例では図 3.5で示されるシソーラスを用いる。

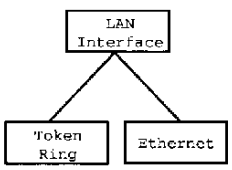

Figure 3.5: 例として用いたシソーラスの定義例

関係 “オフィスワークステーション” および “エンジニアリングワークステーション” に以下の SQL マンドを発行し、その問合せ結果を帰納推論の入力とする。

SELECT * FROM reaction-name
WHERE PRICE >1000;

帰納推論アルゴリズム2の実行例をトレースする。

1. “Token Ring” と “Ether” の BT 語を得る。その BT 語で各語をが置き換える。
2. これらの例の共通属性名は、“name,CPU,memory,network,price” である。
3. 各属性の積をとる。

最後に次の結果を得る。

```
("result002"
        (Network → ("LAN Interface")))
```

この帰納推論の結果は、オフィスワークステーションとエンジニアリングワークステーションの共通特性を示している。

ここまでに述べた2つの手続きを同時に適用することにより、以下の結果が得られる。

```
("result003"
        (CPU → ("i80386","SPARC"))
        (memory → ("> 16M"))
        (Network → ("LAN Interface"))
        (price → ("> 10000"))).
```

3.6 AnchorPage

AnchorPage[1] は Web に対するインデックス、要約の作成そしてハイパーテキストのリンク作成を自動的に行なうシステムである。

AnchorPage はコンテンツ主導のナビゲーションをユーザに提供する。AnchorPage 化されたドキュメントは自分自身にリンクされているだけでなく、ドキュメント内そしてドキュメント間にもリンクを張る。

AnchorPage は意味を持つフレーズと概念を各 WWW ドキュメントから抽出し、それらのフレーズと概念に対してアンカーを自動的に挿入する。そしてドキュメントに関する4つのナビゲーションビューすなわち、コンテンツビュー・テーブル、フレーズビュー、コンセプトビュー、そしてアブストラクトビューを作成する。これらのナビゲーション・ビュー上の各エントリは、オリジナルテキストの発生元にリンクさていれる。

AnchorPage のプロセスは次の3ステップから構成されている。

- ドキュメントのパース。
- アンカーをドキュメントに置く。

[1] AnchorPage は米国 ICONOVEX 社の登録商標である。

- それらのアンカーにリンクされたナビゲーションページの構築。

これらの各ステップは自動化されているが、ユーザが自分のニーズを優先できるように変更できる。ユーザは選択閾値設定を通して、意味をもつ概念やフレーズの構成に関するAnchorPageの決定水準を変更できる。AnchorPageで供給されている辞典ユーティリティの制御リストを造ることによって、特定の用語やフレーズの重みを増減できる。どのナビゲーション・ビューが作られるかを制御でき、ハイパーテキスト機能によってそれらの間のリンクを作成できる。

ドキュメント間のジャンプで概念間の連鎖を表現するために、ある範囲内のオリジナルテキストを循環するようなリンクを作ることもできる。

3.6.1 ナビゲーション・ビュー

ナビゲーション・ビューは、ドキュメントごとに異ったエントリポイントで表現する。コンテンツビュー・テーブルはドキュメントの見出しと副題をアウトライン形式で見せる。アブストラクトビューは意味をもつ概念の要約をオリジナルテキスト上での順序に沿って表示する。コンセプトビューはキーワードをアルファベット順に示す。フレーズビューは概念から抽出されたキーワードと重要なフレーズのABC順にリスティングする。これらのナビゲーション・ビューはプレゼンテーションページにリンクできる。

次の例は、Microsoft Encarta Encyclopediaから取られた冷戦に関する記事に対するAnchorPage作成した各ナビゲーションビューを示している。

- コンセプトビューテーブル：
  - 冷戦
    - 背景の動きと反動と雪解けと凍結
- フレーズビュー：
  - 冷戦
  - 共産主義、東西対立、国内世論、核の力、ベトナム戦争：
  - 米国のかかわり合い
- コンセプトビュー：
  - 冷戦
    - 共産主義：
      共産主義の世界的な脅威に関する認識に根源を発する取り組むの一部として行なわれた戦争
    - 国内の圧力：
      軍拡競争の上昇するコストと他の国内に助けられて、米国とソビエト連邦のリーダを話し合いのテーブルへと導いた。

* 東西対立：
  東西対立は、より伝統的な争い形式を仮定した。
* 条約：
  1987 年 12 月の超大国の間の会談は中距離核削減条約に合意した。
* ベトナム：
  ベトナム戦争における米国の財政悪化により不信が形成された。

- 抽象的 View：
  - 冷戦
    * ベトナム戦争における米国の財政悪化により不信が形成された...
    * 共産主義の世界的な脅威に関する認識に根源を発する取り組むの一部として行なわれた戦争...
    * 東西対立は、より伝統的な争い形式を仮定した...
    * 軍拡競争の上昇するコストと他の国内に助けられて、米国とソビエト連邦のリーダを話し合いのテーブルへと導いた...
    * 1987 年 12 月の超大国の間の会談は中距離核削減条約に合意した...

3.6.2 コンテンツ主導ナビゲーション

Web サイトのブラウズは、

- ドキュメントからドキュメントへトラバースすることによって異ったディレクトリを調べる。
- 特定対象を探すためにサーチエンジンを使用する。

などによって行なわれる。

AnchorPage によってユーザはドキュメントをトラバースするようなコンテンツ主導ナビゲーションが行なえる。ユーザは AnchorPage 化された情報を集約することによって、フレーズや要約で表現される概念をトレースできる。ユーザはドキュメントのアブストラクトビューを使って、ある概念をより深く調べるか否かを素早く評価できる。ユーザが参照しているキーワードが読んでいる文脈にあるか否かをコンセプトビューで確認できる。

コンテンツビュー・テーブルは、ドキュメントのすべての見出しに対する高速なアクセスを提供する。コンテンツ主導ナビゲーションを行なう、4 つのナビゲーションビューの全てがドキュメントの内容を別のエントリポイントで表現している。そして、それらはハイパーテキスト機能によって相互に接続されている。

3.6.3 AnchorPage の特徴と利点

- **処理速度**
  66MHz の PC で 600K のドキュメントを分析した場合 9.5 分であった。この値は

一晩で全ドキュメントを処理できることを意味する。ゆえに、新しいドキュメントは数分で追加される。AnchorPage を使用たサーバでは、余分な管理のオーバヘッド無しに最新情報を追加できる。

- **プロセスの自動化**
  一度、管理者がオプションをセットするとエンドユーザが見るナビゲーションのページ生成するドキュメント分析プロセスは自動化される。

- **ナビゲーションビューのカスタマイズ**
  サイト管理者は 4 つのナビゲーションビューのいずれか、またはそのすべてをユーザに提供できる。それらのビューの詳細化基準を 6 つの異なるレベルでセットできる。

- **異なった関心や優先順位に適合できる**
  LexEdit（AnchorPage で提供されるユーティリティ）によって、管理者は特殊な重要性を考慮しなければならない用語リストによって、ドキュメントの分析を変更できる。

- **ダウンロード時間の短縮**
  AnchorPage によって、ユーザはダウンロード時間を短縮し、より多くの制御を行なうためにセグメント化できる。管理者がドキュメントとユーザのニーズから見て最もふさわしいと思う大きさのセグメントを指定できる。

- **既存のページへの影響**
  AnchorPage はメニューページと実際のドキュメントとの間にシームレスなリンクを張るが、サーバの既存構造や構成に影響を与えない。

- **フォーマット変換**
  AnchorPage は RTF を HTML フォーマットに変換するプログラムを提供している。

3.7 Harvest

Harvest[2]はインターネット上の情報の収集、抽出、構成、検索、キャッシュなどを行なうツールである。ユーザのカスタマイズによって、Harvest はさまざまなフォーマットで表現された情報のダイジェストを作成できる。さらにインターネット上でユーザ独自の検索サービスを提供できる。WWW クライアント（たとえばNCSA Mosaic）のユーザであれば、Harvest サーバにアクセスできる。

[2]Harvest は米国コロラド大学で開発されたシェアウェアである。

3.7.1 Harvest の特徴

1. Harvest は他のプロダクトよりも、ネットワークバンド幅、サーバロードとディスクスペースなどの点で、スケーラブルなアーキテクチャを提供する。既存の情報検索システムと比較するとインデックシングでは 1/4 に、リモートノードの上のインデックスからの情報報抽出では 1/6,600 にサーバロードを減らせる。そしてネットワークトラフィックは 1/59 に、インデックシングに必要なスペースは 1/43 になる。

2. Harvest は、構造化クエリを行なうために（すなわち、ドキュメントの著者やタイトルだけにキーワードをマッチングさせる）アブストラクトオブジェクト交換フォーマット（SOIF）と呼ばれる構造化されたインデックスフォーマットを定義している。このフォーマットは非常に効率的な検索プロトコルであり、フィールド内に任意のデータを保持できるオブジェクトサマリーストリームを使用しているので、インターネットにおけるの Anonymous FTP Archives IETF Working Group（IAFA）のフォーマットよりも強力である。SOIF が任意のデータ型をサポートしていることは、画像や音声のような複雑な検索でも使用できることを意味している。

3. Harvest は構造化された高品質のインデックス情報に対して、自動インデックシングを行なう。WHOIS++は、IAFA テンプレートを満足するようなインデックシングをサイト管理者が手作業で行なう事にしている。一方、GILS はインデックスデータがどのように収集されるか定義しない。Harvest はこれらのシステムにインデックスグデータを提供できる。

4. FreeWAIS は構造化フィールドに対してのみ、AND/OR 検索が行なえる。Harvest は AND/OR 問合せ、近接サーチ、正規表現、ケース依存（あるいは非依存）検索、ワードの部分マッチ、全体語あるいは複数語、可変粒度での結果表示などの機能をもつ。

5. Harvest はオリジナルな WAIS、コマーシャルベースの WAIS、freeWAIS、Glimpse、Nebula などにサーチエンジンにプラグ-インできるインタフェースを提供している。ゆえに、Harvest は各エンジンのもつ長所をユーザが享受できるようにする。現在、他のサーチエンジンと Harvest との統合化作業が行なわれている。

6. FreeWAIS そして他のインデックス索引作ツールは個別のインデックス生成を行なう。これらのシステムは固定化された内容抽出方法で特有のデータフォーマットを処理する。それに対して Harvest では sed のような UNIX 標準プログラムや正規表現によって抽出されたインデックス情報は何でもカスタマイズできる。それに加え、Harvest はデフォルトのサマライザーを提供している。これにより 3 ～11 パーセントのディスク容量で、精度と再現性の点で WAIS に匹敵する性能が得られる。一方、Harvest はフルテキスト・インデックシングもサポートしている。

7. 全体的にみてHarvestの表現力は、他の分散インデックシングシステムを包含している。以下のリストはHarvestがどのように他のインデックシングシステムを実現しているかを示している。

   - Archie, Veronica, WWWW, etc.:
     Gatherer コンフィギュレーションとエッセンス抽出スクリプト
   - Content Router:
     WAIS エミュレータとエッセンス抽出スクリプト
   - WAIS:
     フルテキストのエッセンス抽出とランク付け可能なサーチエンジン
   - WebCrawler:
     Gatherer コンフィギュレーションとフルテキストのエッセンス抽出
   - WHOIS++:
     Gatherer コンフィグレーションとエッセンス抽出スクリプト

3.7.2 Harvestの今後

Harvestに対して以下のような拡張が計画されている。

- いくつかの商用の検索/情報検索システム、そしていくつかの商用DBMSの統合的サポート。これはHarvestの標準化作業にも結びついている。
- Harvestにおける分散サーチのサポート：
  query routing mesh、parallel query tool、taxonomy/annotation tool、improved quality control of Broker meta data。
- Object APIの拡張とSOIFフォーマットの拡張
  - より複雑で構造化されたデータのサポート（すなわち、関係またはオブジェクト指向）。
  - ILUコンパチブルな“tap”のSOIFストリームへの追加。すなわち、Harvest収集したインデックスデータをCORBAコンパチブルな外部データへ追加する機能。
- 検索処理におけるスケーラブルな分散クエリーが可能なローカルキャッシュ、ユーザがカスタマイズ可能な類似性フィルタリングなどが可能な検索結果に対するローカルな後処理のサポート。
- ナショナルワイドのヒエラルキーを作るための領域キャッシュ。
- “メタプロトコル記述言語”や領域特殊な言語や翻訳系に基づいた、構造化されたデータソースへの拡張可能なインタフェースの提供。これにより、構造化された作成者/利用者のスキーム間の適切なマッチング制約、Z39.50のような他の検索インタフェースのサポートなどの互換性を追加できる。

- アタッチドメソッド、ボキャブラリ、プロファイルなどを使用したカスタマイズのサポート。
- 非テキストインデックス/サーチエンジンの統合。すなわち、音声/イメージ検索のサポート。
- ユーザの情報ベースの成長に対応できるレプリカブローカの開発。

3.8 “浅い” 構造化

情報の有用性は状況に依存する。データはユーザーのおかれている状況と相互作用しながら情報となる。

このような情報の構造化を検索という観点からを考えると、以下のような問題がある。

- 状況依存の問題
  ユーザが必要な情報はユーザのおかれている状況に依存する。またユーザの要求はその要求が依存する領域特殊性にも依存している。
- 構造化則の変化
  情報が依存する領域自体が変化する。つまり、情報の領域における地位も変化する。

すべての状況を予測し、その構造を陽に表現することは不能である。

このように動的に変化する情報構造に対して、

- 情報の自己組織化
- 試行錯誤的な情報検索支援

といった立場の異なった二方向からのアプローチがある。

構造化情報の自己組織化については機械学習の枠組みで行なわれている研究が多い。この文脈における研究の課題は、

- 文献におけるヒューリスティックの適用範囲の拡大。それらによる帰納推論能力の向上
- 対象データモデルの拡大
- 帰納推論によるシソーラス、インデックスといった構造化知識の自己組織化

などである。これらのアプローチの問題点は、完全な意味処理が難しい点である。

一方、試行錯誤的な情報検索、それによるアドホックな構造化を支援するシステムでは、以下の課題がある。

- 自由語による検索

- 精度と速度とのトレードオフ

これまでに研究が行なわれてきた情報検索の知能化では、以下のようなアプローチがとられて来た。

- 主題–検索情報間の自動マッピング。
- 検索情報の自動生成、管理の自動化。

このアプローチに基づいたシステムで実用的なシステムとして利用されているシステムは少ない。それらのシステムでは、“深い” 構造化すなわち精緻な意味解析を行なうことに焦点があてられて来た。情報は本質的には状況依存性を持つものであり、各状況に対応可能な意味処理は困難である。

意味処理機能の知能化に関する研究やシステム開発は今後とも続けて行く必要性はあるが、一旦この問題の原点に戻って見る必要もある。情報検索を経験したことが無いユーザが、この分野のエキスパートにアドバイスを求める時、どのようなアドバイスを受けるのであろう？エキスパートが使う方略として、想定できる以下のような方略がある。

- 主題ごとに著者、タイトル、キーワードなどの二次情報を使い分ける。
- 情報源ごとに著者、タイトル、キーワードなどの二次情報を使い分ける。

ある主題ごとにどの情報源を使うか、また著者、タイトル、キーワードなどのいずれから検索したら良いかを決定するモジュールを組み込む。つまり、このアプローチはエキスパートが持つヒューリスティックを、より陽な形で利用しようするアプローチである。

これまでのアプローチは速度よりも精度を優先してきたと言える。この優先順位を逆転させることにより新たな検索処理の高機能化できると予想できる。検索精度の低下は検索→確認のサイクルを早くすることによってカバー可能であり、長い試行錯誤過程におけるエンドユーザのフラストレーションを低減できよう。また、検索→確認のサイクルのスピードアップすることにより、情報の状況依存性にも貢献できるとも予想出来る。

今後、このような精度よりも速度を優先した意味処理すなわち “浅い” 意味処理に関する研究を行なっていく必要がある。このアプローチによるシステムは、ゼロからの構造化を狙うのではなく文書の持つ構造を積極的に利用して自己組織化を行なう、意味構造の抽出よりも情報縮約を優先する、などを特徴としている。

4 KDDによる構造化情報発見

4.1 KDDの研究動向

今日、さまざまな分野においてデータベースを応用に対するニーズが高まっている。その反面、多くの分野でデータベース上のデータを分析してデータの背後にある知識を発見するエキスパートが不足している。このことから、データベースから自動的に知識を発見するシステムの役割が重要になっている。

近年、データベースからの知識発見（KDD：Knowledge Discovery in Databases）は人工知能研究における重要なトピックのひとつとなっている[43]。KDDとは、データベース上の"生"データや一見したところ無秩序に見えるデータから、ルール、属性間の因果関係などの規則性といった有用な知識を見いだし、人間の発想、発明、設計、意志決定、予測推定などの知的活動の支援に関する研究である。

一般的に言えば、発見とはある未知の事実に遭遇した人間がその事実を説明する仮説を作り、検証する思考過程である。言い替えれば、発見とは教師なしで人間や機械が知識を獲得する学習形態である。科学的発見の手段は理論駆動型とデータ駆動型に分類できる。科学技術における発見はデータ駆動型のほうが多い。例えば、天文学でのケプラーの第三法則や物理学でのボイルの法則などの発見などがそれにあたたる。人工知能においても発見や発明を行なうシステム（特に自律性が高く、汎用性があるもの）を作ることが人工知能研究の発端からの研究者の夢であり、最も難しい問題でもある。KDDの問題は発見の問題であるので、この研究の重要性や難しさは十分予想できる。

データベースは重要な知識源であり、知識獲得に応用されるべきという認識が高まっている。KDDはエキスパートシステム、知的データベース、知識獲得、機械学習、事例ベース推論、統計学などの分野と関連がある。従って、KDDは伝統的な機械学習の研究とは異なる側面もあるが、機械学習の研究結果を利用も行なう。

4.1.1 研究の手法と目的

KDD研究の手法と目的には、トップダウン的アプローチとボトムアップ的アプローチがある[64]。

1. トップダウン的アプローチ
   トップダウン的アプローチには以下の三つの技術的側面がある。

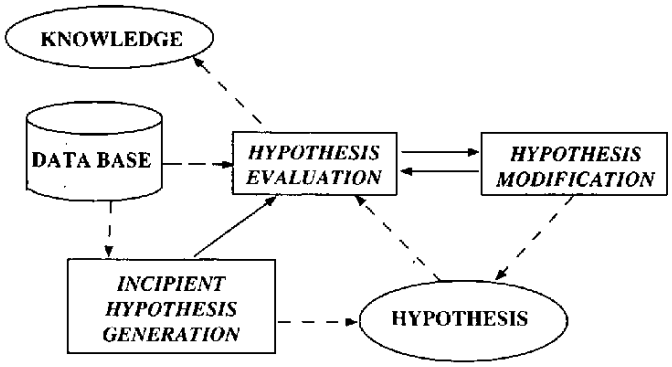

Figure 4.1: データベースからの知識発見プロセス

- **データベースからの知識抽出**
  データベース上の“生”データや一見したところ無秩序に見えるデータから有用な知識（ルール、属性間の因果関係などの規則性）を発見する。言い替えれば、データベースに格納された概念の事例を構造化することによって、初期概念や仮説を獲得し、さらにルールとして表現する。
- **獲得された知識に対する管理と精緻化**
  多重世界とマルチレベル機構によって獲得された初期概念や仮説を表すルールを管理し、さらにメタ推論に基づく高次推論などによって精緻化する。
- **データベースの動的な構成**
  推論などによって動的に発見された属性や概念化されたデータを格納するデータベースを動的に作成し、さらにフラットなデータベースから構造を持つデータベースへ自動変換する。

データベースを作成する際、データベースの属性間にどのような因果関係（たとえば、関数関係、依存関係など）があるかは必ずしも明確ではなく、データベース化した後にそれを処理するのが普通である。従って、データベースの背後にある因果関係を発見するために、仮説を作りそれを検定する手段が必要となる。大須賀は図 4.1の一般問題解決モデルの KDD への応用を提案した [40]。因果関係の発見を行なう際に新しい属性が生成する場合もある。この属性は何らかの意味を持つので、これを記録し再利用するためにデータベースの動的な再編成の機能も必要である。

2. **ボトムアップ的アプローチ**
   新しい知的応用システムの構築をボトムアップ的に見ると、以下のような問題点がある。

   - 知識ベースとデータベースまたは機械学習と知的データ分析のメカニズムに対して何が要求されているか。

- また機械学習と知的データ分析のメカニズムを実現するという立場からみれば、知識ベースとデータベースの構造と機能に対して何が必要であるか。
- それをどのように実現したらよいか (図4.2を参照)。

これらの問題を次の三つの側面から考察する。

- ルールおよびデータに関する要求：
  (a) 固定的なルールとデータだけでなく、状況の変化に追随でき、動的に生成されるルールとデータを扱えること。
  (b) 定量的なルールとデータに基づく結論を出すだけでなく、定性的なルールとデータを扱えること。
  (c) 明確なルールとデータに基づく結論を出すだけでなく、不明確/非線形なルールやデータを処理できること。
- 知識ベースとデータベース機構への要求：
  (a) 知識ベースとデータベースとを統合的に利用できること。
  (b) 知識ベースとデータベースとが複雑な構造を表現できること。
  (c) 知識ベースとデータベースとが自己組織化、動的な特性を持つこと。
  (d) 知識ベースはマルチメタレベル推論と多重世界管理が行なえ、データベースは自動分割、概念抽象化が行なえること。
  (e) 集中単一型処理だけでなく、協調分散処理が行なえること。
- 機械学習と知的データ分析の手法に関する要求：
  (a) 多様なデータを格納している大規模データベースから知識を抽出できること。
  (b) 演繹推論と統計推論、帰納推論、類推などの統合利用できること。
  (c) いろいろな学習の方法が利用でき、汎用性があること。
  (d) 教師あり学習と教師なし学習の両方が可能であること。

また、データベースから知識を迅速かつ正確に発見するために、次の二つの側面を重視する。

1. **機械学習や知識発見の対象となるデータベースの特性**
   データベースは他の学習の対象とは異なり、四つの特性 (複雑性と大規模性、動的性、不完全性と曖昧性、データの多様性) を持つ。従って、データベースから知識を発見するために、一つの学習/発見方法を利用するだけでなく、多面的データ分析、多段階学習、概念抽象、ノイズデータに対する処理などができる多様な学習機構を集めたツールボックスを作成する必要がある。

2. **知識ベースとデータベースの統合利用の環境**
   機械学習や知識発見では知識ベースとデータベースの統合環境が必要である。す

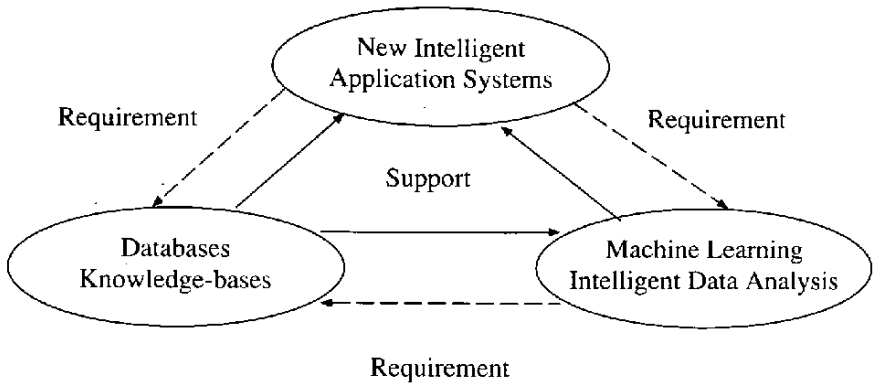

Figure 4.2: 3つの問題間の関係

なわち、データベースは重要な知識源であり知識獲得に適用される。一方、KDDには知識ベースの支援が必要であり、発見された知識は知識ベースに追加される。また、発見された知識と知識ベース上の既存知識を組み合わせることによって、利用、管理、精緻化が可能になる。これまでの機械学習研究では知識ベースとデータベースの統合環境上での研究は少ない [32, 51]。

KDDの研究はエキスパートシステム、知的データベース、知識獲得、機械学習、事例ベース推論、統計などと関連がある。従って、知識ベースとデータベースの統合環境上でKDDを行なうためには、以上述べた機能と問題、データベースの特性をなどを良く検討し、新しい発見方法論とその方法論に基づいたメカニズムを開発しなければならない。

4.1.2 これまでの研究

Piatetsky-Shapiroは、関係データベースのタプルにおいて成立する従属性に注目してルールを求めるタプル指向のアルゴリズムを提案した [44]。このアルゴリズムは、関係データベースにおける関数従属性の研究との関連性が深い。この研究の発展として、Piatetsky-Shapiroはタプル指向のアルゴリズムを使ってサンプルから導出されたルールが全データベースに適用された時の統計的正確さに関する解析を行っている。

Hanは属性間の概念的階層構造に注目してルールを求める属性指向のアルゴリズムを提案した [20]。このアルゴリズムは、学習化に関する知識を概念木として与え、属性ごとにその概念木を葉からルートに辿りながら、各属性の値をより一般的な概念をもつ値に置き換え、その結果成立するルールを導出する方法である。

Zhongの研究では知識ベースとデータベースの統合環境で自動的にデータベースから知識を発見する問題について研究を行い、知識ベースシステム開発ツール-KAUS上にKDDの方法論およびシステム-GLS（Global Learning Scheme）を構築している。また、この開発結果もKAUSの機能の追加と拡張になる（図4.3の2を参照）。

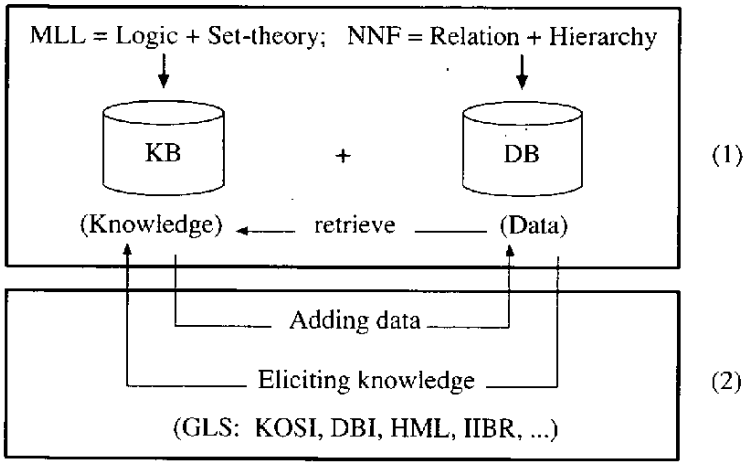

Figure 4.3: GLS と KAUS

図 4.3の 1 に示すように、現時点の KAUS システムは多層論理に基づいて知識を表現する知識ベースと非正規関係モデルに基づいてデータを表現するデータベースを持ち、マルチメタレベル推論と多重知識世界の表現が行なえる。また、動的に知識世界とレベルを生成でき、知識とデータの管理や変換などに有効な手段を提供している [41, 63]。

4.2 GLS 発見方法論

GLS 発見方法論は前章で示したような KDD の特性や問題などを考慮して開発した方法論である。また、大須賀の提案した一般問題解決モデルを KDD へ応用したものとも言える。

GLS の核はデータベースからのグローバルな学習スキーマ（Global Learning Scheme）である。知識ベースとデータベースの統合利用に関する研究における汎用化や知識発見と知識利用の統合に対する要求に基づいた、データベースからのグローバルな学習スキーマ（GLS）は図 4.4のように表される。GLS はプリプロセス、知識抽出、知識の精緻化/管理という三つのフェーズに分けられる。

1. プリプロセスは知識抽出の準備フェーズである。このフェーズの主要なタスクはユーザとの対話によってユーザ要求を収集し、使用したい学習方法（知識指向統計推論法や分割に基づく帰納法など）、データベース名と属性名のなどを確定し、データの収集と整理などを行う。たとえば KOSI 法を選択した場合、このフェーズでは利用すべき新しい属性を発見するための属性演算やクラスタリング、利用すべきクラスターデータを選択するための逐次判別分析などを行なう。また、属性演算により発見された新しい属性を記録するためのデータベースを動的に生成することもできる。一方、DBI 法を選択した場合、このフェーズでは属性に基づくクラスタリングと概念抽象やデータベースの確率空間の生成などを行う。

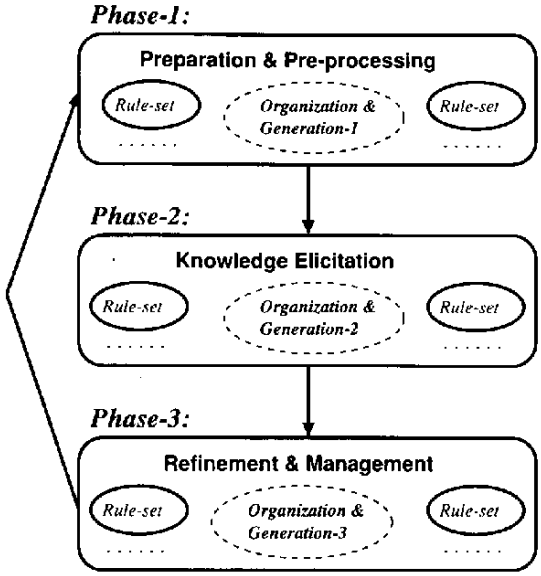

Figure 4.4: Global Learning Scheme

2. 知識抽出のフェーズではプリプロセスフェーズの結果を基に知識抽出を行う。たとえばKOSI法を使用する場合、データに隠れている最良の構造特性を発見するために、前フェーズの属性演算で発見した新しい属性を回帰分析などの統計手法によって評価し、次に継承推論に基づく精緻化法（IIBR）により特性知識を抽出する。一方、DBI法を使用する場合は、まず教師ありまたは教師なしのデータベースの分割法により概念クラスターを形成し、次に階層モデル学習法（HML）により階層モデルを持つ分類知識を抽出する。

3. 知識の精緻化/管理フェーズでは前フェーズで発見された初期概念や仮説をルールとして表現して知識ベースに追加し、さらに精緻化や管理を行う。すなわち、データベースから初期概念や仮説を発見し、それを知識ベース管理システムによって管理すると共にメタ推論と多重世界の機能を用いた知識の精錬手法と組み合わせて、それを精緻化する。それらを行なうために、階層モデル学習法（HML）や継承推論に基づく精緻化法（IIBR）などが用いられる。また、利用フェーズで知識が不足したり新データから知識を抽出したい場合はプリプロセスフェーズに戻ることができる。

GLSではこの三つの学習フェーズによって、知識ベースとデータベースとを統合した環境でKDDを行なうための多面的データ分析および多段階学習や概念抽象化が可能になる。

一方、GLSでの発見/学習とは、ユーザの要求によるKDDだけではなく動的な発見プロセスの組織化や発見プロセスの制御と性能改善なども含んでいる。知識ベースを用いて処理が行われたGLSシステムでは、すべての知識が図4.5のように多重世界のメカニズムによってマルチレベルに分けられて管理される。

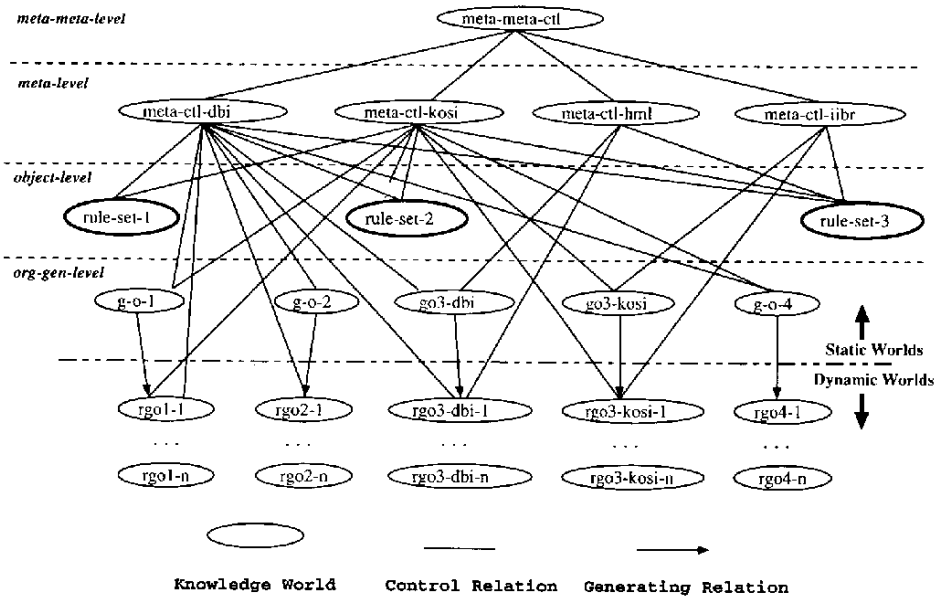

Figure 4.5: GLS における多重世界の階層構造

知識世界を大きく分ければ静的世界と動的世界に分けられ、GLS の各学習フェーズで動的な知識の組織化/生成が必要となる。組織化は発見の目的に応じて動的に発見プロセスを構成し、知識抽出、精緻化/管理、利用などの統合のために必要になる。つまり、発見の目的に応じて資源の有効利用や多段階で連続/並列処理できる発見プロセスを動的に構成するために、GLS には自己組織化機能が必要である。

一方、GLS の知識ベースが必ずしも完全ではなく、前もって与えられていない知識がある。たとえば、もし知識発見に利用すべきデータを収集するためのデータベース検索やデータ整理に関するルールなどが不足していたら、まずこれを生成しなければならない。このようなルールはユーザとの対話によって、あるいは推論と学習によって動的生成を行ないながら、推論する必要がある。また、データベースから知識を発見する前には、知識の形式や精緻化/管理方法は分からない。獲得された知識と GLS の知識ベースにある既存の知識を融合して、精緻化/管理したり利用するためにも生成/組織化機能が必要である。

4.3 GLS 発見システム

GLS 方法論に基づいた GLS システムは、所与の機能を有する幾つかのサブシステムを用意し、それらサブシステムが統合的に利用できるようにすることである。GLS では二つの発見サブシステム – 分割に基づく帰納法（DBI:Decomposition Based Induction）と知識指向統計推論法（KOSI:Knowledge Oriented Statistic Inference）を利用してデータベースから初期仮説を発見できる。さらに、その二つの発見サブシステムにつながる

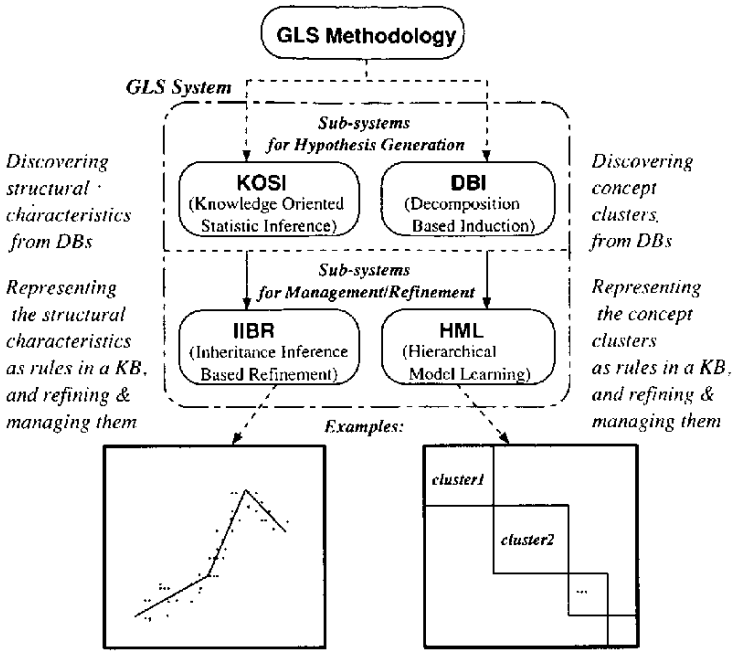

Figure 4.6: GLS システムの概要

精緻化/管理サブシステム – 階層モデル学習法（HML：Hierarchical Model Learning）と継承推論に基づく精緻化法（IIBR ：Inheritance Inference Based Refinement）を利用して、初期仮説を精緻化/管理できる [70, 74]。つまり、GLS システムは多戦略発見システムである。図 4.6に GLS システムの機能を概念的にまとめている。

4.3.1 分割に基づく帰納法 – DBI

DBI 法はデータベースの属性間の依存関係に注目し、分割に基づく帰納によりデータベースから概念クラスターを発見するものである [65, 69]。

DBI 法は Simon-Ando らの近似完全分割法を基に開発した方法である [53]。近似完全分割法に以下の幾つかの拡張と変形を行なうことにより、データベースの近似分割法が得られる。

1. 教師あり学習としてのクラスに基づく対角化。
2. 教師なし学習としての最尤分割による対角化。
3. ノイズ（出現の確率値が小さいデータや矛盾するデータ）を削除しながらの分割。

データベースの近似分割法において最も重要な特性は、ノイズデータを分析/削除しながら概念クラスターまたはサブデータベースを形成することである。データベースを分割するために、DBI 法では事例空間、確率空間、学習空間という三つのデータベースの空間を定義している。

事例空間とは対象問題に関する事例データを記述する空間である。確率空間とは対象問題に関する事例データの確率分布を記述する空間である、学習空間とは事例空間と確率空間の変化と学習状態を記述する空間である。特に確率空間はデータベースの分割時に使われる。すなわち、事例空間から生成された確率空間を表現する確率分布行列に対して、分析/評価/近似的分割によって、幾つかの概念クラスターやサブデータベースが求められる。

伝統的なデータベースと異なり、この三つのデータベース空間により対象問題に関する事例データだけでなく、特性分析、概念説明、事例間の関係、事例の変化状態などの事例データに関する情報を得ることができる。

また、DBI 法を実用化するために以下の三つの補助的な技術がある。

1. 一つは属性に基づくクラスタリング（概念抽象化、連続値の離散化など）である。これは確率空間を生成するための前処理の一つである。これによって、それぞれの属性値に対して概念抽象化、連続値の離散化などが行なわれる。また、領域知識を使うかどうかによって、領域知識に基づくクラスタリングと連続値の範囲の分割による離散化という二つの方法がある。

2. 第二は学習空間による知識の精緻化である。学習空間によって、確率空間に対する制御、説明、学習のための確率分布と誤差を記録し、データベースの摂動問題を処理する。学習空間によって、分割された結果はデータの小さい変化に影響されないようになる。すなわち、データの変化が許容される最大誤差以内であれば、学習空間に記録するだけで分割されたデータベースは変わらない。分割されたデータベースに対して、データの変化は誤差の増加と減少両方の可能性があるため、これは合理的な方法である。従って、学習空間で誤差を記録して、データの変化よる誤差の蓄積が誤差制限上界を越える場合は分割されたデータベースを修正する。

3. 第三は確率分布行列の生成における学習および性能改善である。確率分布行列は確率空間の実体であり、DBI 法の主な操作対象である。従って、確率分布行列の性能が DBI 法による発見された知識の良さと発見過程の計算量などに対する重要な鍵となる。データベースの複雑性と大規模性などを考え、計算量の爆発などを防ぐために、確率空間を生成する時に、知識ベースの支援によって確率空間の性能を改善することが必要である。また、データベースの属性は互いに無関係でなく、ある領域知識に基づいて属性間にある関係が成り立つ可能性が大きい（たとえば、関数関係、等価関係、類似関係など）。従って、属性間に幾つかの明確な関係が存在する場合、その中の一つを選択してそれ以外の属性との間の確率分布行列を生成する。そして、確率分布行列を生成する時に利用されなかった属性も、確率分布行列を用いた分割の結果と知識ベースを利用して推論によって導出できる。

4.3.2 知識指向統計推論法–KOSI

KOSI 法はデータベースの属性間の関数関係に注目して、AI 手法と統計推論を統合してデータベース上のデータに隠れている構造特性を発見するものである [67, 68, 72]。

KOSI 法の重要な背景に BACON、FAHRENHEIT、ABACUS などの発見システムがある。ただし、KOSI 法では次のような拡張がなされている。

1. 不完全性と曖昧性を含むデータを処理できること。
2. 特定の分野だけでなく、一般的な属性間の関数関係を発見できること。
3. 精密な関数関係を発見するだけでなく、近似的な関数関係も発見できること。

以上の拡張のために、KOSI 法では "生成/評価" 過程を使用している。まず、生成のフェーズではモデルベースおよびメタレベルの制御を基に、発見的知識および問題領域に依存する領域知識を用いて新しい属性を見つける。すなわち、それらの知識を用いて探索空間を制限することにより定性/定量的属性演算を行ない、さらには属性間の関数関係を発見し、新しい仮定属性やデータベースを動的に生成する。このフェーズで用いられる発見的知識は以下の二種類である。

1. BACON などで用いられた発見的知識を拡張した知識。
2. 定性推論を基に開発した知識。

この際、データベースの大規模性に対応するために、区間演算やクラスタリングの機能が提供されている。評価のフェーズでは以下の二つの機能がある。

1. 生成フェーズで幾つか仮定的な関数関係を発見した場合の統計的手法。たとえば、回帰分析などを用いてそれを評価することにより、最も良い関数関係を選択する。
2. 生成フェーズで作成された新しい属性を含んだデータベースにおける属性間の近似的な関数関係を、統計的手法により再発見する。

このフェーズではメタ知識を用いて統計推論を制御する。すなわち、ルールで表現された知識を用いて統計の戦略と方法を選択し、統計処理の過程を制御する。

KOSI 法では主に次の二種類の統計手法を使用している。

1. 標本データの背後にある構造特性を発見し、最良モデルを選択する AIC 基準と H 変換に基づいた回帰分析。
2. 標本データの背後にある構造変化点を発見し、クラスタリングする逐次 Chow 検定とクラスタリング。

4.3.3 階層モデル学習法–HML

一般的に言えば、データベース中のデータは不完全であり曖昧性があるので、データベースから発見された知識は仮説と呼ばれる。また、データの追加/修正/削除によって仮説も変化する。従って、データベースからの知識発見の過程は仮説の構築と精緻化すなわち評価と修正の過程とも言える。発見された初期仮説に対して、どのように精緻化と管理をするのが良いのかということも重要な問題となる。HML 法はこのための一方法である [66, 75]。

HML 法を開発する重要な背景として多層論理や情報理論がある。HML 法の開発の基礎は多層論理によるモデル表現である。すなわち、情報理論（エントロピー）を用いて多層論理式の持つ情報量を定量的に評価する。この評価の過程を効率化するために、HML の理論的な基礎となる多層論理式を持つ情報量に関する三つの定理、すなわち多層論理事後濃度の補足定理、多層論理情報量の等価性に関する定理、多層論理の包含に関する定理が開発された。さらに、この三つの定理を用いて効率的なアルゴリズムが実現された。

DBI 法につながる HML 法は次の三つの機能を持つ。

1. **知識生成**
   DBI 法により発見された概念クラスターを自動的に階層構造化し、多層論理式によって表現して、階層モデルを持つ分類知識として知識ベースに加える。

2. **知識の精緻化**
   データベースにあるデータの変化または領域知識を基に、多層論理式によって表現されてた階層モデルを持つ分類知識の情報量によって、自動的に“よい”階層モデルを選択して精緻化する。

3. **知識の管理**
   メタレベル制御をにより、まずデータベースから発見された階層モデル族が ISA 階層を表す集合チェーンに保存する。次に評価と精緻化のメカニズムを用いて精緻化し、最後に階層モデル族の継承グラフを用いて階層モデルの履歴を管理する。

4.3.4 継承推論に基づく精緻化法–IIBR

HML 法と対応するものとして KOSI 法とつながる精緻化と管理のための IIBR 法が開発された [71, 73, 76]。IIBR 法を開発する重要な背景として KAUS のメタ推論と多重世界の機能や回帰モデルの継承推論がある。KOSI 法ではデータベースから発見された構造特性を表す回帰モデルは特性知識の中核となるので、特性知識間の継承関係は特性知識の構造特性を表す回帰モデルの継承関係となる。

継承推論は人工知能における重要な推論の一つである。回帰モデルへの応用では、下方への継承、上向きの継承、横の方への継承、対角の継承、類似の継承などの幾つかの継承関係が考えられる。一方、これらの継承関係の強さを定量的に評価するために、これらの継承関係の数量化は重要である。この数量化方法の一つは誤差分析である。多

層論理の展開およびメタ推論と多重世界の機能を基に、この誤差分析を用いてデータベースから発見された特性知識の継承関係を調べられる。

IIBR 法では KAUS のメタ推論と多重世界の機能を用いた精緻化の手法を用いて、特性知識の継承関係の管理を行う。IIBR 法は次の四つの機能を持つ。

1. 知識ベースに知識を生成する。すなわち、KOSI 法で発見した構造特性を特性知識として知識ベースに追加する。

2. 回帰モデルの変化量を推定する。すなわち、データベース上のデータが変化した場合、メタ知識と領域知識および数量化の誤差分析法によって回帰モデルの変化量を推定する。

3. 回帰モデル族を管理する。すなわち、メタレベルの制御を基に生成された特性知識を集合チェーンに保存し、評価と精緻化のメカニズムを用いて精緻化し、さらに回帰モデル族の継承グラフを用いて特性知識の履歴を管理する。

4. 回帰モデル族から利用すべきモデルの選択。すなわち、ユーザの要求に応じてメタ知識と領域知識、逐次判別分析法を利用して回帰モデル族から利用すべきモデルを選択する。

4.4 文献情報と KDD

以上述べた GLS 発見方法およびシステムは、主に関係データベースから知識発見を目的としたものである。本節では、前節で述べた GLS という発見方法論およびシステムをどのように拡張してテキストデータベースの構造化にするかを述べる。

一般的に言えば、関係データベースから知識を発見するために、属性間の依存関係の分析による構造化は最も基本的な方法である。ところが、テキストデータベースは関係データベースと違って、シーケンシャルな文字列であるので、構造化の目的と手法がかなり違う。

4.4.1 構造化の目的

ここではテキストデータベースの構造化の目的は以下のように幾つか列挙することができる。

- **自動インデックシングと自動抄録**
  テキストを分析してキーワードを見つけ、文献のインデックスや抄録などを自動生成する。つまり、文献検索や情報圧縮などのために、一次情報とする原テキストから二次情報を抽出することである。

- **文献分類**
  テキストデータベースに保存する各テキストを分析し、文献間の関連性によって文献を分類する。

- **情報の組織化**
  テキストにおける専門用語間の上位、下位、同位関係を明確化し、概念木などによって表現する概念の体系を生成する。

- **関係データベースから知識発見のための領域知識の提供**
  テキストデータベースの構造化の結果は文献検索の精度と速度の向上に使える他に、関係データベースから知識発見のための領域知識としても利用できる。つまり、概念の体系は用語の体系で表現される。これは、関係データベースから知識発見のための領域知識として利用する。

4.4.2 構造化の方法の分類

構造化の方法については次の2種類に分類できる。

- **テキストデータベースのユーザ要求による構造化**
  ユーザの要求やデータの意味を重視するために、シソーラスを領域知識として利用し、ユーザとインタラクションしながら構造化を進める。

- **文献情報の固有価値による構造化**
  データの意味を考慮せず、数学的式や発見的知識などの客観的基準を利用して構造化を進める。

4.4.3 構造化結果の表現形式

テキストデータベースの構造化結果は以下の形式で表現できる。

- **概念階層**
  テキストにおける専門用語間の上位、下位、同位関係を表す概念木である。

- **概念ネットワーク**
  テキストにおける専門用語間の相関性や依存関係を表すネットワークである。

- **IF-THEN ルール**
  テキストの分類やテキストにおける専門用語間の依存関係を表し、文献検索や推論などに使われる。

- **表**
  シーケンシャルなテキストから単語や数値情報などを抽出し、関係データベースに変換する。

4.4.4 構造化の制御

1. **教師ありの制御**
   ユーザが教師として構造化過程を制御することによって、構造化を進める。

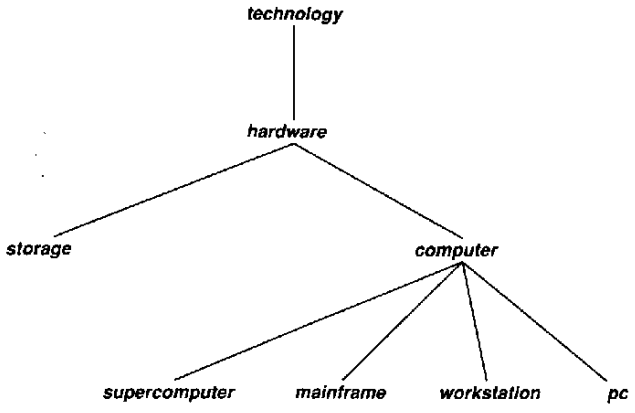

Figure 4.7: 概念階層の一例

2. **領域知識としてのシソーラスの使用**
   電子化された専門用語のシソーラスを領域知識として利用し、構造化の過程を制御する。

4.5 概念階層と概念分布

概念階層と概念分布はデータに対して有力な要約/ブラウズ手段である。概念階層は定性的に概念間の階層関係を表現し、概念分布は定量的に概念の分布を表す。

4.5.1 概念分布

テキストデータベースから概念階層を作るために、シソーラスを領域知識として利用できる。一方、生成された概念階層を基に概念の分布を調べることができる [13]。

1. **ある概念のサーブ概念の分布**
   テキストデータベースから生成された概念階層に一つの概念ノードを設定し、このノードの子ノードの分布を調べる。たとえば、図 4.7に示す概念階層における概念 "computer" のサーブ概念の分布は次のようになる。

$$p(C = \text{“}supercomputer\text{”}) = 0.05$$
$$p(C = \text{“}mainframe\text{”}) = 0.1$$
$$p(C = \text{“}workstation\text{”}) = 0.3$$
$$p(C = \text{“}pc\text{”}) = 0.55.$$

2. **Joint 概念分布**
   幾つかの概念ノードの Joint 分布を調べる。Joint 概念分布は複数の概念の共発生

を表現しているが、出現順序は表現していない。たとえば、"company" と "computers" という概念の Joint 分布は次のようになる。

$$p(C1 = IBM, C2 = mainframe) = 0.07$$
$$p(C1 = Digital, C2 = mainframe) = 0.03$$
$$p(C1 = IBM, C2 = workstations) = 0.2$$
$$p(C1 = Digital, C2 = workstations) = 0.2$$
$$p(C1 = IBM, C2 = PCs) = 0.4$$
$$p(C1 = Digital, C2 = PCs) = 0.1$$

3. **条件付き概念分布**
   ある条件に基づいて概念の分布を調べる。たとえば、"announcement" に関する文献における単語 "computer" の分布は $p(C = computer|announcement)$ と表現できる。

4.5.2 分布の比較

概念の分布を比較するために K–L 情報量が使われる。

$$D(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = E_p \log \frac{p(x)}{q(x)} \tag{4.1}$$

ここで $p(x)$ は真の離散分布であり、$q(x)$ は離散分布モデルである。

一般的には、K–L 情報量は真の分布とモデルの近さを測る客観的な基準として使えるが、逆に真の分布とモデルの遠さを測るためにも使用できる。

1. **一様分布**
   $P(C = c|X)$ であるノード Xの子ノード x_1の分布と一様分布の遠さを測る。たとえば、Ford 社の活動状況と一様分布の遠さを測ることにより Ford 社の活動がどのくらい頻繁であるかがわかる。また、ある会社の製品 x_iに関する単語の分布と一様分布の遠さを測ることによって、この製品を製造する会社の規模などがわかる。

2. **Sibling 分布**
   $Avg\ P(C = c|X)$ を期待 Sibling 分布と呼ぶ。概念ノード Xのある子ノード x_1の分布と $Avg\ P(C = c|X)$ の遠さを測る。たとえば、$p(C = activity|Ford)$ と $Avg\ p(C = activity|X)$ の遠さを測ることによって、Ford 社と他の車製造会社の活動状況の差が分かる。

3. **Past 分布**
   過去のデータから抽出された概念分布と現在のデータから抽出された概念分布の遠さを測ることによって、傾向分析/予測できる。たとえば、Ford 社の過去のデータから抽出された概念分布と現在のデータから抽出された概念分布の遠さを測ることによって、Ford 社の活動状況が変化が分かる。

4.6 テキスト分類と関係学習

テキスト分類のために、一階帰納学習の方法を利用することができる [9]。一階帰納学習の方法とは、一階述語論理に基づいた教師あり帰納論理プログラミグ法である [36]。この方法の特徴は、一階述語論理を用いて単語の順序関係を指定できることである。テキストを分類する時、学習例としてのテキストはキーワードを属性に、そして関連性の有無をクラスに対応づけて学習システムに与えられる。

4.6.1 基本学習法

基本学習法は次の幾つかのステップに分けることができる。

1. **ユーザによってテキストの予分類を指定する**
   たとえば、ユーザは "Machining Learning Session" に関する論文であるかどうかによって、IJCAI で発表された論文のタイトルにクラスを表すラベルを付けるられる。つまり、"+" のクラスのラベルを付けたテキストは "Machining Learning" に関連したものである。一方、"-" のクラスのラベルを付けたテキストは "Machining Learning" に関連しないものである。

d1	+	improving efficiency by learning intermediate
d2	+	learning dnf by decision trees
d3	+	constructivc induction on decision trees
d4	-	a comparison of atms and csp techniques
d5	-	the specialization and transformation of constructive existence proofs
⋮	⋮	⋮

2. **予分類したテキストを一階述語形式に変換する**
   たとえば、(1) で予分類したテキストは次の形式に変換できる。

+mlsession(d_1)	improving(d_1,1)	efficiency(d_1,2)	by(d_1,3)	...
+mlsession(d_2)	learning(d_2,1)	dnf(d_2,2)	by(d_2,3)	...
+mlsession(d_3)	constructive(d_3,1)	induction(d_3,2)	on(d_3,3)	...
-mlsession(d_4)	a(d_4,1)	comparison(d_4,2)	of(d_4,3)	...
-mlsession(d_5)	the(d_5,1)	specialization(d_5,2)	and(d_5,3)	...
⋮	⋮	⋮	⋮	⋮

   こで、+mlsession(d_1) はテキスト d_1の予分類を表す。この変換によって、すべての単語の順序関係が決まる。

3. **領域知識を定義する**
   領域知識は一階述語の形式で用意する。たとえば、文字の順序関係に関する領域

知識は以下のようになる。

- $p2 = p1 + 1$ であれば、succ(p1,p2) は true である。
- $|p1 - p2| \leq 1$ であれば、near1(p1,p2) は true である。
- $|p1 - p2| \leq 1$ であれば、near2(p1,p2) は true である。
- $p2\ p1$ であれば、after(p1,p2) は true である。

4. 再現率（Recall）、適合率（Precision）などの基準および FOIL 学習アルゴリズムを用いて、以下のような分類ルールを生成できる。

$$Recall = \frac{\#true\ \ positives}{\#true\ \ positives + \#false\ \ positives} \tag{4.2}$$

$$Precision = \frac{\#true\ \ positives}{\#true\ \ positives + \#true\ \ negatives} \tag{4.3}$$

$mlsession(S) \leftarrow learning(S, P1).$
$mlsession(S) \leftarrow decision(S, P1), trees(S, P2), succ(P1, P2).$

これらのルールの意味は次のように解釈できる。

- もしテキスト中に単語 learning があれば、このテキストは機械学習に関するものである。
- 単語 decision と単語 trees があり、かつ単語 decision が単語 trees の前に出現すれば、このテキストは機械学習に関するものである。

4.6.2 FOIL 学習アルゴリズムと帰納論理プログラミング

FOIL 学習アルゴリズムは帰納論理プログラミグ法の一種であり、分解法 (Divide-and-conquer) および "貪欲な探索" に基づくアルゴリズムである [46, 47]。学習開始時、学習するいルールの右辺は空であり、この右辺に一つずつリテラルを追加していく。節を追加したするごとにそのリテラルがカバーする例を削除し、正の事例がなくなるまでこの処理を続ける。また、どのリテラルを追加するほうが良いかに関しては、すべて可能なリテラルを考慮し、発見的知識や情報量などの評価基準を用いて選択する。

4.6.3 関係選択/文字選択

以上、機械学習手法を利用してテキスト分類について述べた。しかし、この機械学習が対象とするテキストの単語は非常に多く、特に全文検索のではその計算量が大き

くなる。従って、この基本学習方式の前処理として単語の順序を考慮した単語の選択による関係選択が必要である。

文字選択および関係選択の方法は以下の三方法である。

1. 単語頻度 tf_{ij}
   テキスト i にある単語 j の頻度の高さによる単語を選択する。

2. テキスト頻度 df_j
   収集した n 個テキストで単語 j が存在するテキストの数量。

3. 逆テキスト頻度 $\frac{n}{df_j}$
   収集した n 個テキストで単語 j が存在しないテキストの数量。

4.7 不確定サンプリング

4.6節に述べたテキスト分類と関係学習法は、基本的に教師あり学習法である。通常、関係データベースに対する教師あり帰納学習の場合、関係データベースにクラスとして使える属性があるのは普通なので、教師によってクラスとする属性を指定し、この属性に従って分類/帰納を行える。一方、テキスト分類の場合、関係データベースのようなクラスとして使えるものはテキストデータベースには無いので、教師によってクラスのラベルを付ける。つまり、付けられたクラスのラベルによって、テキストの予分類を指定する（4.6.1節を参照）。しかし、テキストデータベースにテキストが沢山があるので、ユーザによって、それぞれのテキストに対してクラスのラベルを付ける作業は大変な仕事になる。

不確定サンプリング法は、この問題点を解決する一つの方法である。

4.7.1 不確定サンプリングのアルゴリズム

以下は、単一分類子に対する不確定サンプリングのアルゴリズムである。

1. 初期分類子を作る。

2. 教師が次の手順で例に対してラベルを作る。

   - ラベルを付けてない例に対して、現在の分類子を利用して分類する。
   - あるクラスに分類された例に対して、一番不確定な b 個の例を探す。
   - 教師によりこの一番不確定な b 個の例に対して、サブサンプル用のラベルを付ける。
   - すべてのラベルを付けた例を用いて新しい分類子を作る。

このアルゴリズムは、他の分類法（たとえば、確率的分類法やニューラルネットによる分類法など）と共に利用できる。不確定サンプリングは分類の誤り例を訓練する方法と似ている。その手法と異なる点は、各事例に対して分類のラベルを付けてない時に、分類子を使用してどの事例の分類を誤ったかを推定する点である。不確定サンプリングにおいて初期的分類子が重要な役割を果たす。もし初期的な分類子がなければ、低い頻度の分類例を偶然的に発見する前に長時間にわたるのランダムサンプリングが必要になる。

4.7.2 確率的テキスト分類

本節では不確定サンプリングと共に使える確率的テキスト分類法を説明する。ベイズの定理により事後確率を推定する分類法は、テキスト分類にも応用される。

ここで、各テキストの幾つかのクラスへの分類を考える。ただし、簡単のためクラスはCと$\bar{C}$の二つとする。すると、ベイズの定理は次のように書ける。

$$\frac{P(C|w)}{P(\bar{C}|w)} = \frac{P(C)}{P(\bar{C})} \times \frac{P(w|C)}{P(w|\bar{C}}. \tag{4.4}$$

ここで$w = (w_1, \ldots, w_d)$は各テキストに存在する異なる単語である。式 (4) にロジスティック回帰法 [31] を適用すると次式が求められる。

$$P(C|w) = \frac{exp(a + b\sum_{i=1}^{d} log\frac{P(w_i|C)}{P(w_i|\bar{C})}}{1 + exp(a + b\sum_{i=1}^{d} log\frac{P(w_i|C)}{P(w_i|\bar{C})})} \tag{4.5}$$

この式に基づくと、確率的テキスト分類の具体的手順は次のようになる。

まず以下の近似式を利用して、式 (5) の$\frac{P(w_i|C)}{P(w_i|\bar{C})}$の部分を求める。

$$\frac{P(w_i|C)}{P(w_i|\bar{C})} \doteq \frac{\frac{c_{pi}+(N_p+0.5)/(N_p+N_n+1)}{N_p+d(N_p+0.5)/(N_p+N_n+1)}}{\frac{c_{ni}+(N_n+0.5)/(N_p+N_n+1)}{N_n+d(N_n+0.5)/(N_p+N_n+1)}} \tag{4.6}$$

ここでN_pは、正事例の集合に含まれる単語の数。N_nは、負事例の集合に含まれる単語の数。C_{pi}は、正事例の集合に含まれる単語w_iの生起回数。C_{ni}は、負事例の集合に含まれる単語w_iの生起回数。dは、一つのテキストに含まれる異った単語の数。

次に、式 (6) による計算結果に基づいて式 (5) の以下の部分を求める。

$$\sum_{i=1}^{d} log\frac{P(w_i|C)}{P(w_i|\bar{C})} \tag{4.7}$$

最後にロジスティック回帰法の次式を適用し、aとbを求めてから式 (5) の最後の結果を求める。

$$P(C|x) = \frac{exp(a + b_1x_1 + \ldots + b_mx_m}{1 + exp(a + b_1x_1 + \ldots + b_mx_m)}. \tag{4.8}$$

4.7.3 確率分類による不確定サンプリング

以上述べた確率分類法によって、不確定サンプリングを行なえる。簡単的に言えば、不確定サンプリングは $p(C|w)$ を計算する分類法と言える。このとき、分類の誤りは C に分類されるべきテキストが $\bar{C}$ に分類されてしまう誤りと、$\bar{C}$ に分類されるべきテキストが C に分類されてしまう誤りがある。

反復計算を行なうの時、その時点の分類子が各例に毎回適用される。$p(C|w)$ の値が 0.5 に近い例を一番不確定な b 個の例として選択する（4.7節を参照）。

4.8 概念ネットワークの構成

テキストデータベースから概念ネットワークを構成するために、自動インデックシング技術、統計およびニューラルネット技術を統合的に利用することができる [7, 5, 3, 6, 48]。

4.8.1 自動インデックシング技術の利用

1. **単語の登録**（Word Identification）
   単語を登録する時には句読点や大文字/小文字などを無視する。また各単語が出現するテキスト/節/文などを記録する。

2. **ストップワード**
   ストップワードリストを用いて一般的な単語を削除する。この一般的な単語には次の 2 種類がある。

   - **インデックスとして意味のない単語**
     たとえば、on, in, at, this, there などはインデックスとして意味のない単語である。
   - **"pure" 動詞**
     たとえば、calculate, articulate,each,listen など。

3. **ステミング**
   ステミングとは単語の接尾辞を削除することである。このためには以下の二つの機能が必要である。

   - 正当な接尾辞の形式を表示したフラグを含む辞書。
   - 接尾辞のフラグを解釈するためのルール。以下はこのためのルールの一部である。

     "V" flag:
     ... $E \rightarrow$ ...IVE as in CREATE $\rightarrow$ CREATIVE
     if $*$ .ne. E, ...*, $\rightarrow$ ...* IVE as in PREVENT $\rightarrow$ PREVENTIVE
     "N" flag:
     ... E $\rightarrow$ ... ION as in CREATE $\rightarrow$ CREATION
     ... Y $\rightarrow$ *ldots* ICATION as in MULTIPLY $\rightarrow$ MULTIPLICATION

if * .ne. E OR Y, ... → FALLEN
"X" flag:
... E → ... IONS as in CREATE → CREATIONS
... Y → ... ICATIONS as in MULTIPLY → MULTIPLICATIONS
if * .ne. E OR Y, ... * → ... * EN as in WEAK → WEAKENS

ここで*は変数であり任意のアルファベットを表わす。大文字は定数、"..." は英文字列であり、".ne." は "not equal" を表す。

このルールを用いて、ive, ion, tion, en, ions, ications, ens, th, ly, ing, ings のような接尾辞を付いている単語をステミングできる。

4. **単語–フレーズの生成**
   隣接する単語を用いて、最大三つの単語までフレーズを構成する。たとえば、隣接する単語"information retrieval system"から次のような一単語、二単語、三単語のフレーズを生成することができる。

   - "information", "retrieval", "system"
   - "information rerieval", "retrieval system"
   - "information retrieval system"

   図4.9は図4.8のテキストに対して、以上の自動インデックシングによる処理結果である。

4.8.2 概念関係分析

1. 単語頻度とテキスト頻度の計算

   (a) **単語頻度** tf_{ij}
   テキスト i にある単語 jの頻度によって単語を選択する。

   (b) **テキスト頻度** df_j
   収集した n 個テキストに単語 j存在するテキストの数量。図4.10は、図4.9の自動インデックシング結果のテキスト頻度である。

2. 情報損失分析による閾値の決定
   テキスト頻度に関する閾値を逐次増加させ、情報損失の状況を分析ながら合理的な閾値を決める。

3. 結合重みの計算
   以下の式を用いて結合重みを計算する。

$$d_{ij} = tf_{ij} \times \log df_j \tag{4.9}$$

$$d_{ijk} = tf_{ijk} \times \log df_{jk} \tag{4.10}$$

```
================================================================================
What are the problems our company needs to address to improve the product
development process and engineering/support interface?
================================================================================

1.1 Vision

1.2 Selection of markets

1.3 Selection of products

1.4 Individuals who should be focused on market or product selection full time
are burdened with day to day activities that greatly reduce their effectiveness.

......

1.10 LACK OF STREAMLINED BACK END EXTENDING FROM SILICON TO PRODUCTION.
CURRENTLY, DESIGNERS ARE RESPONSIBLE FOR MAINTAINING AND COORDINATING
THE ENTIRE PROCESS.

......

2.2 Accountability a serious problem. Commitment to a customer or to critical
market niche entry timing seems to drive us better.

......
```

Figure 4.8: テキストの一例

```
1 1 1 1 1 VISION
2 1 1 1 1 SELECTION
2 1 1 2 1 MARKET
3 1 1 1 1 SELECTION
3 1 1 2 1 PRODUCT
4 1 1 1 1 INDIVIDUAL
4 1 1 2 1 MARKET
4 1 1 3 1 PRODUCT
4 1 1 3 2 PRODUCT SELECTION
4 1 1 3 3 PRODUCT SELECTION FULL
4 1 1 4 1 SELECTION
4 1 1 4 1 SELECTION FULL
4 1 1 4 1 SELECTION FULL TIME
4 1 1 5 1 FULL
4 1 1 5 2 FULL TIME
```

Figure 4.9: 自動索引技術による処理の結果例

```
PRODUCT df = 101
DESIGN df = 59
MARKET df = 36
TIME df = 34
DEVELOPMENT df = 34
SUPPORT df = 28
ENGINEER df = 28
SCHEDULE df = 24
PRODUCT DEVELOPMENT df = 22
.
.
.
REALISTIC df = 5
REQUIREMENT df = 5
DEFINE df = 5
SIGN df = 5
```

Figure 4.10: テキスト頻度の計算の結果の例

4. 概念空間の生成
   次の非対称クラスタ関数を用いて概念空間を生成する。

$$ClusterWeight(T_j, T_k) = \frac{\sum_{i=1}^{n} d_{ijk}}{\sum_{i=1}^{n} d_{ij}} \tag{4.11}$$

$$ClusterWeight(T_k, T_j) = \frac{\sum_{i=1}^{n} d_{ijk}}{\sum_{i=1}^{n} d_{ik}} \tag{4.12}$$

非対称クラスタ関数は、二つの単語の共発生が判定できる点で自動テキスト処理で使用されている Cosine 関数と同じであるが [48]、二つの単語の順序関係を判別するために非対称クラスタ関数にしか適用できない。

合理的な数の共発生単語を獲得するために、非対称クラスタ関数を用た二つの単語の共発生の判定値は 0.1 に設定されている。それによって一番関連ある単語を概念空間に表現できる。

図 4.11に二つの単語の共発生表の例を示す。ここで強調したいのは"PRODUCT"から"DESIGN"への重みは 0.1869 であるが、"DESIGN"から"PRODUCT"への重みは 0.3460 である。この意味で"PRODUCT"と"DESIGN"の間は非相称であり、二つの単語の順序関係を判別することができる。

4.8.3 ニューラルネットによる概念分類

以上述べた概念関係分析の出力は概念記述および概念間の重み関係である。これはニューラルネットの表現とよく似ている。ゆえに相関概念を分類するために、ニューラルネットが有効である。つまり、概念空間にあるそれぞれの単語はニューラルネットでのユニットになり、クラスタ重みはユニット結合の重みとなる。

ここでは変形 Hopfield ネットによる概念分類手法を紹介する。

PRODUCT : DESIGN : 0.1869
PRODUCT : DEVELOPMENT : 0.1107

DESIGN : PRODUCT : 0.3460

MARKET : PRODUCT : 0.3373
MARKET : DESIGN : 0.1887

TIME : DESIGN : 0.1242
TIME : PRODUCT : 0.1179
TIME : SCHEDULE : 0.1023

Figure 4.11: 共発生表の例

1. **結合重みの指定**
   概念関係分析で生成した概念空間はHopfieldネットの初期状態とし、クラスタ重みはユニット間の結合重み t_{ij}（ユニット i からユニット jへの結合重み）とする。

2. **未知の入力パターンの初期化**

$$\mu_i(0) = x_i, 0 \leq i \leq n-1. \tag{4.13}$$

   ここで、$\mu_i(t)$ は時刻 t におけるユニット i の出力で、x_iはユニット i に対する入力パターンである。時刻0の時、入力ユニットは1にセットされる。各単語はそれぞれ一つの入力ユニットとして使われる。すなわち、時刻0における単語は一つだけ1、ほかの単語すべて0になる。また、この初期化および活性化の処理を n 回反復的に実行する。

3. **活性化と反復**

$$\mu_j(t+1) = f_s[\sum_{i=0}^{n-1} t_{ij}\mu i(t)], 0 \leq j \leq n-1. \tag{4.14}$$

   ここで、f_sは次のような連続シグモイド関数であり、

$$f_s(net_j) = \frac{1}{1 + exp[\frac{-(net_j - \theta_j)}{\theta_0}]} \tag{4.15}$$

   $net_j = \sum_{i=0}^{n-1} t_{ij}\mu_i(t)$ である。θ_j（たとえば、$\theta_j = 0.1$）はしきい値またはバイアスとして使用されて、θ_0（たとえば、$\theta_0 = 0.01$）はシグモイド関数の形状を修正するために使用される。

4. **収束性**
   以上述べた処理は式(11)を満足するまで反復的に行われる。

$$\sum_{j=0}^{n=1} [\mu_j(t+1) - \mu_j(t)]^2 \leq \epsilon. \tag{4.16}$$

```
neuron 0 (PRODUCT) : 0, 1, 4,  11
neuron 1 (DESIGN) : 0, 1, 4, 11
neuron 2 (SALE) : 0, 1, 4
neuron 3 (MARKET) : 0, 1, 4
neuron 4 (DEVELOPMENT) : 0, 1, 4, 11
neuron 5 (TEAM) : 0, 1, 4, 11
neuron 6 (ENGINEER) : 0, 1, 4, 6, 7, 8, 11, 23, 25
neuron 7 (SUPPORT) : 0, 1, 4, 6, 7, 8, 11, 23, 25
neuron 8 (PRODUCT ENGINEER) : 0, 1, 4, 6, 7, 8, 11, 23, 25
neuron 9 (SCHEDULE) : 0, 1, 4, 11
neuron 10 (PROJECT) : 0,1, 4, 11
neuron 11 (PRODUCT DEVELOPMENT) : 0, 1, 4, 11
.
.
.
neuron 99 (PRODUCTION) : 0, 1, 4
neuron 102 (STANDARD) : 0, 1, 4
neuron 105 (RESPONSIBILITY) : 0, 1, 2, 4, 5, 6, 7, 8, 10, 11, 23, 25, 37, 46, 63, 108
neuron 108 (REQUIREMENT) : 0, 1, 4, 11
neuron 109 (DEFINE) : 0, 1, 4, 11
```

Figure 4.12: Hopfield ネットによる概念分類の結果の例

ここで、ϵは最大許容誤差（たとえば、1）である。最後の出力は、最初に与えられた単語と関連性ある単語の集合である。

図 4.12に示すのは Hopfield ネットによる概念分類の結果の例である。この例では、各単語（全部 111 個単語）を最初の入力パターンとして使用し、Hopfield ネットは 111 回活性化される。また、単語発生によって一つの単語ずつ活性化する。たとえば、図 4.12ではニューロ 0 は単語"PRODUCT"、ニューロ 1 は単語"DESIGN"、ニューロ 3 は単語"MARKET"をそれぞれ表現する。図 4.12に単語の収束グループを示している。たとえば、ニューロ 0 すなわち"PRODUCT"は、次の四つの単語、"PRODUCT", "DESIGN", "DEVELOPMENT", "PRODUCT DEVELOPMENT" すなわち、ニューロ 0, 1, 4, 11 に収束する。

4.9 遺伝的アルゴリズムによる概念の抽出

最近、遺伝的アルゴリズムによるテキストから概念を抽出することも試されている [4, 17]。この抽出方式は、学習の進化的モデルに由来している [21]。遺伝的アルゴリズムは、予測するために競争する個体の集まり（個体群）からなる。うまく予測しない個体は、よりうまく予測する個体が少し異った子孫を作りながら増殖する間に捨てられる。"適者生存" というダーウイン説の類推から、その個体群は時間とともに改善される。

4.9.1 遺伝的アルゴリズムの一般的な手順

遺伝的アルゴリズムは、具体的に次の五つのステップに分けられる。

1. 初期化
   ランダムな染色体を持つ個体を N 個生成して、初期世代の個体群を設定する。

2. 再生
   各個体の適合度を計算して、適合度に依存した一定の規則で個体の再生を行う。ここで、適合度の低いいくつかの個体は淘汰され、この個数だけ適合度の高い個体が増殖することになる。

3. 交叉
   設定された交叉確率や交叉の方法により交叉を行い、新しい個体を生成する。

4. 突然変異
   設定された突然変異確率や突然変異の方法により突然変異を行い、新しい個体を生成する。その結果、新しい世代の個体群が生成される。

5. 終了判定
   終了条件を満たせば、その時に得られている最良の個体を問題の準最適解とする。そうでなければ、手順 (2) へ戻る。

以上述べた遺伝的アルゴリズムの主要部分は、手順 (2) での適合度の高い個体を次世代により多く残す再生の手続きと、手順 (3)(4) における交叉と突然変異により新しい個体を生成しながら次世代の個体群を生成する手続きである。遺伝的アルゴリズムは、このように再生により望ましい解を重点的に探索すると同時に、交叉と突然変異によって解の探索範囲を広げているので、これらの両方の手続きが有効に作用すれば、その効果を十分に発揮できる。

4.9.2 遺伝的アルゴリズムによるテキストから概念抽出の例

テキストから概念を抽出するために遺伝的アルゴリズムを利用する場合、各染色体すなわち文字列は、あるテキストに含まれるキーワードまたは概念を表す。染色体での遺伝子の位置はその染色体を表すテキストにある概念が、存在するかどうかを決める。存在すれば、その位置は 1 をセットし、存在しなければ、その位置は 0 をセットする。従って、染色体はキーワードまたは概念を表すコード化したテキストである。

初期的個体群は、ユーザによって選択したテキストの集合である。表 4.9.2にはユーザが選択した五つのテキストに含まれるキーワードである。また、この五つのテキストに含まれる異なったキーワードは表 4.9.2に示すように 33 個ある。

次に、この五つのテキストにある異るキーワードをコード化した個体の適合度を計算する。この適合度の計算は、テキスト間の相関性を評価するための次の Jaccard 得点記録（0–1 間の値）法を用いる。

Table 4.1: ユーザによって選択されたテキストとキーワード

テキスト番号	Keywords
DOC0	Data Retrieval, Database, Computer Networks, Improvements Information Retrieval, Method, Network, Multiple, Query Relation, Relational, Retrieval, Queries, Relational Database US, Carat.dat, Gqp.dat, Orus.dat, Query.opt
DOC1	Information, Information Retrieval, Information Storage, Indexing Retrieval, Storage, US, Kevin.hot
DOC2	Artificial Intelligence, Information Retrieval Systems Information Retrieval, Indexing, Natural Language Processing US, DBMS.AI, Gqp.dat
DOC3	Fuzzy Set Theory, Information Retrieval Systems, Indexing Performance, Retrieval Systems, Retrieval, Queries US, Kevin.hot
DOC4	Information Retrieval Systems, Indexing, Retrieval, Stairs US, Kevin.hot

Table 4.2: 五つのテキストにある異るキーワード

Data Retrieval, Database, Computer Networks, Improvements
Information Retrieval, Method, Network, Multiple, Query
Relation, Relational, Retrieval, Queries, Relational Databases
Relational Database, US, Carat.dat, Gqp.dat, Orus.dat, Query.opt
Information, Information Storage, Indexing, Storage, Kevin.hot
Artificial Intelligence, Information Retrieval Systems
Natural Language Processing, DBMS.AI, Fuzzy Set Theory, Performance
Retrieval Systems, Stairs, Kevin.hot

$$\#(X \bigcap Y)/\#(X \bigcup Y) \tag{4.17}$$

他のテキストと共通概念 (Keywords) が多いのテキストは、Jaccard 得点記録が高い。つまり、高い得点をもらったテキスト間は相関性が高い意味する。たとえば、テキスト 0 とテキスト 0,1,2,3,4 間の Jaccard 得点記録および平均適合度は次のようになる。

DOC0 と DOC0 の Jaccard 得点記録 = 1.0
DOC0 と DOC1 の Jaccard 得点記録 = 0.1154
DOC0 と DOC2 の Jaccard 得点記録 = 0.12
DOC0 と DOC3 の Jaccard 得点記録 = 0.1154
DOC0 と DOC4 の Jaccard 得点記録 = 0.0833
DOC0 の平均適合度（Jaccard 得点記録）= 0.28682

従って、ユーザによって選択したテキストに対して、初期的コード化した個体群における染色体および平均適合度は以下のようになる。

染色体	適合度
11111111111111111111100000000000000	[0.2868]
00001000000100010000111110000000	[0.3702]
00001000000000010100001001111000	[0.3502]
00000000000110010000010101001110	[0.3845]
00000000000100010000010101000001	[0.3914]

平均適合度 = 0.35662

また、再生、交叉および突然変異などの操作によって、次の最良の個体群における染色体および平均適合度を求めることができる。

染色体	適合度
00000000000100010000010101000001	[0.4512]
00001000000100010000010101000001	[0.4512]
00001000000100010000010101000001	[0.4512]
00000000000110010000010101000001	[0.4512]
00000000000100010000010101000001	[0.4512]

平均適合度 = 0.4512

最後に最良の個体群をデコード化して、次の概念が獲得される。

Retrieval, US, Indexing, Kevin.hot,Information Retrieval Systems, Stairs.

つまり、これらのキーワードはこの五つのテキストに対する代表的な概念として判定された。

4.10 テキストデータベースの構造化プロセス

以上、テキストデータベースからの構造化の情報を抽出するための統計および確率的な手法、一階帰納学習、ニューラルネット技術、遺伝的アルゴリズムなどの幾つかの方法を述べた。

構造化の対象となるテキストデータベースの特性（たとえば、大規模性、曖昧性、データの多様性、動的性など）を考慮するする必要がある。テキストデータベースから構造化の情報を抽出するためには、一つの学習/発見方法を利用だけでなく、

- 多面的データ分析
- 多段階学習
- 概念抽象化やノイズデータに対する処理

などが可能な学習機構を集つめてツールボックス化することが必要であり、多段階学習や概念抽象化を行う構造化プロセスを動的生成する技術が必要である。

4.10.1 GLS 発見方法論への適用

4.2節で述べたGLS発見方法論に基づいて考えると、テキストデータベースからの構造化の情報を抽出するためのプロセスは、大分類するとプリプロセス、構造抽出、管理/精緻化という三つの学習フェーズに分けられ、これらの各フェーズで多面的データ分析および多段階学習や概念抽象化を行なえる。

1. **プリプロセス**
   このフェーズでの重要な処理はユーザとの対話に基づいてユーザの要求を収集し、使用したい構造化情報抽出法を確定し、テキストデータの収集と整理などを行う。たとえば、
   - 自動インデックスによる単語の登録
   - ストップワード
   - ステミング
   - 単語-フレーズの生成
   - 頻度による単語選択
   - 不確定サンプリング

   などを行う。

2. **構造抽出**
   構造抽出フェーズではプリプロセスフェーズの処理に基づいて構造抽出を行う。たとえば、このフェーズでは統計および確率的な手法、一階帰納学習、ニューラルネット技術、 遺伝的アルゴリズムなど方法を利用し、概念や概念クラスタを生成する。

3. **管理/精緻化**
   管理/精緻化のフェーズでは前フェーズで抽出した初期概念や仮説を適当な表現形式に変換して知識ベースに追加して精緻化や管理を行う。すなわち、テキストデータベースから初期概念や仮説を発見し、それを知識ベース管理システムによって管理すると共にメタ推論と多重世界の機能を用いた知識の精錬手法と組み合わせて、それを精緻化する。また、テキストデータベースの変化や新しいデータを収集した時に、この変化や新しいデータに対応して精緻化や管理を行う。

この三つの学習フェーズによって、知識ベースとデータベースの統合利用の環境上でテキストデータベースからの構造化の情報を発見するための多面的データ分析および多段階の学習や概念抽象化が可能になる。ここでの発見/学習とは、ユーザの要求によってテキストデータベースからの構造化だけでなく、動的に発見プロセスの組織化、発見プロセスの制御と性能改善なども行う。このために、図4.5に示すような多重世界のメカニズムによってマルチレベルに分けてシステムを管理する。

4.10.2 統合化のシステムに向けて

全ての文献を検索する場合、文献に多種類データがあることを考慮する必要がある。たとえば、文献には文字列だけでなく表や図も含まれている。特に、表には数値データがある場合は普通である。この場合は4.3.2節、4.3.1節で述べたKOSI法やDBI法などを利用して表に意味ある関係を発見することができる。

一方、テキストデータベースの構造化の結果は関係データベースからの知識発見のための領域知識として利用できる。データベースの構造化の方法を大分類すれば、データの固有的価値による構造化とデータの意味的価値による構造化になる。データの固有的価値による構造化は、データの意味を考えず数学的式や発見的知識などのような客観的基準を利用して評価して行なう。この場合は、構造化の自律性が高くなれるが、構造化の結果はある分野の常識やユーザの要求と合わなく場合がある。一方、データの意味的価値による構造化は、データの意味を重視して領域知識の利用やユーザとインタラクションしながら構造化を進める。この場合は、構造化の結果がある分野の常識やユーザの要求と合うが、構造化の自律性がなくなる。

以上の議論から、理想的なKDDシステムはデータの固有的価値と意味的価値との両方を重視して行なうべきである。このために、大量な分野特異な領域知識を保存している知識ベースが必要である。テキストデータベースからの自動構造化技術はこの知識ベースを自動構築する方法としても適用できる。また、全文献を検索する場合は、文献の内容を意味的に解釈するために自然言語解析の技術の利用も必要になる。

5 エージェント技術の応用

本章では、まずエージェント技術の動向についてサーベイを行なう。特に、自律分散型エージェントについて取り上げ、ネットワーク上のサーバから構造化情報を得る方式について考察する。

5.1 エージェント

近年、エージェントに関する研究が盛んに行なわれている。しかし、“エージェント”の意味するものはインテリジェントエージェント、インタフェースエージェント、ソフトウェアエージェント、ネットワークエージェントなど多岐にわたっている。研究開発が行なわれているエージェントは、以下のような機能をもつエージェントに整理できる。

- ソフトウェアエージェント
  ソフトウェアで実装されるエージェントの総称である。ユーザの代理人として種々の作業を支援する。
- インテリジェントエージェント
  エージェント内部に問題解決や学習を行なう仕組みやそのための知識を持つもので、知的に振舞うエージェントの総称である。
- 自律分散型エージェント
  エージェントの基本特性としての自律性に着目した場合の呼称である。各エージェントは相互に協調しながら分散処理を行なう。

5.2 インテリジェントエージェント

インテリジェントエージェント研究の一貫として学習機能を備えたインタフェースエージェントの研究が行なわれている [30, 29]。インタフェースエージェントを構成するアプローチとして次の3つのアプローチがある。

1. プログラミングアプローチ
   ユーザがその目的やタスクに応じた機能を、例えば、ルールとして記述すること

によってエージェントを実現する。そしてエージェントは状況に応じてこれらのルールを自動的に実行し、ユーザの作業を支援する。

2. 知識型アプローチ
知的インタフェースエージェントを構成する代表的な手法といわれている。インタフェースエージェントに対して、ユーザやアプリケーションに関するさまざまな領域知識をドメインモデルもしくはユーザモデルといった形式で与えておく。エージェントはこうした知識を利用して、ユーザのプランを認識したり、ユーザをサポートする時期などを判断する。

3. 学習アプローチ
ある条件下で学習を行なうインタフェースエージェントを構成するものである。エージェントの学習方式としては、以下の3方式がある。

(a) 観察学習
インタフェースエージェントはユーザの操作や振舞いを観察し、そこに現れる行動パターンを発見し、それを自動化するルールを学習する方法である。

(b) 強化学習
ユーザからの直接的あるいは間接的なフィードバックに基づいて学習を行なう方式である。間接的フィードバックはユーザがエージェントからの提案を無視して別の動作を行なったときに行なわれる。直接的なフィードバックはエージェントによって実行された動作を検出してそれに対する明示的な反論を提示したときに行なわれる。ネガティブなフィードバックに基づいてその状況に適した動作を提供する新たなルールが学習される。

(c) 教示学習
ユーザがある事象や状況に関する仮想的な例題を与え、それに対してエージェントがどのように対処すべきかを示すことによりエージェントを訓練する方法である。エージェントは与えられた例の動作系列を記憶し、種々の要素間の関係を調べその例をを取り込むように知識ベースを変更する。

5.3 分散エージェント技術

エージェントがもつ基本特性は、主に自律分散型エージェント焦点を当てた場合、以下の各点に整理できる [60]。

- 自律性
エージェントは自己の行動や内部状態を制御する仕組みをもち、他のシステムや人間から直接的な干渉を受けることなく自律的に動作する。つまり、エージェントは自分の知識や種々の情報を活用して自ら判断を行なえる。こうした自律性はエージェントを特徴づける主要な特性として認知されている。

- 社会性
  エージェントは他のエージェントや人間との相互作用が必要不可欠である。そこでは双方の対象や理解可能な言語やプロトコルが導入され、エージェント同士あるいは人間との情報交換が実現される。これによって、たとえば複数のエージェントが一つの作業組織を構成して互いに協力しながら協調的に問題解決を行なう。

- 反応性
  エージェントは自分が置かれた環境を認識し、そこでの変化に対して適切に反応する。すなわち、エージェントの枠組では他のエージェントや人間を含む外部環境との相互作用が、各エージェントの振舞いに影響を与える重要な要素となる。

- 自発性
  エージェントは、単に外部環境に応じて反射的に動作するだけではなく、ある目標を目指して自発的に行動できる。つまり、エージェントは何らかの目標達成に必要となる処理や作業に対して能動的に協力する。

自律分散型エージェントの研究分野、またはインフラとなる技術を列挙すると以下のようになる [39]。

- 共通知識（オントロジー）
  - ボキャブラリー
  - シンタックス
  - セマンティックス
  - 用語法
- 言語/エージェント間のインタフェース
  - KQML
    Search–Act 理論に基づくヘテロな知識ベース間の情報交換プロトコル。
  - KIF
  - Java
    WWW 間のコミュニケーション。
  - AgenTalk
    ポイント・ツー・ポイントコミュニケーション
- ゴールの認知
  部分ゴールプランニング。
  - 共有ゴール、すなわちグループ構成
  - 個別ゴール
    * 協調作業

    * 非協調作業

- 協調プランのスキーム

エージェントの基本特性である自律性、社会性、反応性、自発性を実現するためには、以下のような機能を備えた機能モジュールが必要である。

1. 処理モジュール
    - 協調制御機能
    - 問題解決機能
    - 知識管理機能
2. 知識モジュール
    - 協調知識
    - 問題解決知識/制御知識
    - 知識管理機能
3. コミュニケーションモジュール
    - メッセージ/イベント処理機能
    - 協調プロトコル
    - 通信プロトコル

1980年中ごろから分散処理型のシステムやソフトウェアモジュール間の協調処理に関する研究がなされて来た。現在の協調分散型エージェントはこれらの研究の延長線上にあるものといえよう。以下の節で代表的な研究を2例紹介する。

5.3.1 MACE

MACE（Multi-Agent Computing Environment)[15] は分散人工知能システム構築環境の汎用的な枠組として提案されたシステムである。MACEでは、互いにメッセージを交換しながら並列に動作する多数のシステム構成要素をエージェントと呼んでいる。

MACEエージェントは能動的にメッセージパッシングを行なう自己完結的なオブジェクトである（図5.1参照)。

各エージェントは2種類の知識、すなわちエージェントが担当する処理に関するドメイン知識と獲得モジュールと呼ばれる他のエージェントのモデルに関する知識を持つ。これらの知識を用いて各エージェントは外部から送られたメッセージによって外部環境を認識し、他のエージェントやMACEシステムにメッセージを出したり、自分の内部状態を変更する。こうしたメッセージ解釈やエージェントの内部処理はMACEエージェントのエンジンによって実行される。

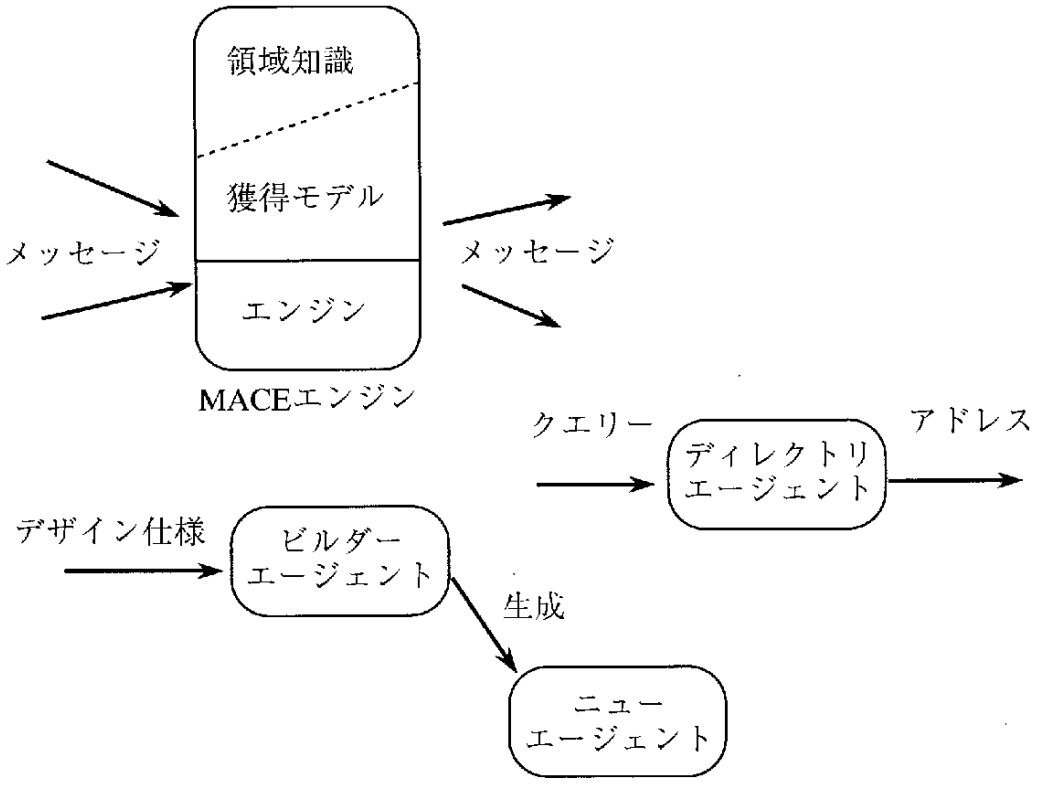

Figure 5.1: MACE

5.3.2 ARCHON

ARCHON[59] は ESPRIT プロジェクトで研究開発されたマルチエージェントシステムに関するフレームワークであり、監視/制御システム、CIM、ロボットなどに応用されている。ARCHON では ARCHON レイヤーとよばれる機構をシステムに付加することによりエージェントが構成される（図 5.2参照)。

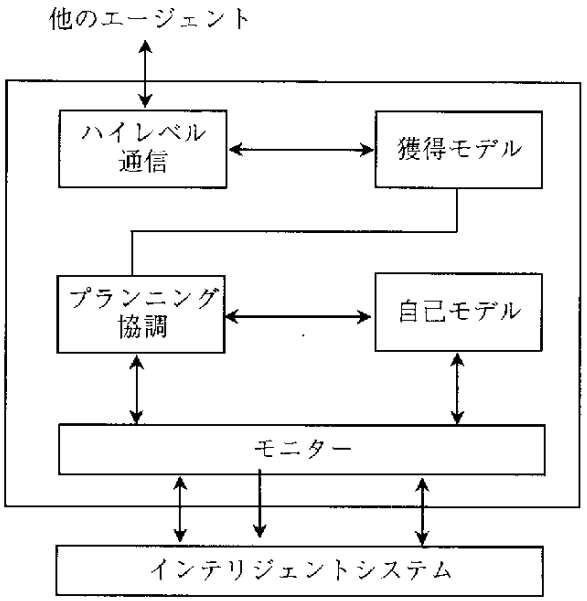

Figure 5.2: ARCHON

システム機能要素はインテリジェントシステムと呼ばれ、既存のエキスパートシステムやデータベースシステムがこれに相当する。ARCHON レイヤーは次の 5 つの要素から構成されている。

- 知識モデル
  他のエージェントに関する知識。
- 自己モデル
  自分自身に関する知識。
- 高次通信モジュール
  他のエージェントとの情報交換や通信を行なうモジュール。複数の協調プロトコルを提供する。
- プランニング/協調制御モジュール
  エージェントの振舞いの監視や、他のエージェントと協調する際のプランニングを行なったりするモジュール。

- モニタ
  ARCHON レイヤーとインテリジェントシステムを接続するモジュール。インテリジェントシステムの動作の監視と制御を行なう。

5.4 トランスポータブル・エージェント

ネットワーク上ではデータは分散し多様なフォーマットで蓄積されている。そしてデータの解釈には固有の矛盾がある。分散した電子的リポジトリーと相互作用するときの基本的な問は“どこで計算を行なうか?”、“計算を行なうためにデータをもっていくべきか、それとも計算機構をデータの所に持って来るのか”ということである。自律型エージェントは計算機機構をデータにもってくるような計算パラダイムを提供する。これはエージェントが決断をしたり情報フィルターを行なうときに実質的な自律性をもつ必要があるからである。

このような機構を提供する計算機構として、最近 Java[49] や MagicCaps[16] といったテレスクリプト言語が製品化されている。自律型エージェントはヘテロなネットワーク上をナビゲートするソフトウェアモジュールである。自律型エージェントはトランスポータブル（transportable）な、またはマイグレート（migrate）できるソフトウェアである。すなわち、実行を任意のポイントで一時停止し、他のマシンへ移動した後に実行を再開できる。

機能的にみると自律型エージェントは以下のような点でネットワーク上で処理に適している。

- 自律的ナビゲーション
- ネットワーク環境の変化のセンシングと対応能力
- 情報収集における協調的処理

自律的ナビゲーションとは、エージェントが自由にかつ独立的にネットワーク上をトランバースできることを意味している。ネットワーク環境変化のセンシングとは、エージェントがハードウェアやソフトウェアの条件をセンシング値として独立的に認識する能力である。そして、協調的な情報収集とは複数のエージェントが分散した情報を入手するために相互作用することを意味する。

自律エージェントの基本的なモジュールは“センサー”と“エフェクター”である。自律エージェントはファイル、データベース、ネットワークトラフィックそして他のエージェントの状態の検知を行なう仮想センサー、そして、マイグレーションによって異なった物理的位置に移動する仮想エフェクターのネットワークから構成されている。

このようなエージェントは以下のような利点を持つ。

- エージェントは全てのデータを要求されたサイトに送らず、小さなエージェントをデータソースに送るのでネットワークトラフィックを減らせる。

- データはリポジトリから移動させる必要がないのでデータのインティグリティを向上できる。
- エージェントはネットワーク接続が保持されないPCやラップトップでもネットワークを移動し自律的に稼働するので、信頼性の低い非定常的なネットワーク接続であるモーバイルコンピュータをサポートできる。

一方、自律型エージェントにおけるな問題は以下の各点である。

- 悪意のあるエージェントからマシンを守る、もしくは悪意のあるマシンからエージェントを保護する機能。
- ネットワーク上の不確実な世界における効果的なフォルトトレランス性の実現。
- プログラマーが簡単かつ素早くエージェントを書いたりデバッグできるような環境。

5.4.1 仮想エフェクター

自律型エージェントの重要な機能である仮想エフェクターを Agent Tcl[18] を例として述べる。Agent Tcl は Tcl[42] を拡張した言語である。Agent Tcl におけるマイグレーションは agent_jump コマンドによって行なわれる。他の場所に移動しようとする Tcl スクリプトは agent_jump コマンドを発行することによってマイグレートできる。agent_jump コマンドが発行されると、それをコールした Tcl スクリプトは実行が一時停止され、その時のスクリプトの内部状態はパッケージングされ保護される。そのイメージはマイグレート先のマシン上のサーバに送られる。サーバはそのイメージをリストアし、実行を一時停止したポイントから Tcl スクリプトは再開する。Tcl スクリプトは、サーバを仲介者とした方式もしくは TCP/IP 接続のいずれかによるメッセージパッシングによって通信を行なう。

5.4.2 仮想センシング

マシンがアップしたあるいはダウンした、リポジトリに蓄積されている情報が変更された、そして情報収集を行なうために必要な作業を停止するための正確な手続きはエージェントが“世界”へ発射されたときには完全にはわからない。その環境における動的な変更を認識し適用する方法がないので、外部状態の認識が無いと自律的エージェントは活動不能になってしまう。Agent Tcl では以下の 4 つの特徴による操作、そして決断を自律的に行なう。

- 彼らの世界で発生した動的な変更をセンスする能力
- それらの変更に対応する能力
- 他のエージェントと通信する能力

- より安定したネットワークへマイグレートする能力

外部環境のセンシングは以下のように類別できる。

- ハードウェアの変更
- ソフトウェア的な変更
- 他のエージェントによる状態変更

[18] におけるエージェントはネットワークサイトが到着可能かどうかを決定できる。また、あるネットワークを通過する時間やあるサイトでの処理時間を予想できる。

この情報はリソースの複製コピーあるいは同じ情報を与えるリソースの選択をエージェントにさせる。この情報は、どこに、いつ、エージェントを作るかどうか、そしてリソースを共有するためにどこでランデブーさせなければならないかの判断をエージェントにさせる。

ネットワーク環境のセンシングを行なうエージェントの例を以下に示す。

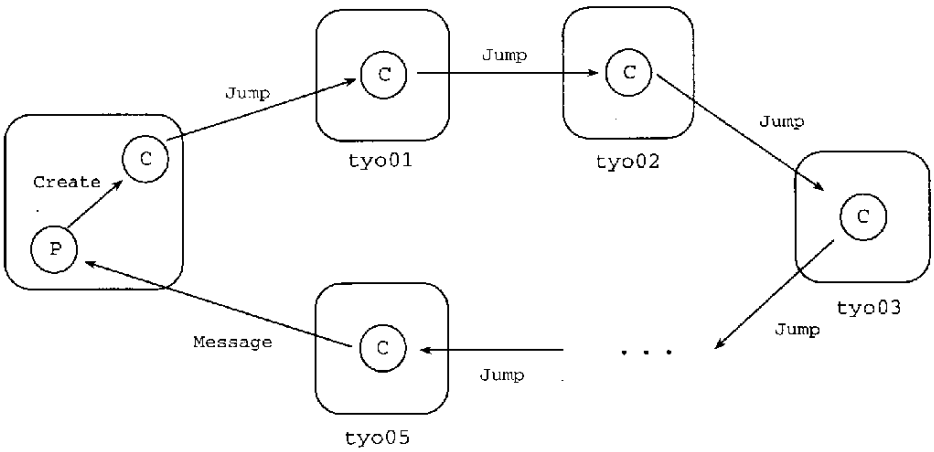

Figure 5.3: ネットワークセンシングエージェント

自律型エージェントは、リソースが使用できない、希望した情報がない、未知のサイトにさらに関連する情報が含まれかもしれない、と言った問題に直面する可能性が高い。アプリケーションに応じて、エージェントはエラーレポートを出力する、他のリソースへと移動する、リソースが使えるようになるまで待つといった選択を行なう。

このような選択を行なうためにエージェントは現在のリソースの状態をセンスしなければならない。エージェントが外界のセンスを行なうためには 2 つの選択手段がある。

- エージェントが一定のインターバルでポーリングをかける。
- リソースが変更されたときそれを知らせるメッセージを送るローカルサービスに登録する。

```
#子エージェント用のプロシージャの定義
proc who machines{
  global agent
  set list ""

  #マシン間をジャンプし、who コマンドを実行する
  foreach m $machines {
    if {catch "agent_jump $m"} {
      append list "$m:\n unable to JUMP to this machine"
    } else {
      set users [exec who]
      append list "$agent(local-server):\n$users\n\n"
    }
  }
  return $list
}

#サーバ名を得る
set machines "tyo01 tyo02 tyo03 tyo04 tyo05"

agent_begin

#子エージェントを実行する
agent_submit $agent(local-ip) -vars machines -proc who -script {who machines}

#終了を待つ
agent_recive code string -bloking
put $string

#エージェント終了
agent_end
```

Figure 5.4: Agent Tcl によるポータブルエージェント

ほとんどの自律型エージェントがこの２つの手段を混合して使用している。

エージェントにとってのもう１つの情報ソースは他のエージェントである。ある情報を提供しようとするエージェントがあり、そのサービスを必要とするエージェントがいる場合、そのエージェントは適切なサーバとなるエージェントと通信する必要がある。エージェントはどのエージェントがどのリソースにアクセスしているかを、サーバとなるエージェントがすでに情報にアクセスしているかどうかを尋ねなければならない。同様に、あるエージェントが他のユーザに情報を提供しているエージェントを観察し、この情報をフィルタリングした後で自己の組織化に利用したいかもしれない。

このような技法は処理の重複を減らすのに有効であり、双方の技法ともシステム的なサポートを必要とする。

最初の技法を実現するためには、

- ある時間枠内にリソースにアクセスしたすべてのエージェントのヒストリーを記録する。
- 場所独立な名前空間とトレースメカニズムによって、あるエージェントがサイト去ったあとでも他のエージェントが検索や問い合わせを行なえるようにする。

二番目の技法を実現するためには、名前変換とディレクトリサービスが必要である。それらによって、エージェントはどのエージェントが同じあるいは類似したサービスを他のユーザに提供したかを知ることができる。

5.5 WWW ロボット

WWW ロボット（ワンダラーもしくはスパイダーとも呼ばれる）は Web 上のドキュメントを検索する、あるいはそのドキュメントが参照している他のドキュメントを再帰的に検索することによって Web 上のハイパーテキストを自動的にトラバースするソフトウェアである。ここでの再帰的とはある特定のトラバースアルゴリズムを意味するわけではない。現在、様々なタイプの WWW ロボットがリリースされており、それらはいろいろな検索アルゴリズムを用いている。通常の WWW ブラウザーは参照するドキュメントの検索がユーザによって行なわれていることから、WWW ロボットには分類されない。

WWW ロボットは Web ワンダラー、WWW ワーム、スパイダーとも呼ばれている。この名称によって WWW ロボットはコンピュータウィルスのようにサイト間を自律的に移動するソフトであるといった誤った印象を与えられる。しかし、WWW ロボットはあるドキュメントからの参照経路に沿って単にドキュメントを渡り歩くだけである。

WWW ロボットはサーチエンジンと密接な関連がある。サーチエンジンとは、自ノード内のデータベースに蓄積されたデータ使用して検索を行なうプログラムである。Web のコンテクストにおけるサーチエンジンとは WWW ロボットが収集した HTML ドキュメントのデータベースを用いて検索を行なうソフトといえる（図 5.5参照）。

WWW ロボットが行なう “情報収集” 活動は以下のように分類できる。

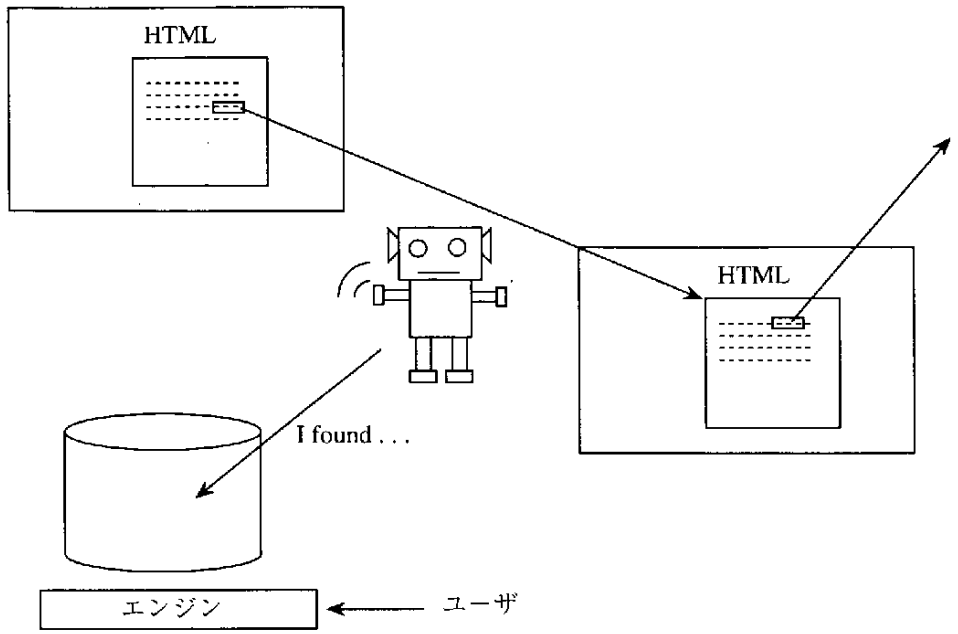

Figure 5.5: サーチエンジン

- インデックス作成
- HTML の確認
- リンクの確認
- “What’s New” のモニタリング
- ミラー化

これらの機能のなかで、現在最もよく用いられているのがインデックス作成 WWW ロボットである。WWW ロボットがどのサイトをどのようなアルゴリズムで訪問するかは各ロボットごとに異なる。一般的には以下のようなリソースから検索を始める。

- URL のヒストリーリスト
- サーバリスのような他のノードへのリンクを多く持っているドキュメント
- “What’s New” ページ
- Web 上のポピュラーなサイト

HTML ドキュメント上の情報の他に USENET へアクセスしメーリングリストのアーカイブなどから情報を得るシステムもある。

このような情報収集の開始点が与えられるとWWWロボットはそのサイトを訪問し、インデックスを作成するURLを選択し、それらをパースして新たなURLとして使用する。

また、HTML情報のパースの仕方も各WWWロボットによって多様な方法がとられている。WWWロボットによって用いられるHTML上の情報は、次のような情報である。

- タイトル
- 最初の2～3パラグラフ
- HTML全体をパースしHTMLの構成に応じて全てのワードを重み付きでインデックス化する。
- MATAタグ情報
- "隠れ"タグ情報

5.5.1 ロボット除外ののための標準化

1993年から1994年にかけて、いろいろな理由からロボットの訪問を歓迎しないサーバをロボットが訪問してしまうことがあった。その理由とは、ロボットが特殊であった、すなわち一定のロボットが頻繁に要求を出すことによってサーバをダウンさせたり、何回も同じファイルを検索したりするなどである。また、ロボットが不安定なサーバにトラバースしてしまうこともあった。

これらのことからロボットにWWWサーバのどの部分をアクセスしてはならないかを知らせるWWWサーバ上のメカニズムを確立する必要があることが指摘された。

サーバからロボットを排除するために使われた方法は、サーバ上にロボットに対するアクセスポリシーを記述したファイルを作成しておくことであった。このファイルはローカルなURL"/robot.txt"でHTTP経由でアクセスできなければならない。

"robot.txt"は一つあるいは複数のレコードから構成されている。各レコードは一行あるいは複数行のDisallow行に続く、User-agent行から構成されている。各行の詳細は以下のとおりである。

- User-agent
  このフィールドの値はそのアクセスポリシーを与えるロボットの名前である。
- Disallow
  このフィールドはアクセスしてはならないURLを記述する。

5.5.2 WWWW

WWWWはコロラド大学で開発されたWWWサーチエンジンである。WWWWによってWWWハイパーテキストやWWW上の情報リソース（URL）をキーワード検索できる。WWWWは以下の4種類の検索用データベースを提供している。

1. 引用しているハイパーテキスト

2. 引用しているアドレス（URL）

3. HTML のタイトル

4. HTML のアドレス

WWWW における検索はキーワードリストによって行なう。これらのキーワードを AND 結合あるいは OR 結合することもできる。また、WWWW によって返される結果の件数を指定することもできる。

WWW ワームは各サイトの WWW 上のページにリンクされたページを再帰的に検索しながらトラバースし、その結果を WWWW に返す。WWW ワームが検索する対象は HTML 間のアンカー情報だけではなく URL 間のアンカー情報も対象となる。

5.5.3 WWW ロボット

インターネット上では WWW ロボットによる情報収集サービスやリソースインデックスサービスが行なわれている。コマーシャルベースあるいはフリーウェアベースを含め、すでに 60 以上の WWW ロボットがインターネット上でサービスを行なっている。以下で代表的な WWW ロボットサービスについて紹介する。

- *WebCrawler*
  WebClawle 社によってサポートされている WWW ロボットである。WebClawle の原型はワシントン大学におけるリソースディスカバリーの実験システムであり、America Online でサービスが提供されている。WebCrawler の目的はリソースを発見するためのデータベースの生成と統計情報の生成である。

- *fish search*
  蝿に関する情報を収集する WWW ロボットであり、オランダで開発された。

- *MOMspider*
  リンクの妥当性をチェックし、統計情報を生成するロボットである。カリフォルニア大学で 1993 年に開発されたロボットである。

- *Lycos*
  カーネギーメロン大学で開発された WWW ロボットである。Lycos は知的で直接的な検索を行なうために Web の関する特殊なモデルを使用している。

- *Emacs-w3 Search Engine*
  Emacs Lisp で書かれた WWW ロボットであり、リソース探査用のデータベースを生成する。Emacs-w3 Search Engine はブラウザーとサーチエンジンが統合された形で提供されている。

- *Mac WWWorm*
  Macintosh 上で稼働するフランス語のキーワードサーチロボットである。ロボット本体は HyperCard で記述されている。

- *Webwalk*
  HP 社内で使用されている WWW ロボットである。Webwalk の目的は、リソース発見用データベースの生成、リンクの妥当性チェック、HTML の妥当性のチェック、ミラーリング、ドキュメントツリーのコピーそして統計情報の生成などである。Webwalk は仮想的なメンテナンスコマンドを実行できるような拡張がなされている。

- *InfoSeek Robot*
  InfoSeek Robot は InfoSeek 社によってサポートされている WWW ロボットである。InfoSeek 社におけるフリーの WWW サーチおよびコマーシャルベースでの WWW サーチの双方で使用されている。最も人気がある WWW ページを発見するために独自の情報収集アルゴリズムを用いている。

- *TITAN*
  NTT で開発された WWW ロボットである。現在の TITAN の目的はリソース探査用データベースの生成、ドキュメントツリーのコピーであり、その最終目的は WWW ドキュメントのインデックス手法の開発とされている。TATAN はスタンドアロンプログラムであり、Perl で記述されている。

- *WebWatch*
  URL の変更をチェックする WWW ロボット。シェアウェアである。

- ArchitextSpider
  Architext ソフトウェア社によってサポートされている WWW ロボットである。ArchitextSpider の目的は、リソース探査用データ生成および統計情報の生成である。ArchitextSpider は Architext のインターネットナビゲーションサービスから実行情報を収集する。

5.6 自己組織型情報カタログ

エージェントはタスクと環境双方に関する部分的あるいは部分的知識を必要とする。この情報はエージェントの内部状態を構成し、エージェントの行動方法を決定するのに使われる。

エージェント間の共通知識あるいはエージェント間のコミュニケーションを成立させるためにエージェント間の通信に用いられる語彙に関する合意が必要である [38]。エージェント間の共通語彙を実現するアプローチとしてオントロジーを用いるアプローチがある。このアプローチでは、システム開発に先だってエージェント系の共通オントロジーを作成する。ARPA（Adbance Research Promotion Agency）の知識共有部会で

はOutolingua[19]を用いてエージェント系で用いられる共通オントロジー（系に共通する概念/用語体系）を陽に記述することが提案された。

各エージェントは一定のナビゲーションシーケンスによってナビゲートされる。最初のプランは情報カタログによって提供され情報から抽出されたディレクトリ情報によるサイトのシーケンスである。この情報は静的に保持されるのではなく、エージェントの仮想センサーによって認識された情報によって動的に再構成される。ここで参照される情報は以下のような情報である。

1. ネットワークトラフィックをモニターする能力と、あるサイトが稼働中であるか否かに関する情報により、エージェントはそのマシンを通過する時間を最小限にするためにプランを再配置する。

2. ソフトウェアの変更を認識する能力により、エージェントはそのマシンでの処理時間を最小限にするためにプランを再配置する。

3. あるサイトにおける検索や問い合わせ処理結果によりプランを変える。

一方、WWWロボットのような検索情報では、あるコンテンツとそのコンテンツを持つリソースのアドレスが重要な処理要素となる。ユーザが要求する検索事項とそれにマッチするリソースとの対応は、URL名やHTMLの一部とのマッチングに限定されている。また、検索結果の処理方法もプリミティヴな方法に限られている。

ネットワーク上で情報検索やリソース情報を作成するためにはトランスポートエージェント機構は有効な機構であろう。ここまでの議論から、ネットワーク上のエージェントが協調分散処理を行なうためには、以下の機構が必要である。

- 共通知識
- ナビゲーション情報

これらの要求を統合する機構として、自己組織型情報カタログ[24]が提案されている。情報カタログはネットワークワイドのシソーラス機構であり、またリソースマップである。図5.6に自己組織型情報カタログの概要を示す。

情報カタログがシソーラスであることにより、各エージェント間で構造化された知識を共通知識として使える。また、ネットワーク上のリソースを探査する場合も、一定の知識体系に基づてネットワーク上のトラバースが可能になると同時に、エージェント間のタスクプランニング管理が容易になる。たとえば、図5.7のような概念階層が与えられて、"hardware"以下の概念を検索する場合、"storage"以下の概念を探査するエージェントと、"computer"以下の概念を探査するエージェントとに処理を並行分散化できる。

自己組織型情報カタログは以下のような機構から構成されている

- タスク・スケジューラ

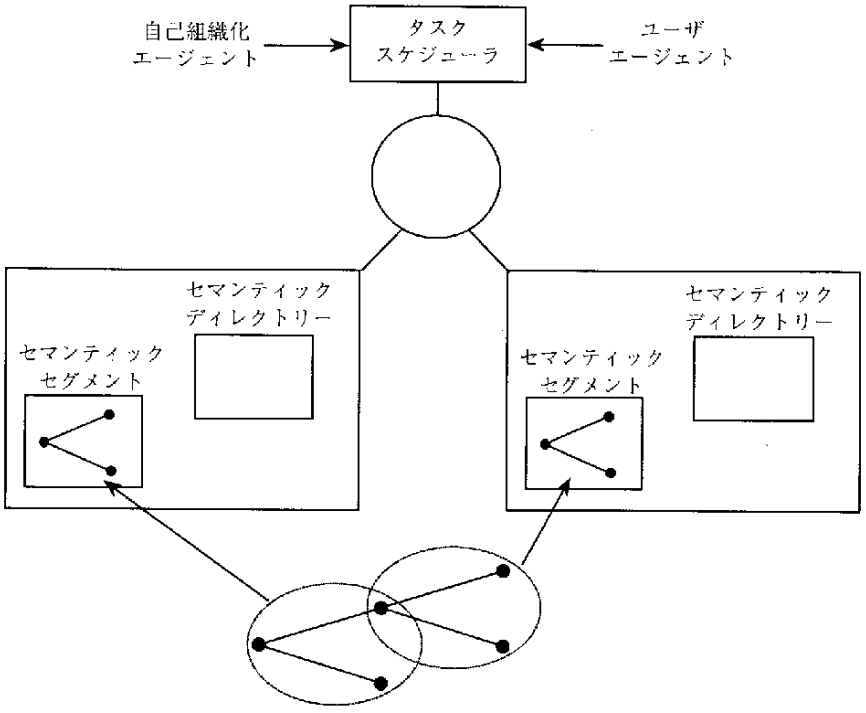

Figure 5.6: 情報カタログ

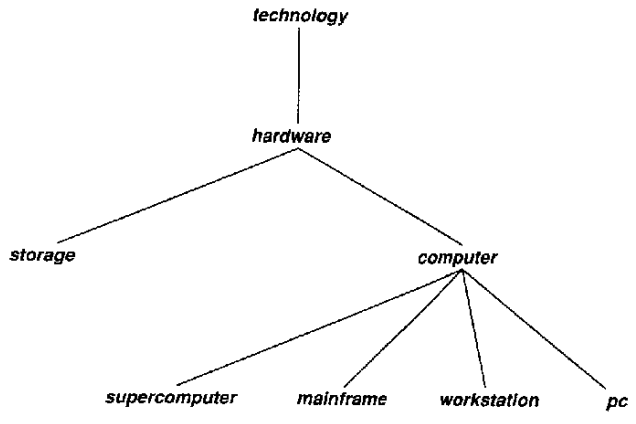

Figure 5.7: 概念階層の一例

- スケジューラから各セマンティックセグメントへの指示をブロードキャストする。
- クエリーとその結果をキャッシュする。
- セマンティックツリー全体を保持する。
- セマンティックツリーに関するあるレベル以下の処理は、その詳細情報を保持しているマスターノードにデリゲートする。
- セマンティックツリーやディレクトリ情報に変更があった場合、必要なノードへ変更をマルチキャストする。

- マスターノード（図5.8参照）
  - セマンティックセグメント
    部分的な概念構造（セマンティックツリー）を保持する。
  - セマンティックディレクトリー
    セマンティックツリー上のディスクリプタとそれに対応するURLとの対応関係を管理する。

- セマンティックツリー
  自己組織型情報カタログにおけるシソーラスであると同時にナビゲーション情報を管理する。

- クエリーエージェント
  ユーザのクエリー要求を処理するエージェント。

- 自己組織化エージェント
  - 仮想センサーにより状況判断
  - 自律的に経路選択したり、リソースのモニタを行なう。
  - もし与えられたいた状況と異なっていた場合、これをマスターノードやタスク・スケジューラに通知する。

自己組織化エージェントがネットワーク上をトラバースするアルゴリズムは以下のようなアルゴリズムである。

1. サブタスクに分割する。
2. エージェントに検索経路を示したサイト名リストを渡す。
3. 一定のワードマッチングアルゴリズムでランク付けを行なう。
4. エージェントはサイトリストに従ってトラバースする。

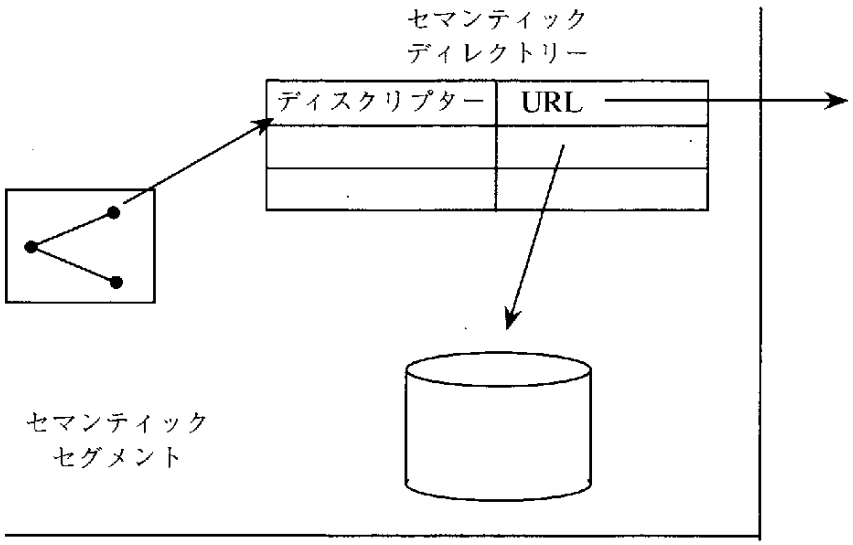

Figure 5.8: マスターノード

5. エラーが発生した場合もしくは他のリソースがあった場合、そのサイトへジャンプする。このジャンプは一度だけ、かつジャンプ先で評価を行なったら戻る。

6. もしトラバースの途中で変更があった場合、それをマスターノードやタスク・スケジューラに返す。

5.7 今後の課題

協調分散処理、あるいはエージェント型システムの今後の課題として、以下の各点をあげられる [26]。

1. エージェントの概念、理論、基盤技術の整理と充実

2. エージェントの知識と機能

3. 異種エージェントの協調メカニズム

4. 実時間、安全性、信頼性等への対応

これらの課題のうち、最初の2点は知識工学研究と深く関連している。また、後者の2点は分散協調処理特有の課題である。

ネットワークを渡り歩くエージェントの場合、そのエージェントにどの程度の能力や権限を持たせておくかが問題となる。これについては、各エージェントにプレース上のエージェントと相互作用する際に提示する身分証明書を与えるとともに、そのエー

ジェントがネットワーク上で実行できるコマンドや消費可能なリソースの寿命などを付加することにより、エージェントが無制限に振舞うことを回避できる。

PersonaLink サービス [16] ではテレスクリプト言語で記述されたエージェントがテレスクリプトネットワークを介して他のサイトに転送されて処理される場合、エージェントにはその目的地や処理に関する情報が添付され、ネットワークに送り出される。このとき、効率的かつ安全なエージェント転送を実現するためにテレスクリプトは暗号化され、専用の情報転送形式にコード化される。到着先のサイトでは受け取ったプログラムをデコードし、テレスクリプトエンジンによてエージェントの解釈実行がインタラクティブに行なわれる。こうしたサイトにおけるインタプリタ形式のエージェント処理は送られたエージェントが他のサイトに対してセキュリティ侵害を行なうことを防ぐ手段となっている。

6 マルチモダリティデータ処理

本章では非文字情報コンテンツ提供サーバーの形態を比較し、非文字情報に対するインデックシング、および検索形態に関する現状と、今後の展望について述べる。

6.1 マルチメディア

現在、マルチメディア対応を唱ったプロダクトが多数製品化されている。それらのプロダクトの多くはテキストだけではなく次のようなメディアのうち、いくつかを組み合わせたものが含まれている。

- 音声
- グラフィック
- イメージ
- 動画

マルチメディアを支える業界標準、国際/国内標準として以下のような標準がある。今日のマルチメディアに関する最も大きなニュースはこのような標準策定団体の設立、あるいは標準化策定団体間の "陣とり" 合戦に関するものがほとんどである。

- CD-I
- DVI
- Indeo
- JPEG
- MIDI
- MPEG
- Photo CD

このような状況の中で、“マルチメディア” 市場はハードウェア、あるいはそれらハードウエアを利用するドライバーのような基本ソフトウェア主体に進んでいる。現在、市場に販売されているマルチメディア応用ソフトウェアプロダクトは以下のような製品が主流である。

- ゲーム
- 教育用ソフト
- 子ども向け製品
- 写真集
- マルチメディアドキュメント作成ツール

教育ソフトやその他エンターテェイメント関連のソフトウェアは CD–ROM の普及に伴ってその市場を拡大している。しかしそれらの大半が、VTR や音楽 CD 製品の焼き直しの範囲を出ておらず、コンピュータでマルチメディア処理を行なう必然性に欠けるものが多い。

日常的な業務を支援するアプリケーションで、マルチメディア処理に最も近いアプリケーションは DTP 支援ソフトウェアであろう。

6.2 マルチメディア文書作成支援ツール

マルチメディア関連アプリケーションの中で、我々の日常業務にとってもっとも一般的なツールは DTP ツールとである。DTP ツールは以下のような機能を備えた “高級” ワープロとも言える。

- タグベースフォーマット
- ビルトイン・ワープロ機能
- ドローツール
- 各種フォーマットのグラフィック、テキストデータのインポート/エクスポート機能
- テンプレート機能
- 目次や索引の自動作成機能
- 各種引用物や DTP データの版管理、一貫性管理機能

現在出荷されている DTP ツールの大半で取り扱われるメディアは以下のようなメディアである。

- 文字（テキスト）
- イメージ
- ベクトル図形
- グラフや表

これらのメディアのうち、DTP ツールで処理されるメディアはテキストで表現される情報が大半である。その他のメディアは他のアプリケーションで作成されたものを、DTP ツールで作成されたドキュメントに “張り込む” ことによってドキュメント内部に取り込まれる。

DTP ツールが専ら行なうことは、様々なメディアで表現されたオブジェクト（テキスト、グラフィック、イメージなど）をいかに見栄えよく配置するかに関する処理、すなわちレイアウト処理である。極論すれば、DTP ツールはドキュメントのもつ表示的構造のみを処理し、その論理構造に関して何らの言及もしない。

DTP ツールや DTP ツールと等価な機能を備えたワードプロセッサが増加するにつれて、それらの再利用に対する要求が高まって来ている。その要求を背景として、マルチメディアに対応した情報管理システムが製品化されつつある。

このようなツールは DTP ツールよりも意味処理に一歩ふみ入れたツールといえよう。以下の各節でマルチメディア対応の情報管理システム処理の概要を述べる。

6.2.1 Compel

MS–Windows[1]ベースの表示用プログラムであり、映像/音声/動画などのリンクの作成、編集、制御を行なう。これらの機能により、ユーザは対話型ハイパーテキスト的なプレゼンテーションが行なえる。Compel[2]はスライドのような一般的な表示ツールを使い、画像のアウトラインを表示する。

音声や画像、動画などの表示要素は他の表示要素とリンクされる。このリンクは、スライドを差し込んだ、マウスをクリックした、あるスライドにリンクしたなどの動作に応じて再生/停止できるように組み込まれる。

6.2.2 HSC Inter Active

HSC InterActive[3]は AimTech 社の IconAuther[4]の個人向けバージョンであり、プレゼンテーション用パッケージとオーサリング・システムの中間的役割を満たす。

HSC InterActive でプレゼンテーションを作る場合、フローチャート形式のフォーマットの中にアイコンを組み込むことによって行なう。アイコンは、アイコンツールボックスの中からドラッグ・ドロップすることができる。このとき、アイコンにはそ

[1]MS–Windows は米国 Microsoft 社の登録商標である。
[2]Compel は米国 Asymetrix 社の登録商標である。
[3]HSC InterActive は米国 AimTech 社の登録商標である。
[4]InterActive は AimTech 社の登録商標である。

れぞれ詳細内容が割り当てられている。ここでユーザが定義するものは、ファイル名やフェードアウト、横にづらす、拡大するといった変化の特徴である。

Composite icons はアイコンの集合体であり、フローチャートの中に自動的に複合構造をつくるものである。Composite icons には以下のようなものが含まれている。

入力アイコンメニュー： ユーザが入力を行なうためのもの。

選択アイコン： 命令の分岐のためのもの。

ループ制御： メニューのためのフロー制御。

6.2.3 DECimagae

DEC 社の DECimage[5]は以下のような複数の製品の集合体である。

- DECimage Express[6]
  イメージを取り込むためのアプリケーションであり、多様なスキャナーでも使える。ファックス、E-mail アプリケーションやイメージなどを取り込める。

- DECimage Magadoc IV[7]
  ドキュメントマネージメントソフトウェアである。

- Image Now[8]
  既存のアプリケーションにイメージを付加でするツール。

6.2.4 FrameBuilder

FrameBuilder[9]はドキュメントの大きな目録、管理を目的にしたツールである。このプロダクトは次の 2 つのサブシステムから構成される。

- FrameBuilder Developer Edition
  ドキュメント主体の構造化されたアプリケーションを作成するために使われるツールである。このシステムによって、アプリケーション開発者は DTD 構造と階層化規則の編集/管理ができる。

- FrameBuilder
  FrameBuilder は SGML オーサリングツールであり、SGML タグ付きドキュメントの DTD を異なったプラットフォームで処理できる。また、FrameBuilder にはガイド付き編集機能があり、ユーザは WISIWIG なインタフェースを通じて構造化ドキュメントを作成できる。また、テキストツール、グラフィックツールに

[5]DECimage は米国 DEC 社の登録商標である。
[6]DECimage Express は米国 DEC 社の登録商標である。
[7]DECimage Magadoc IV は米国 DEC 社の登録商標である。
[8]Image Now は米国 DEC 社の登録商標である。
[9]FrameBuilder は米国 Frame Technology 社の登録商標である。

よって作成された非構造化データを構造化ドキュメントに変換する機能を提供している。

6.3 マルチメディア情報検索

マルチメディアオーサリングツールは各ドキュメントが内包する意味処理という点ではDTPよりも前進している。しかし、これらのツールにおける意味情報とは、異なったメディアをその生成時の同期関係、各メディアを提示する時の同時関係などメディア管理に関連するものである。

各データリソースのコンテンツの意味構造主体の処理はデータリソースの検索を行なう時に必要となる。マルチメディアデータの意味構造処理といった観点から、非テキストデータが主体となっているWebサイトを調査した。今回対象としたサイトは以下のサイトである。

- グラフィック主体：
  - Sandraのクリップアートサーバー[50]
  - Clipart Collectionsのクリップアート検索[8]
  - ワシントン大学の静止画像ライブラリ・インデックス[57]
- サウンド主体：
  - ワシントン大学の音声ライブラリ・インデックス[58]
  - ミシガン州立大学の音声データライブラリー[33]

サイト内に蓄積されている各データには、それぞれデータリソースを代表するようなインデックスが付けられそのインデックスにそって分類されている。これらのサイトにおける検索は、テキストベースによる検索が主体となっている。

インデックス主体の検索方式に加え、さらにイメージの内容を表すアイコンを検索の補助手段として用いているサイトもある。検索の観点から見た場合、このようなナビゲーション方式の問題点として以下の各点が挙げられよう。

- テーマを特定しているサイトやページではない場合、インデックスが表す内容からそのコンテンツを特定しづらい。
- 逆に、特定のテーマ関するデータを集中的に集めたサイトの場合、専門知識画必要になる。
- 動画のように複数の話題を同一リソースを持つ場合、その内容を特定しずらい。
- 人手によるメンテナンスを行なっているため、データ更新のタイムラグが大きい。

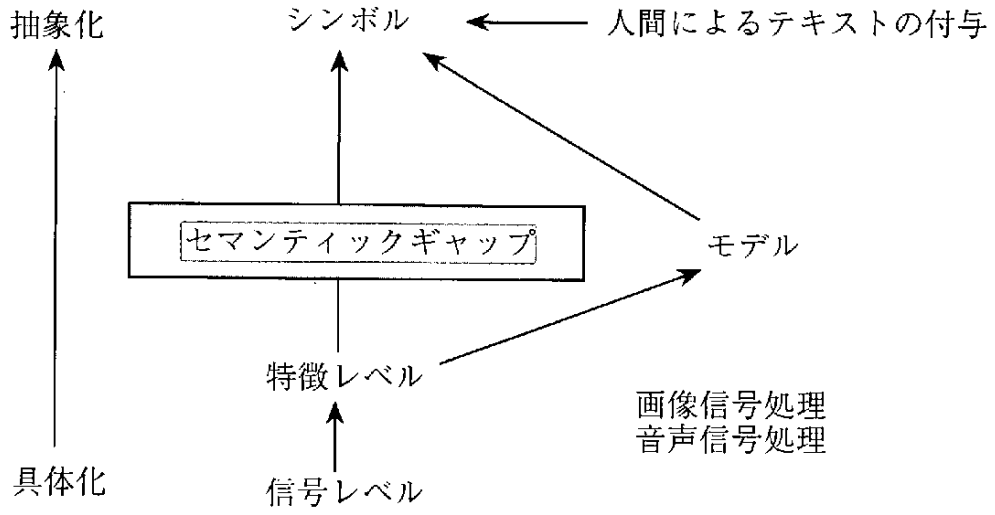

Figure 6.1: 情報抽出軸

6.4 マルチモダリティ処理

ハードウェアの低価格化やWWWの普及に伴い、テキスト情報と非テキスト情報とを複合化した情報の蓄積や検索に対する要求は高まってくるであろう。

しかし、検索や検索支援技術そしてデータソースの自動インデック作成技術の成熟度は低い。その一方で既存のパターン認識技術や知識処理技術でこれらの課題をブレークスルーするような技術についても明確な展望が得られないのが現状である。この問題の根源は図6.1[34]に示すようなセマンティックギャップを解消する技術が未熟であることである。

すなわち、これまでの知識処理やテキスト処理は記号としての記述レベルである。一方、パターン認識で扱われている情報は多重化した信号レベルの情報である。計算機が非テキスト情報から意味情報を抽出するためにはこの信号レベルの処理と記号レベルの処理の橋渡しを行なう“モデル”が必要となる[34]。

このモデルを考えるためには、非テキスト情報とこれまでに行なわれて来た非テキスト情報に対するインデックス付けに立ち戻って考察を行なう必要があろう。

6.4.1 マルチモダリティ

これまでに述べてきたようにマルチメディアという言葉にはさまざまな意味がある。マルチメディアデータとされるデータは音声、動画、静止画などがあるがデータ処理の方式やユーザの利用方式などを併せて考えると、数値データや表データなども非テキストデータすなわちメディアの一部とする必要があろう。

さまざまなメディアで表現されるデータは、その根本となる情報（あるいは意味）という点では同じレベルのデータであるが、その表現される様式（モダリティ）が異なる

```
\begin{figure}
  \begin{center}
    \leavevmode
    \psbox[scale=1.]{image-web}
  \end{center}
  \caption{WEB における非テキストデータの検索}
  \label{FIG:IMAGE-WEB}
\end{figure}
```

Figure 6.2: TeX 文書におけるグラフィックの定義例

もとして位置付けられる。テキスト情報はそれが表現されるモダリティと、コンピュータ上で使用されるモダリティが等しいために検索を始めとするコンピュータ処理を行ないやすい。

このように考えると、先に述べたセマンティックギャップを解消する手段として以下の2つの方式が考えられる。

1. あるモダリティから直接意味情報を抽出する技法の開発する。

2. 同等の情報を異なったモダリティで表現し、それを処理する。

現状の基礎技術を考えると、後者のアプローチの方がより現実的な解を出しやすい。本報告ではまず、後者のアプローチに沿った考察を行ない、その後で前者のアプローチの可能性について考察を行なう。

意味表現を行なう上で、あるモダリティの代替として最もよく用いられるのはテキストである。たとえば本報告の前節の図は図6.2に示すようなLaTeXのコードで構成されている。ここで、グラフィック（EPSFフォーマットで表現されている）本体は\psbox以下の節で表現されている。また、\captionの行で表現されている情報はこの図の脚注を、そして\label行では本文中の参照ラベルを表現している。

このスキームで文中のグラフィック情報を、グラフィックとしてのモダリティではなくテキストというモダリティを使って表現している。

6.4.2 マルチモダリティ検索

複数の異なったモダリティを統合してデータベースに格納する際、概略的には図6.3に示すような枠組となろう[34]。

この形態をとるシステムは、大きく以下の4つのモジュールに分けられる。

1. マルチモダリティの入力/構造化

2. データ蓄積機構

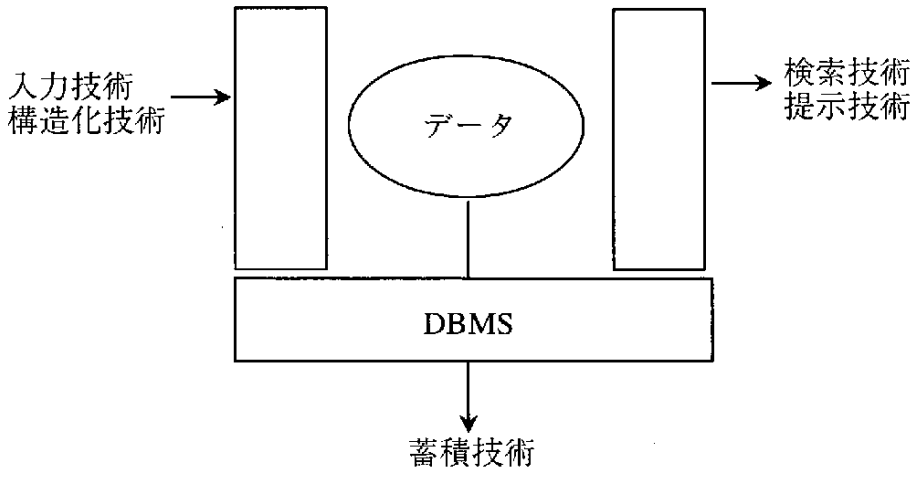

Figure 6.3: マルチメディアデータベースの枠組

3. データ検索/表示技術

4. DBMS

ここで、DBMS が行なう機能はデータの保全、トランザクションの管理などである。近年、マルチメディア対応を唱った DBMS が製品化されているがそのほとんどが VLBO (Very Large Binary Object) サポートに留まっている。マルチメディアデータベースでは、特にその表示/検索系が大きな意味をもつ。

この検索系について、モダリティを並列させた場合の検索について考える。テキストによる検索を図示すると、図 6.4のようになる。テキストを介した検索では、非テキストモダリティで表象されたデータとそのインデックスや意味記述子が同等の意味を持つことが仮定される。この方法は、リモートセンシングデータや化学物質データのように、データ採取に関するモデルや領域が明確な場合、有効な手法となる。

このようなデータの検索は基本的には、既存のデータベースに対する問い合わせ処理と同じ枠組となる。一方、"赤い服をきた女の子が写っている写真" といったようなモダイリティを直接記述した検索条件指定に対する要求も強い。

この方式の問題点とし以下の各点があげられる。

- 適切なインデックシングが難しい場合が多い。
- 適切な検索条件指定が行なえない。
- メンテナンスやインデックスの付与の工数がかかる。

モダリティに依存したアプローチとは、モダリティに特殊な特徴量を用いて検索を行なうアプローチである。このアプローチの枠組を図示すると図 6.5のようになる。

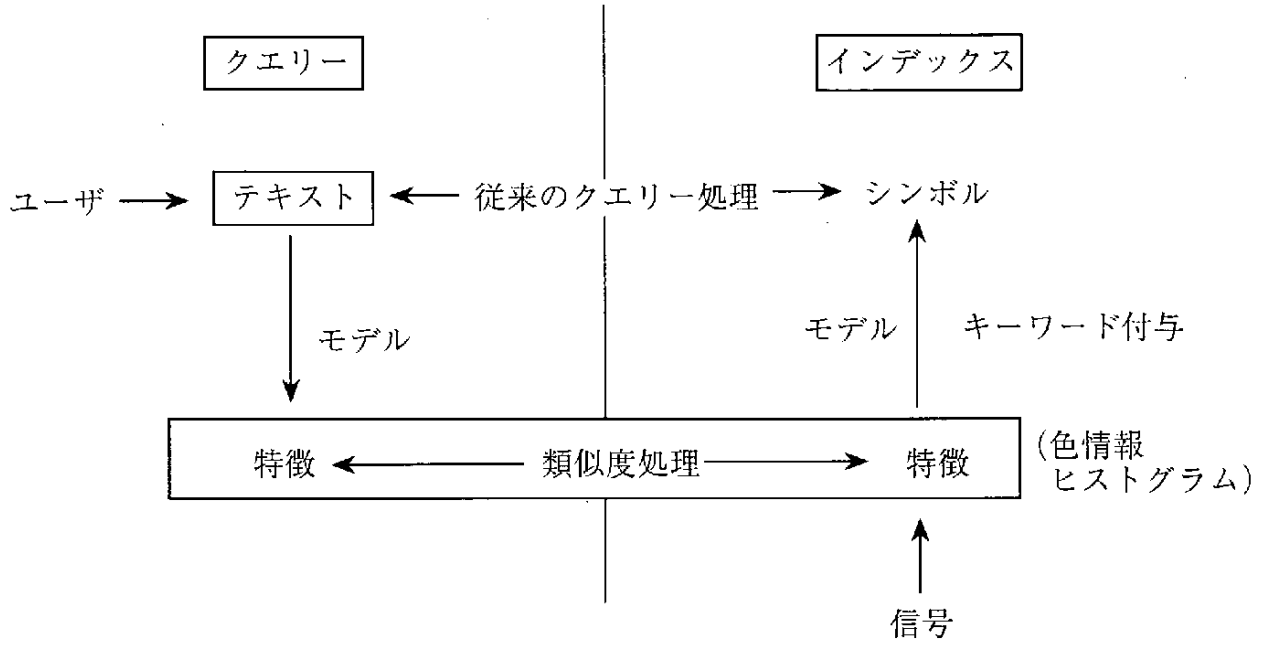

Figure 6.4: テキストによる検索

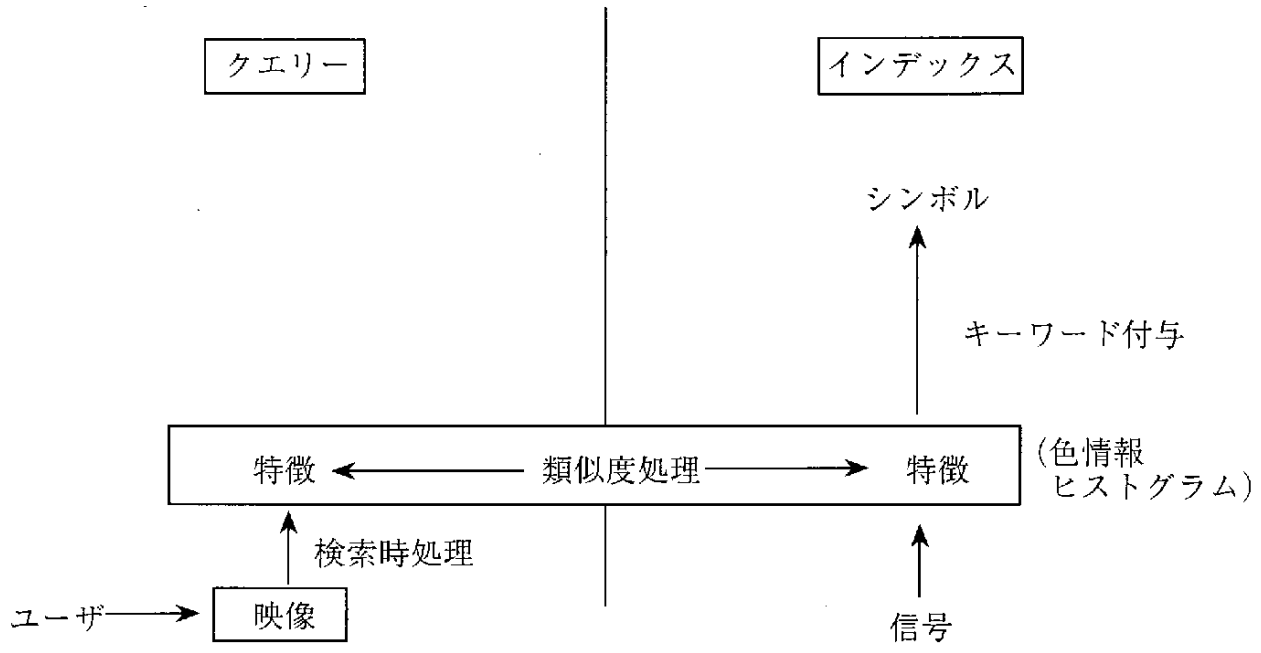

Figure 6.5: 映像による検索

ここでの特徴量とは静止画像データの場合、色情報、濃度ヒストグラムそしてテクスチャ情報などが挙げられる。また、音声においてはフォルマントやVOT（Voice On-set Time）、声紋情報等があげられる。

このようなモダリティに依存する特徴量を積極的に適用した例として、スポーツのシーンを色の組み合わせで表すモデルを作成し、それによってサッカーや相撲の映像を検索するシステムの研究が行なわれている[61]。また、指紋の照合システムのように領域に特殊な特徴量を用いて検索を行なえるシステムの開発および実用化も進められている。

モダリティに特殊な特徴量を用いて検索を行なうアプローチは、モダイリティとデータの対象領域にうまく適合するものが見つかれば、強力な方式となる。しかし、現状では研究途上のシステムが大半であり、処理可能なデータもかなり限定されたものである。

また、モダリティに特殊な検索ではあるモダリティ特殊な処理を検索プロセスに組み込むことも可能である。例えば静止画像の場合、量子化、標本化、アンチエリアシングそして、画像の拡大/縮小などが考えられる。このような処理を検索プロセスに組み込むことは検索そのものの効率にはあまり寄与しないであろう。しかし、実際の検索やその後のデータ処理を行なう上での利便性を提供できる。

6.5 マルチモダリティデータの構造化

動画像は静止画像とは違って図6.6に示すように一定のストーリーを持っている。このストーリーはさらにいくつかのサブストーリーに分割でき、さらにこのサブストーリーもいくつかの構成要素に分割できる。動画像は一般的に階層構造をしており、木構造で表現できる[52]。

これはテキストで表現された情報には文書→段落→文→単語といった階層が存在するのと似ている。非言語的データにおいても適切な単位に分節化することが、より深い処理を行なうための出発点となる。

ニュース画像を対象にしてモダリティ間の並行性を利用して構造を抽出する方式の研究が行なわれている[62]。この研究では、ニュース画像と分節化にはこれまでのカット検出と同様な方法を用いているが、検出されたカットに対して意味情報を付与するためにニュース原稿を用いている。この手法のようなモダリティ間の並行性を利用した手法は、たとえば文字多重放送のように類似したデータに対して適用可能であろう。

さらに、この手法は動画だけではなくHTMLドキュメント上のクリッカブル・マップ（clickable map）にも同様な手法が適用できよう[25]。クリッカブル・マップはHTMLドキュメントに表示されたイメージ図形の一部を指定し、その部分と他のHTMLリソースとの間にハイパーリンクが張られている。このようなクリッカブルマップをインターネット上のサイトから収集することよりセグメント化されたイメージの断片を収集できる。このグラフィックの断片とハイパーリンク先の情報を収集することにより、あるイメージとそれに付随した意味情報を統合的に処理できる（図6.7参照）。この手法は単にHTMLドキュメントだけではなく、他のハイパーテキストにも応用できる。

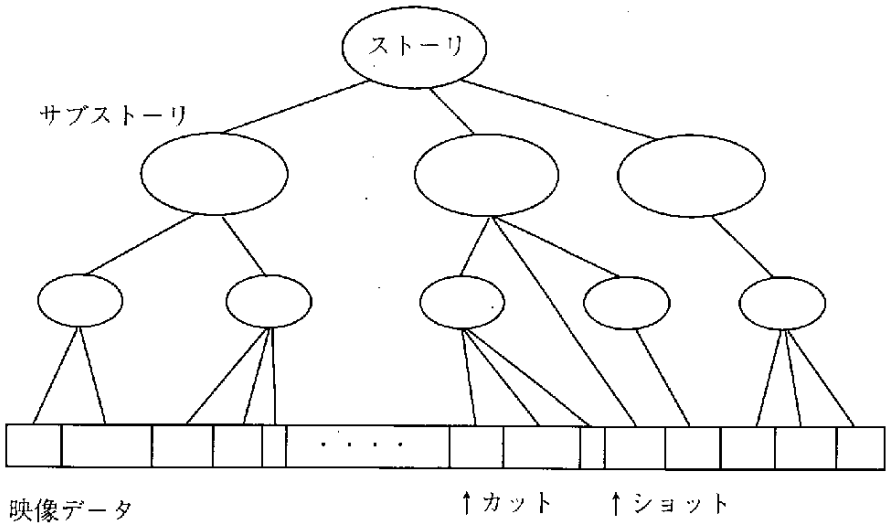

Figure 6.6: 映像データの階層

この方式のポイントは、その状況依存性を無視すれば情報の根源的意味は一つであるが表象形態すなわちモダリティが異なるデータがマルチメディアとして定義できる点である。つまり、マルチメディアとはモダリティが並行したデータであるといえる。

モダリティの並行性を生かした処理を行なうためには、それにあった情報ソースを見つけることと並んで、各モダリティ間のレベル合わせが必要となる。同一の処理単位として並行するモダリティがを扱えるためには、各モダリティ内部での抽象化が必要であり、逆にその抽象化操作を行なうことにより同一単位としての対応づけが可能となる。これは、各モダリティ内の他段階処理とモダリティ間の同期化が必要であることを意味している [25]。モダリティごとに表示や処理の特殊性あるために、同一要素として処理を行なうためにはこのような特殊性をカバーする必要がある（図 6.8参照）。

このれらのことから、マルチモダリティの意味処理を行なうためには、以下のような機構が必要となる。

- **多戦略–多段階処理**
  モダリティの差異を吸収し意味的等価性を確認するためには、元データからシンボルへと至る処理をフェーズに分割して処理を行なう。各フェーズ単位で等価性を確認する方が各モダリティ間のセマンティックギャップが小さくなる。また、意味的な変位に対応するために複数の問題解決戦略をとれる、すなわち多戦略であることが望まれる。

- **モダリティ間の同期化**
  多段階処理の各フェーズ間あるいはフェーズ内での各ステップで、各モダリティにおける処理結果の同期をとる必要がある。この同期とは、各出力結果間での意味的な等価性を確認し、もしユーザの意図と異なってた結果が得られていた場合、

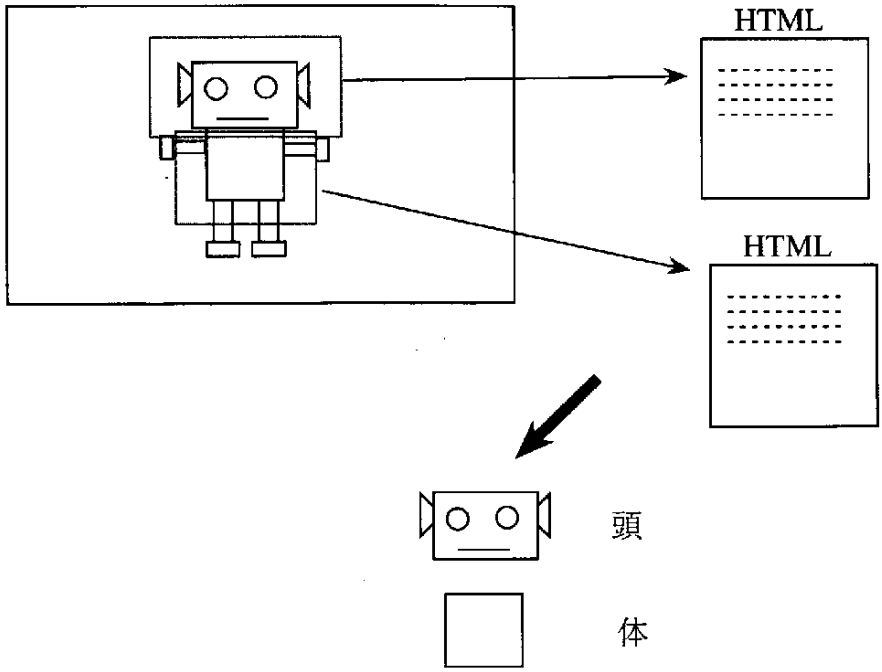

Figure 6.7: クリッカブル・マップ

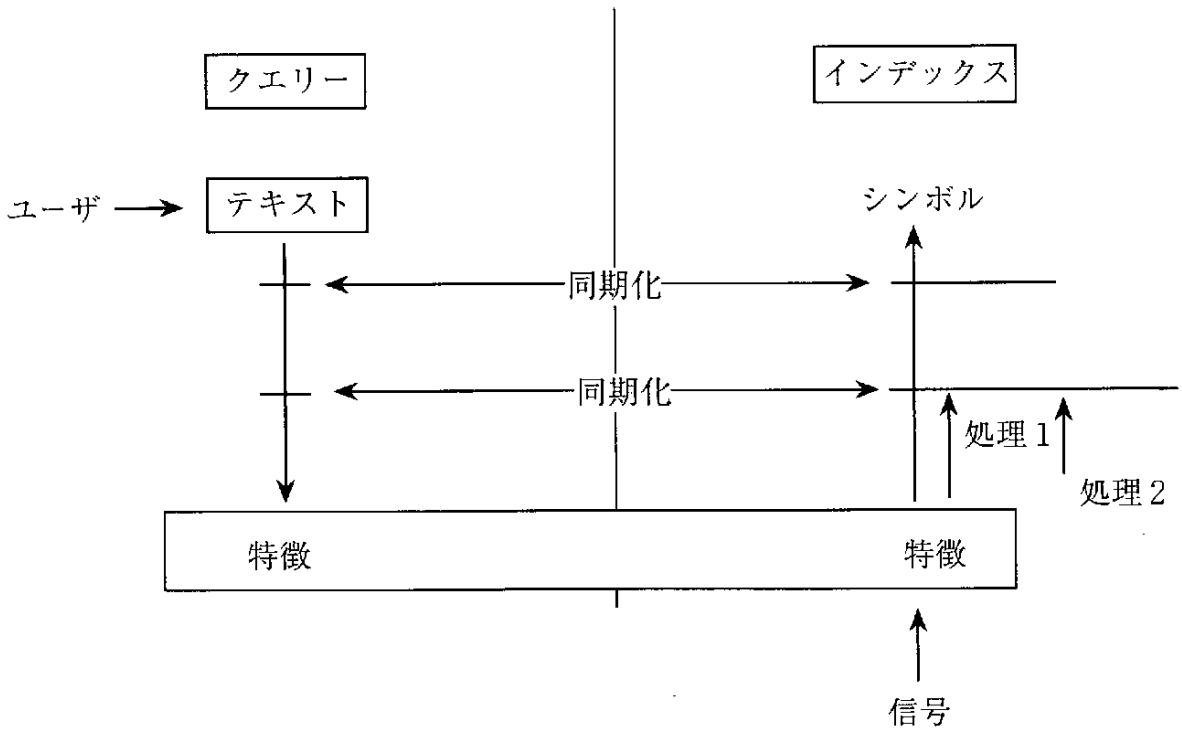

Figure 6.8: 多段階モダリティ処理

その調整を行なうことを意味している。ここでの意味とは元データを始点とし、シンボルを終端とした軸上での評価である。

GLSのスキーム考えると、このようなプロセスは、プリプロセス、構造抽出、管理/精密化という3フェーズに分けられ、これらの各フェーズで多面的データ分析や概念抽象化を行なう。

モダリティの特殊性を考慮し、かつ意味的に等価な複数のモダリティ間の同期をとるためにも、この処理スキームは有効である。意味情報の等価性に着目した、マルティモダリティデータの多段階処理は以下のようにモデル化できる[25]。

1. **プリプロセス**
   このフェーズでの重要な処理はユーザとの対話に基づいてユーザの要求を収集し、使用したい構造化情報抽出法を確定し、テキストデータの収集と整理等を行う。例えば静止画像の場合、

   - 色情報処理
   - 濃度ヒストグラムの作成
   - テクスチャ情報解析
   - 量子化
   - 標本化
   - アンチエリアシング
   - 画像の拡大/縮小

   などの処理を行う。

2. **構造抽出**
   構造抽出フェーズではプリプロセスフェーズの処理に基づいて構造抽出を行う。このフェーズではエッジ検出、プリミティブの処理、統計的、および確率的な手法、ニューロネット技術などの方法を利用しクラスタを生成する。

3. **管理/精密化**
   管理/精密化のフェーズでは前フェーズで抽出した基本的プリミティブやプリミティブ間の類似性と知識ベース中の知識とを照合し、さらに大きな図形単位への合成などを行なう。この画像上のプリミティブ合成過程は、そのプリミティブに対応する概念の合成となる。ゆえに、あるプリミティブ同士の矛盾性のチェックを、それに対応する概念の組合せにおける矛盾のチェックとできる。

この三つのフェーズ処理によって、モダリティ間の並行性を生かした多面的データ分析、多段階の概念抽象化そして意味情報の抽出が可能になる。さらに、ユーザの要求による構造化だけでなく、動的な発見プロセスの組織化、発見プロセスの制御と性能改善なども行う。このプロセスから、マルチモダリティデータ処理に関するメタ知識も抽出できよう。

6.6 複合ドキュメント処理

これまでに述べて来たようなマルチモダリティデータを処理するプログラムは各モダリティに特異な処理が含まれる。このような処理を含んだシステムは必然的に複雑な構造となり、大規模なシステム開発にとって大きな阻害要因となる。

その一方で、DTP の普及あるいはワープロの DTP 化に代表されるように、今後のドキュメントの大半は複数のモダリティを含んだものとなろう。

オブジェクト指向技術をこのような複合モダリティドキュメントに適用して、統合的に処理/管理を行なうアーキテクチャとして複合ドキュメントアーキテクチャが提唱されている。

今後のマルチモダリティデータ処理を考える上で、この複合ドキュメントアーキテクチャはインフラ技術の一つとなるであろう。また、これまでに述べてきた各処理を行なう上でも、ドキュメント構成要素のオブジェクト化は歓迎されるものである。

複合ドキュメントは他のアプリケーションを含む様々なソースから生じるデータを入れる“コンテナ”の集合体である。コンテナには個別のデータ要素に関連付けられた外部アプリケーション用のフックが用意されている。別の見方をすれば、複合ドキュメントは複数の子となったアプリケーションによって作成されたデータをドキュメント中心のインタフェースで表示/編集を行なえる機構であるといえる。つまり、複合ドキュメントはドキュメントを管理するアプリケーションがそのドキュメント内のオブジェクトを所有するアプリケーションと通信できるプロトコルおよびその実行環境であるといえよう。

複合ドキュメントには事実上、OLE2[10]と OpenDoc[11]といった 2 つの標準が存在する。以下の各節でその概要を紹介する。

6.6.1 OLE2

1990 年に Microsoft 社は DDE ベースのオブジェクトのリンクと埋め込み技術として OLE を発表した。さらに 1993 年に COM（複合オブジェクトモデル）と呼ぶオブジェクトカプセル化技術を用いた OLE2 を発表した。

現在の OLE2 の構成を図示すると図 6.9のようになる。

OLE2 の各構成要素は以下の機能を持つ。

- COM
  COM は一つのアプリケーション内または複数のアプリケーション間のオブジェクトインタフェースである。COM はローカルな RPC 機能のみをサポートしており、リモートメッソド呼出や分散オブジェクト機能はサポートしていない。また、継承やポリモーフィズムもサポートされていないので、そのクラスモデルの機能は限定的なものとなっている。

[10]OLE は米国 Microsoft 社の登録商標である。
[11]OpenDoc は米国アップルコンピュータ社の登録商標である。

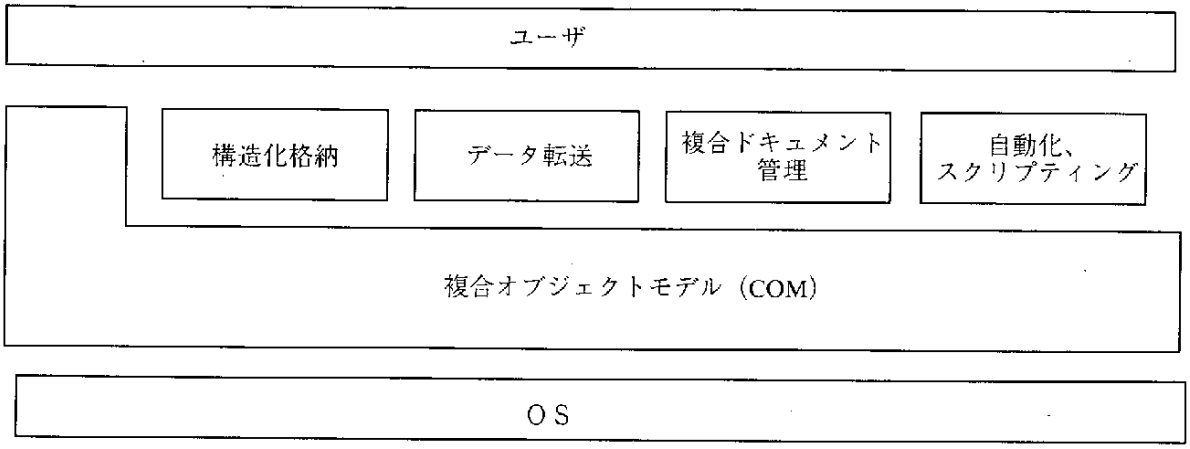

Figure 6.9: OLE2

- **構造化格納**
  ファイル内にファイルシステムを提供する機構である。この機構により、ドキュメント内の構造を木構造を使って表現できる。

- **同一データ転送モデル**
  ドロップ&ドラッグ、コピー&ペーストなどを使ってデータを同一的に転送するための機構である。

- **複合ドキュメント管理**
  ビットマップや音声など異なる形式のデータをシームレスに統合するコンテナアプリケーション内にドキュメントとして実装されている管理機構である。

- **自動化・スクリプティング**
  ツールやスクリプト言語からシステムマクロを作成するためのCOMインタフェースである。

6.6.2 OpenDoc

OpenDoc は Apple 社、IBM 社、Novel 社、Oracle 社、Taligent 社、SunSoft 社、WordPerfect 社そして Xerox 社が共同で設立した Component Integration Laboratories（CI Labs）による複合ドキュメントのアーキテクチャである。

現在の OpenDoc の構成を図示すると図 6.10のようになる。

OpenDoc の各構成要素は以下の機能を持つ。

- SOM と DSOM
  SOM は単一アドレス空間において、または単一マシン上の複数アドレス空間にまたがって通信するオブジェクトのための CORBA 準拠のプロトコルである。DSOM により SOM はネットワークをまたがって通信できる。

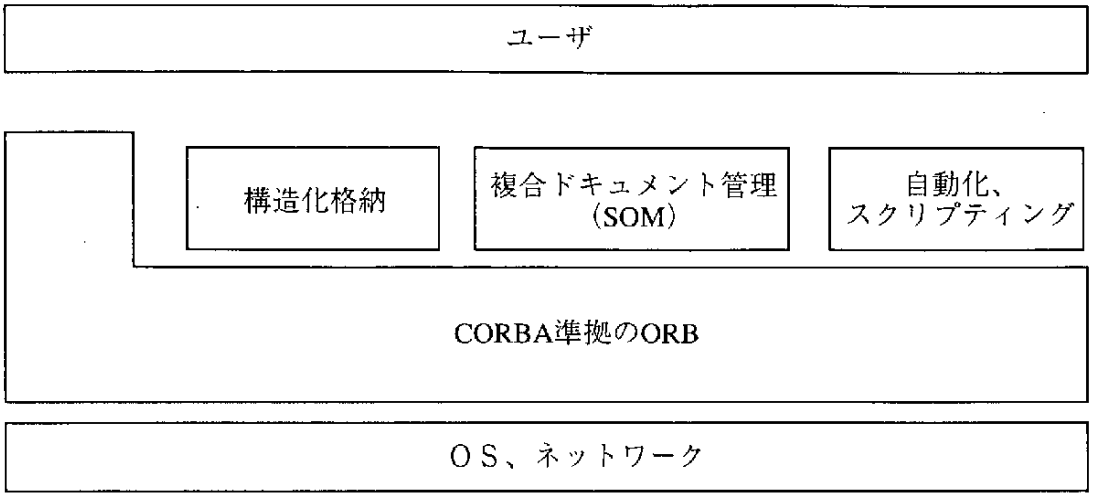

Figure 6.10: OpenDoc

- Bento オブジェクトコンテナ
  構造化ファイルシステムを提供する機構でる。各ファイルには多くのストリームが含まれ、各コンテナには複数のドキュメントオブジェクトを入れることができる。また、各オブジェクトには1つまたは複数のドラフトオブジェクトを入れることができる。

- OpenDoc 複合ドキュメント管理
  部品を挿入できるコンテナを提供する機構である。

- オープンスクリプティングアーキテクチャ（OSA）
  Apple 社の AppleScript をモデルにしたオブジェクト指向型のスクリプティング機構である。

Bibliography

[1] H. AÏt-kaci. An algebraic semantic approach to the effective resolution of type equations. *Theoretical Computer Science*, Vol. 45, pp. 293–351, 1986.

[2] V. Bush. As we may think. *Atlantic Monthly*, Vol. 176, No. 1, 1945.

[3] H. Chen. Collaborative systems: Solving the vocabulary problem. *IEEE Computer*, Vol. 27, No. 5, pp. 58–66, 1994.

[4] H. Chen. Machine learning for information retrieval : Neural networks, symbolic learning, and genetic algotithms. *Journal of the American Society for Information Science*, Vol. 46, No. 3, pp. 194–216, 1995.

[5] H. Chen, et al. Generating, integrating, and activating thesauri for concept-based document retrieval. *IEEE Expert*, Vol. 8, No. 2, pp. 25–34, 1993.

[6] H. Chen, P. Hsu, et al. Automatic concept classification of text. *Comm. of the ACM*, Vol. 37, No. 10, pp. 56–73, 1994.

[7] H. Chen and K.J. Lynch. Characterizing document databases ieee transactions on systems. *Man and Cybernetics*, Vol. 22, No. 5, pp. 885–902, 1992.

[8] http://leviathan.tamu.edu:70/7c/clipart/.cache.

[9] W.W. Cohen. Text categorization and relational learning. In *Proc. 12th International Conference on Machine Learning (ML-95)*, pp. 124–132, 1995.

[10] G. DeJong and R. Mooney. Explanation–based learning: An alternative view. *Machine Learning*, Vol. 1, No. 2, pp. 145–176, 1986.

[11] D. T. Dewire. *Text Management.* McGraw–Hill, Inc., 1994.

[12] T. Doszkocs, J. Reffia, and X. Lin. Connectionist models and information retrival. *Annual Review of Information Science and Technology*, Vol. 25, , 1990.

[13] R. Feldman and I. Dagan. Knowledge discovery in textual databases (kdt). In *Proc. First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, pp. 112–117, 1995.

[14] Y. Fujiwara, et al. The information base system for materials research. *CODATA Bulletin*, Vol. 24, No. 1, pp. 1–7, 1990.

[15] L. Gasser. *An Overview of DAI.* Kluwer Pub., 1992.

[16] A. Gillon, et al. Supporting agent technology in a public network. In *Proc. of Seminar on Agent Software*, 1995.

[17] M. Gordon. Probabilistic and genetic algorithms for document retrieval. *Comm. of the ACM*, Vol. 31, No. 10, pp. 1208–1218, 1994.

[18] R. Gray, et al. Transportable information agents. Technical Report PCS–TR96–278, Dartmouth Universty, 1996.

[19] T. R. Gruber. Ontolingua: A mechanism to support portable ontologies version 3.0. Technical Report Technical Report, Konwledge System Laborarory, Stanford Universty, 1992.

[20] J. Han, Y. Cai, and N. Cercone. Data-driven discovery of quantitative rules in relational datebase. *IEEE Trans. Knowl. Data Eng.*, Vol. 5, No. 1, pp. 29–40, 1993.

[21] J.H. Holland. *Adaptation in Natural and Artificial Systems.* Univ. of Michigan Press, 1975.

[22] Y. Kakemoto. Inductive reasoning using thesaurus. In *Proc. of JAPAN–CIS Conference on Knowledge–base Software Engineering.*, 1994.

[23] Y. Kakemoto. Inductive reasoning using thesaurus. In *Proc. of Advece in DataBase and Information System (in Lecture Note in Computer Science)*, 1995.

[24] Y. Kakemoto and N. Zhong. The self organized information catalog Sapata : Its goal, architecture and current results, 1996. (submitted).

[25] Y. Kakemoto and N. Zhong. A study of self–organized image database, 1996. (submitted).

[26] 木下哲男, 菅原研次 *et al.* エージェント指向コンピューティング. ソフトリサーチセンター, 1995.

[27] T. Kohonen. Self-organizing map. *Proc. IEEE.*, Vol. 78, No. 9, 1990.

[28] A. López, et al. The development of category-based induction. *Child Development*, Vol. 63, pp. 1070–1090, 1992.

[29] P. Mase. Agents that reduce work and information overhead. *Comm. ACM*, Vol. 37, No. 7, 1994.

[30] P. Mase, et al. Learning interface agents. In *Proc. AAAI–93*, 1993.

[31] P. McCullagh and J.A. Nelder. *Generalized Linear Models.* Chapman and Hall, London, 1989.

[32] R.S. Michalski, J.G. Carbonell, and T.M. Mitchell. *Machine Learning - An Artificial Intelligence Approach*, Vol. 1. Morgan Kaufmann Publishers, 1983.

[33] http://web.msu.edu/vincent/index.html.

[34] 美濃導彦. 知的映像メディア検索技術の動向. 人工知能学会誌, Vol. 11, No. 1, pp. 3–9, 1996.

[35] M. Mozer. Inductive information retrival using parallel distributed computation. Technical Report TR-ICS 8460, Univ. of Calif., San Diago, 1984.

[36] S. Muggleton and C. Feng. *Inductive Logic Programming.* Academic Press, 1989.

[37] 仁木和久. ニューラルネットワーク技術の情報検索への適用. 人工知能学会誌, Vol. 10, No. 1, 1995.

[38] 西田豊明. ソフトウェアエージェント. 人工知能学会誌, Vol. 10, No. 5, 1995.

[39] 大沢英一. 情報処理振興事業協会ネットワークエージェントプロジェクト WG 委員会資料, 1995.

[40] S. Ohsuga. Framework of knowledge based systems - multiple meta-level architecture for representing problems and problem solving processes. *Knowledge Based Systems*, Vol. 3, No. 4, pp. 204–214, 1990.

[41] S. Ohsuga and H. Yamauchi. Multi–layer logic – a predicate logic including data structure as knowledge representation language. *New Generation Computing*, Vol. 3, No. 4, pp. 403–439, 1985.

[42] J. K. Ousterhout. *Tcl and Tk Toolkit.* Addison–Wesley, 1994.

[43] G. Piatetsky-Shapiro and W.J. Frawley. *Knowledge Discovery in Databases.* AAAI/MIT Press, 1991.

[44] G. Piatetsky-Shapiro and C.J. Matheus. Knowledge discovery workbench for exploring business databases. *Inter. J. of Intell. Sys*, Vol. 7, No. 7, pp. 675–689, 1992.

[45] J.R. Quinlan. Induction of decision trees. *Machine Learning*, Vol. 1, No. 1, pp. 81–106, 1986.

[46] J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, Vol. 5, No. 3, 1990.

[47] J.R. Quinlan and R.M. Cameron-Jones. FOIL:a midterm report. In *ECML-93*, 1993.

[48] G. Salton. *Automatic Text Processing*. Morgan Kaufman Publisher, INC., 1988.

[49] サンマイクロシステムズ株式会社. Java 言語環境, 1995.

[50] http://www.cs.yale.edu/HTML/YALE/CS/HyPlans/loosemore-sandra/clipart.html.

[51] J.W. Shavlik and T.G. Dietterich. *Readings in Machine Learning*. Morgan Kaufman Publisher, INC., 1990.

[52] 柴田正啓. 映像の内容記述モデルとその映像構造化への応用. 電子情報通信学会論文誌, Vol. 78–D–II, No. 5, pp. 62–72, 1995.

[53] H.A. Simon and A. Ando. Aggregation of variables in dynamic systems. *Econometrica*, Vol. 29, pp. 111–138, 1961.

[54] 有田豊浦. 自己組織型ニューラルネットワークによるドキュメントの自動分類. 情報処理学会自然言語処理研究会資料, Vol. 92, No. 21, 1992.

[55] T. Tunoda and H. Tanaka. Analysis of scene identification ability of associative memory with pictorial directory. *Proc. COLING-94*, Vol. 1, , 1994.

[56] Ch. von der Malsburg. Self-organization of orientation senstive cells in the. *Kybernetik*, Vol. 14, , 1973.

[57] http://wuarchive.wustl.edu/multimedia/images/.

[58] http://wuarchive.wustl.edu/multimedia/audio/internet-talk-radio/.

[59] T. Wittig. *ARCHON: An Architecture for Multi–agent System*. Ellis Horwood., 1992.

[60] M. J. Wooldridge. *Intelligent Agents*. Springer, 1995.

[61] J. Yamane and M. Sakauchi. A construction of a new image database system which realized fully automated image keyword extraction. *IEIC Trans. Inf. and Syst.*, Vol. E76–D, No. 10, pp. 10–28, 1993.

[62] 山瀬忠博. ニュース・オン・デマンドシステムにおける自動番組構成機構の設計と実装, 1995. 大阪大学工学部情報システム工学科特別研究報告発表会資料.

[63] H. Yamauchi and S. Ohsuga. Loose coupling of kaus with existing rdbmss. *Data and Knowledge Engineering*, Vol. 5, No. 4, pp. 227–251, 1990.

[64] N. Zhong and S. Ohsuga. GLS – a methodology for discovering knowledge from databases. In P.S. Glaeser and M.T.L. Millward, editors, *New Data Challenges in Our Information Age*, pp. A20–A30, 1992.

[65] N. Zhong and S. Ohsuga. A decomposition based induction model for discovering concept clusters from databases. In K.P. Jantke, et al., editors, *Proc. 4th International Workshop, ALT'93 (in Lecture Notes in Artificial Intelligence 744)*, pp. 384–397. Springer–Verlag, 1993.

[66] N. Zhong and S. Ohsuga. HML–an approach for managing/refining knowledge discovered from databases. In *Proc. 5th IEEE International Conference on Tools with Artificial Intelligence (TAI'93)*, pp. 418–426. IEEE Computer Society Press, 1993.

[67] N. Zhong and S. Ohsuga. An integrated calculation model for discovering functional relations from databases. In V. Marik, et al., editors, *Proc. 4th International Conference, DEXA'93 (Lecture Notes in Computer Science 720)*, pp. 213–220. Springer–Verlag, 1993.

[68] N. Zhong and S. Ohsuga. Attribute caculation in knowledge discovery in databases. *Journal of Japanese Society for Artificial Intelligence*, Vol. 9, No. 2, pp. 258–267, 1994.

[69] N. Zhong and S. Ohsuga. Discovering concept clusters by decomposing databases. *Data and Knowledge Engineering*, Vol. 2, No. 2, pp. 223–244, 1994.

[70] N. Zhong and S. Ohsuga. The GLS discovery system: Its goal, architecture and current results. In Z.W. Ras and M. Zemankova, editors, *Proc. 8th International Symposium, ISMIS'94(in Lecture Notes in Artificial Intelligence 869)*, pp. 233–244. Springer–Verlag, 1994.

[71] N. Zhong and S. Ohsuga. IIBR–a system for managing/refining structural characteristics discovered from databases. In *Proc. 6th IEEE International Conference on Tools with Artificial Intelligence (TAI'94)*, pp. 468–475. IEEE Computer Society Press, 1994.

[72] N. Zhong and S. Ohsuga. KOSI – an integrated system for discovering functional relations from databases. *Intelligent Information Systems*, Vol. 5, No. 1, pp. 25–50, 1995.

[73] N. Zhong and S. Ohsuga. Managing/refining structural characteristics discovered from databases. In *Proc. 28th Hawaii International Conference on System Sciences (HICSS-28)*, Vol. 3, pp. 283–292. IEEE Computer Society Press, 1995.

[74] N. Zhong and S. Ohsuga. Toward a multi-strategy and cooperative discovery system. In *Proc. First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, pp. 337–342. AAAI Press, 1995.

[75] N. Zhong and S. Ohsuga. A hierarchical model learning approach for refining/managing knowledge discovered from databases. In *Data and Knowledge Engineering*. Elsevier Science Publishers, 1996. (to be appear).

[76] N. Zhong and S. Ohsuga. A system for managing/refining structural characteristics discovered from databases. In *International Journal: Knowledge Based Systems*. Elsevier Science Publishers, 1996. (to be appear).

平成８年３月発行

発行　財団法人　データベース振興センター
東京都港区浜松町二丁目４番１号
世界貿易センタービル７階
TEL ０３－３４５９－８５８１
委託先　日本総合研究所
東京都千代田区一番町16番
TEL ０３－３２８８－４７６２
印刷所　(株)カントー
東京都千代田区九段北１－11－２
TEL ０３－３２３８－６０１１