デスクトップサーチにおける新しい検索モデルの提案

TAIZBRAを用いたデスクトップ上の異種データ検索

藤井　稚子†　　プラダン　スジット††

†† 倉敷芸術科学大学　〒 712–8505 岡山県倉敷市連島町西之浦 2640 番地
E-mail: †E09J078@kusa.ac.jp, ††sujeet@cs.kusa.ac.kp

あらまし　近年，計算機上のファイルを検索するためのソフトウェアとしてデスクトップサーチが注目されている．既存のデスクトップサーチの機能として，Web 検索と同じように，電子メール，Office 文書，XML や HTML 文書，LaTeX 文書，Web の履歴，画像や音楽などのデータをキーワードにより全文検索できる．しかし，問い合わせを行った際にファイルまたはフォルダ単位でしか検索できず，ファイル内容の一部分のみの検索が不可能である．また，複数のキーワードで検索を行った場合，既存のデスクトップサーチでは結果が返されないことがある．そこで本研究では，デスクトップサーチにおいて異種型データを統一的に管理，検索できるシームレスサーチの実現のために，データベース管理システムと情報検索で利用される既存の検索技術の統合を提案する．これにより，デスクトップサーチの検索精度の向上および利用者の利便性を図ることが可能となる．

キーワード　デスクトップサーチ，異種データ，シームレスサーチ，データモデル，代数的な検索モデル

Towards a New Desktop Search Tool

Retrieval of Heterogenous Desktop Data

Wakako FUJII† and Sujeet PRADHAN††

†† Kurashiki University of Science and the Arts
Kurashiki-shi, Tsurajima-cho, 2640 Nishinoura 712–8505 Japan
E-mail: †E09J078@kusa.ac.jp, ††sujeet@cs.kusa.ac.kp

Abstract　The most serious challenges Personal Information Management Systems face today are the results of having to deal with a large number of heterogeneous types of data from diverse data sources, but having no means of managing and searching them in a convenient, unified fashion. We argue that simplicity and flexibility are essential attributes for the next-generation search tools to respond to these challenges. This paper lays out specific issues to realizing such a tool in the context of desktop search and ties them to existing search techniques employed by Database Management Systems and Information Retrieval — the two leading disciplines in search technology. We believe that a combined database and information retrieval approach to searching heterogeneous data is going to benefit a large community of users. Finally, we describe our ongoing work on developing a flexible desktop search tool that takes advantages of this combined approach.

Key words　desktop search, data model, seamless search, tree algebra, heterogenous data

1. はじめに

デスクトップサーチは，Google が 2004 年に Google Desktop [6] を公開して以来，検索システム分野に多大な影響を与えた．今では OS に標準搭載されるようになり，情報検索の手段としてなくてはならない存在になっている．デスクトップサーチが必要となってきた理由として，HDD の大容量化に伴い計算機内に格納されるデータ量が増大したことが挙げられる．エクスプローラによるファイル管理の場合, 計算機内にある膨大なデータの格納場所を把握しておく必要があり，その中から探したいものを瞬時に見つけ出すのは困難であるが，格納場所を気にすることなく保存し，キーワード検索によってデータを素早く取り出せるデスクトップサーチは効率的かつ利便的である．このような背景から，今後ますますデスクトップサーチにより

データ管理が簡単明瞭に行える期待が高まっている．

現在，デスクトップサーチとして代表されるものに Google デスクトップ検索 [6]，Windows デスクトップサーチ [14] や Apple Spotlight [2] がある．これらは，インストール後インデックスを作成し，データベースを構築するため高速に目的のファイルを検索することが可能となる．これらのデスクトップサーチはフォルダ上で階層的に散らばる画像，音楽，電子メール，文章などの異種型データの検索を前提としている．従来のデータベース管理システム（DBMS）で管理される構造化データ，または非構造化データのテキストファイルなど様々である．しかし，そのようなデータの異種性を持つデスクトップ上の多様なデータソースに統一的に対応する手段がない．また，それを従来のリレーショナルデータベース (RDB) で行おうとすると，構造を持ったデータしか効率良く扱わないため相応しくない．このことについては近年注目を浴びていて，例えば [5] では SQL や XQuery のような問い合わせ言語の利用によるデスクトップや電子メールサーバー等の様々なデータソースに分散する個人データの管理，問い合わせや，グラフデータモデルでデータの表現を提案している．しかし，RDB で扱う SQL のような問い合わせ言語では，問い合わせを行うための知識や構造をユーザが把握しておく必要があり不便である．この方法は確かに高度な技術によってデスクトップ上の管理，検索が可能であるが，デスクトップ上の膨大なデータを容易に管理，検索するために Web 検索のような検索を望んでいると思われるデスクトップサーチユーザにとっては利便的ではない．使いやすいシステムであることがデスクトップサーチツールの必要条件であると考えられるので，これまでに親しまれてきたキーワード検索が最も望ましいと考える．

また，iMeMex プロジェクト [5] では，[6] のような既存のデスクトップサーチの問題点として，文章の中身を部分的に取り出せないことが述べられている．例えば，LaTeX ファイル内のある特定の部分のみを取り出そうとすると，検索範囲がオペレーティングシステム (OS) にあるファイルシステムに制限されるため，キーワードに一致するファイル名を探し，そのファイル全体が出力される．これではどの部分が問い合わせに適しているか探す必要があり，利用者の効率性を妨げる．ファイル構造を意識した解を結果として返す検索結果が望まれる．

そこで本研究では，デスクトップサーチの顕著な問題点であるファイルの構造を無視した異種型データの問い合わせの改善のため，検索範囲がファイルシステムかファイル中の部分かに関わりなく検索する方法を提案する．ファイルシステムとファイルの中身を全体的に統合し，シームレスな検索を実現させるためデスクトップにある異種型データを統一的に管理，検索するために従来のデータベース管理システムと既存の情報検索で利用される検索技術を統合した検索モデルを提案する．我々はそのために，異種型データを統一的に検索可能にする木構造に基づくデータモデルを提案する．また，このデータモデルに基づくデータベースが効果的，また効率良く問い合わせ処理を行えるようにこれまで提案してきた木構造データに用いられる代数を利用する．

以下 2. 章では，本研究を行うにあたる動機について述べ，3. 章では，木構造に基づくデータモデルについて説明し，4. 章でこれまでに提案してきた問い合わせモデルの概観について述べる．5. 章では，代数操作による最適化手法について述べ，6. 章では，これまでに提案してきた木構造における代数をデスクトップサーチに適用した例を示す．7. 章では，本研究に関連した研究について述べ，最後に 8. 章でまとめを述べる．

2. 研究動機

ここでは，1. 章でも述べた既存のデスクトップサーチにおける問題点について例を挙げて示し，その原因を明らかにしていく．

2.1 OS が管理するディレクトリ構造の限界による問題点

ファイル内容の部分検索が不可能である．階層的に散らばるフォルダまたはファイルに関して問い合わせを行う能力はあるが，特定のファイル形式 (例えば LaTeX や XML) の内容にある構造情報に関しては，OS のファイルシステムだけに作用するため，ファイル中の構造が無視され検索結果として文章全体を表示する．これでは，利用者は必要な情報をファイル中から探そうとする度に再び検索を行う必要があり不便である．LaTeX や XML 形式のテキストデータを検索対象にした場合では，すべてを表示するとデータが長くなる場合があるため部分的な検索機能が望ましいと考える．図 1 は既存のデスクトップサーチで問い合わせを行った場合の例であり，卒論ファイル内のすべての文章が返される．これは，ファイルシステムとファイルの中身を統一してシームレスな検索を支持することができないということになる．

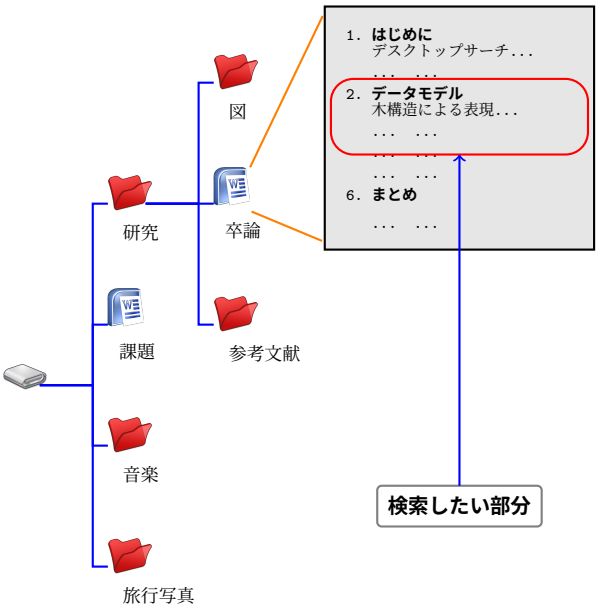

図 1　ファイルの中身の部分的な検索

Fig. 1　Retrieval of a section of a file's contents

2.2 キーワード検索におけるセマンティックスの曖昧さによる問題点

特定の問い合わせ言語（SQL や XQuery）を利用する場合

は，あらゆる問い合わせのセマンティックスが明確にされていて，ある問い合わせに対する問い合わせ結果が何であるべきかが明確である．一方，キーワードによる検索の場合は，検索のセマンティックス自体が曖昧であり，検索結果が一定しないことが多くの研究者より指摘されている．本研究においても，複数のキーワードによる問い合わせを行った場合，そのキーワードをすべて含むものは解が返されるが，複数のうち 1 つでも含んでいないものがあれば検索結果が返されないという問題について採り上げている．つまり，従来の AND 検索のセマンティックスによる検索のみでは不十分であるということになる．これは，電子メールの検索や複数のファイルから構成される HTML 文書の場合でも同様のことが起こる．これらは，個々の要素を検索の単位としている．そのため，複数のキーワードにより検索した場合，ある任意の要素にすべてのキーワードを含む場合はいいが，複数の要素に分散してキーワードが現われる場合は，個々の要素単位でうまく検索できない．例えば，図 2 のように**「ネパール」**と**「ＪＡＬ」**というキーワードで検索を行った場合，個々のメールを検索範囲としていて 092 と 078 にキーワードが分散しているため結果が返されない．

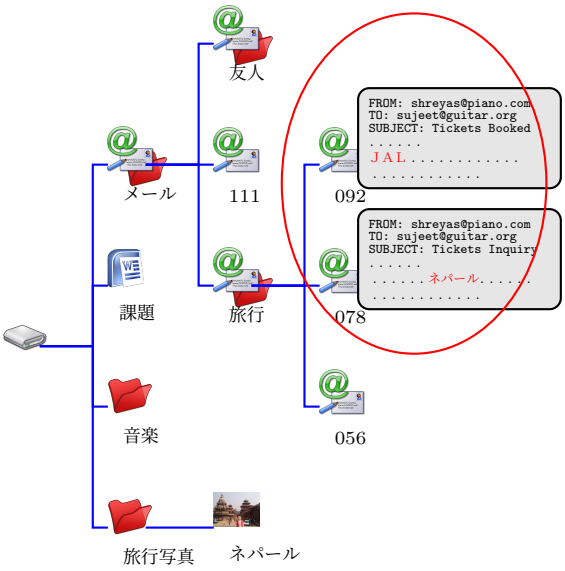

図 2　検索キーワードが関連する複数のデータに跨って現れる場合
Fig. 2　Query keywords scattered across multiple data units

3.　データモデル

ここでは，デスクトップ内のあらゆるデータを統一的に表現できるデータモデルについて提案していく．

3.1　モデルの概要

本研究では，ホスト OS のファイルシステムが管理するファイルやフォルダだけでなく，構造化文書のあるファイル（例えば LaTeX や XML 形式のテキストファイル，ワード・パワーポイントファイル）の中身の部分，メールやメールに添付されているファイルやそのファイルの部分的な中身までを検索対象にし，それらのデータのシームレスな検索を実現させることを目的としている．そのため，デスクトップ内の検索対象になりうるあらゆるデータを統一的に表現できるデータモデルが必要である．

デスクトップ内の多くの異種データは階層型構造で表現できる．例えば，LaTeX，XML や HTML のような構造化文書のあるファイルの中身は基本的に階層型構造により表せることができる．さらに，電子メールにおける FROM，TO，件名，メッセージのような論理的な部分や添付されたファイルの中身まで同様に階層構造で表現できる場合が多い．そこで，既存ファイルシステムがファイルやフォルダを管理するために扱う木構造を拡張し，ファイル内の部分的なデータまで表現できるようにする．

ファイルの中身をモデル化することには，データベース的な観点からも重要な意義がある．デスクトップを管理する場合，関連のあるファイルを同じフォルダ下におき，ファイルやフォルダを整理することが多い．現在のファイルシステムでは，このようなファイルやフォルダ間の論理的な関連による物理的な整理が可能であるが，この作業が容易ではない．そこで，ファイルの中身の部分的なデータ間の論理的な整理が物理的なものと独立に行えることが望ましい．これによって LaTeX や XML のような構造化文書の物理的な構造よりもむしろ論理的な構造に注目できる．例えば，一つの LaTeX 文書ファイルの物理的な構造が複数ありえるが，論理的な構造が唯一である．この考えを利用して物理的な構造を意識せずデータの管理や検索できるようになる．従来のリレーショナルデータベースシステムにおいても，このようなデータ独立の必要性が重要な課題として取り上げられている．デスクトップ内のデータ量が増えつつある現状の上，将来データベース的なモデルのサポートが重要になってくる [5]．

3.2　モデルの形式化

［**定義** 1］（データベース木） データベース木，もしくは単にデータベースは，節点集合 $\mathtt{N}$ と枝集合 $\mathtt{E} \subseteq \mathtt{N} \times \mathtt{N}$ を持つ順序木 $\mathcal{D} = (\mathtt{N}, \mathtt{E})$ であり，その他のあらゆる節点に経路 (パス) を持つ根が唯一存在する木である．

データベース木の各節点 $\mathtt{n}$ は，検索対象になりうる最小データ単位と対応づけられる．以降，検索対象になりうる最小データ単位のことを論理的要素と呼ぶことにする．さらに，各節点 $\mathtt{n}$ と対応する論理的要素中の代表するキーワードを返す関数 $\mathtt{keywords(n)}$，またその論理的要素の型を返す関数 $\mathtt{type(n)}$ が別に存在する．さらに，これらの節点は論理的要素間のトポロジが失われないように順序づけられる．ファイルやフォルダの場合はこの順序には特に重要な意味がおそらくないが，LaTeX やワードの構造化文書においては，この順序が重要な意味を持っている．ここで，全ての節点 $\mathtt{N}$ の代わりに $\mathtt{nodes}(\mathcal{D})$ と表記し，全ての枝 $\mathtt{E}$ の代わりに $\mathtt{edges}(\mathcal{D})$ と表記する．

［**定義** 2］（データベース木における断片） $\mathcal{D}$ が任意のデータベース木であるとすると，$\mathtt{nodes(f)} \subseteq \mathtt{nodes}(\mathcal{D})$ および $\mathcal{D}$ の $\mathtt{nodes(f)}$ により誘導された部分グラフが木ならば，$\mathtt{f} \subseteq \mathcal{D}$ はデータベース木における断片，もしくは単に断片である．すなわち，誘導された部分グラフは接続され，根を持つ木である．断片は，データベース木の節点の部分集合により表すことができる．

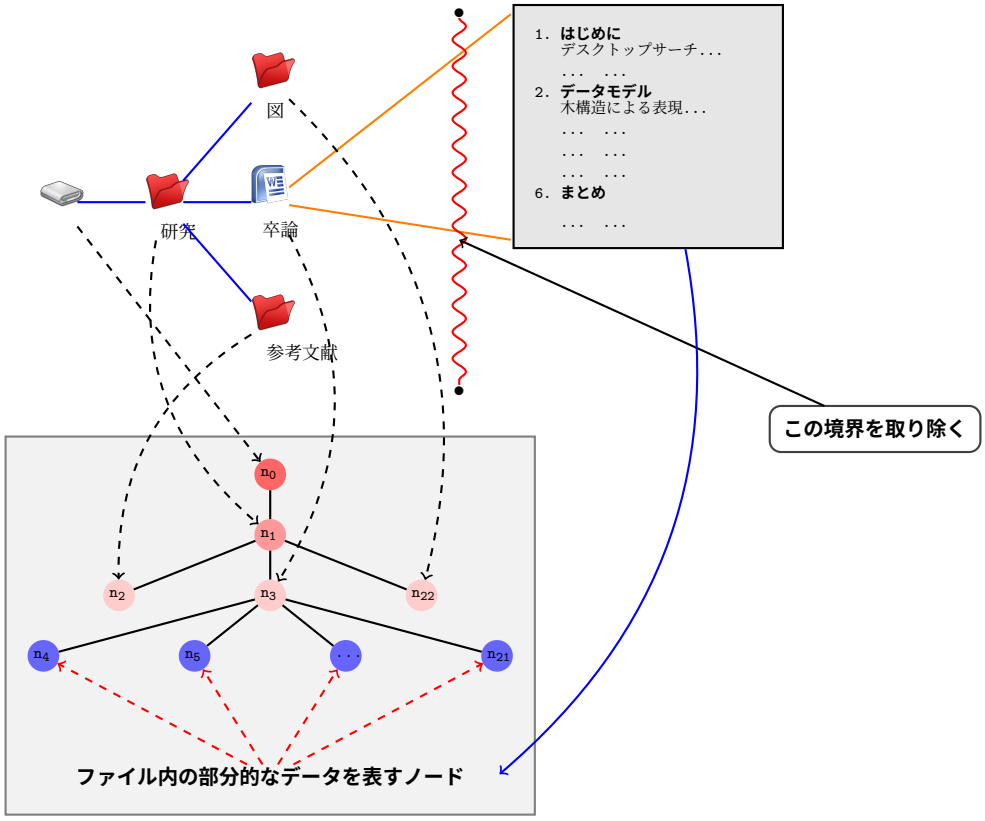

図 3　データモデル
Fig. 3　Data Model

図 3 はデスクトップ上の一部のデータを本モデルにマッピングしたものを示す．また，$\langle \mathtt{n1}, \mathtt{n3}, \mathtt{n4}, \mathtt{n5} \rangle$ を含む木は「研究」フォルダの下にある「卒論」文書の一部の内容を表す断片であるとし，その根は n1 である．以降，部分集合の表記により示される断片のはじめの節点を，誘導された木の根とする．さらに本論文では，唯一の節点からなる断片を単なる節点であると解釈する．

4. 問い合わせモデル

情報検索の分野で使われる問い合わせ処理法では，本研究で求められる検索結果が効果的に得られないため，従来のリレーショナルデータベースシステムにおける処理法と同様の方法を取ることにする．つまり，指定された問い合わせは，まず代数式に変換され，その代数式に定義されているセマンティックスによる解が生成され，それらの解からなる集合が問い合わせ結果として返される．また，本研究で扱う異種型データは，基本的に木構造で表せるため，木構造を用いたデータベースのためにこれまで提案してきた代数 [11] を利用する．以降，この代数のことは TaIZaBRa と呼ぶことにする．

4.1 TaIZaBRa の概観

TaIZaBRa とは，スキーマを持たない XML データの部分的な検索に適した代数である．これには，部分木同士やその部分木集合同士の結合演算群の定義，また，特定の性質を持つフィルタの概念等の理論がある．さらに，多数の問い合わせ処理最適化手法が理論的に証明されており，多面的なアプリケーションへの拡張が期待できる代数である．本研究で扱う異種データに対するキーワードによる問い合わせを，効果的，また効率良く処理を行うためには，TaIZaBRa の理論が基盤となってくるため，以下では，その TaIZaBRa における重要な演算や最適化手法について詳解する．

4.2 基本演算

ここでは，問い合わせ処理に用いる，断片や断片集合に対する基本演算について述べる．

[**定義** 3]（選択） 任意の文書の断片集合を $\mathtt{F}$ であるとし，$true$ あるいは $false$ に各断片を写像する述語を $\mathtt{P}$ であるとすると，述語 $\mathtt{P}$ による $\mathtt{F}$ からの選択は，$\mathtt{F}$ の部分集合 $\mathtt{F}'$ となる．$\mathtt{F}'$ は $\mathtt{P}$ を満たす全てかつ唯一の断片を含む．また，$\mathtt{F}$ からの選択は $\sigma_{\mathtt{P}}$ により示され，$\sigma_{\mathtt{P}}(\mathtt{F}) = \{\mathtt{f} \mid \mathtt{f} \in \mathtt{F}, \mathtt{P}(\mathtt{f}) = true\}$ となる．

以降，述語 $\mathtt{P}$ は選択 $\sigma_{\mathtt{P}}$ のフィルタと呼ばれる．最も単純なフィルタはキーワード 'k' のみを持つ断片を選択する 'keyword $= k$' 形式のフィルタであり，キーワード選択のために存在する．また，'size $< c$'（c は定数である）形式のフィルタは特定のサイズ c より小さい断片のみを選択する．ここで断片のサイズとは，それ自身に含まれる節点の数である．この他にも，実用的なフィルタの提案ができるが，本研究で扱う特定の性質を持つフィルタについて 5.3 章で述べる．

[**定義** 4]（断片結合） $\mathtt{f}_1, \mathtt{f}_2, \mathtt{f}$ をデータベース $\mathcal{D}$ の任意の断片であるとする．このとき，$\mathtt{f}_1 \bowtie \mathtt{f}_2$ により表される，$\mathtt{f}_1$ と $\mathtt{f}_2$

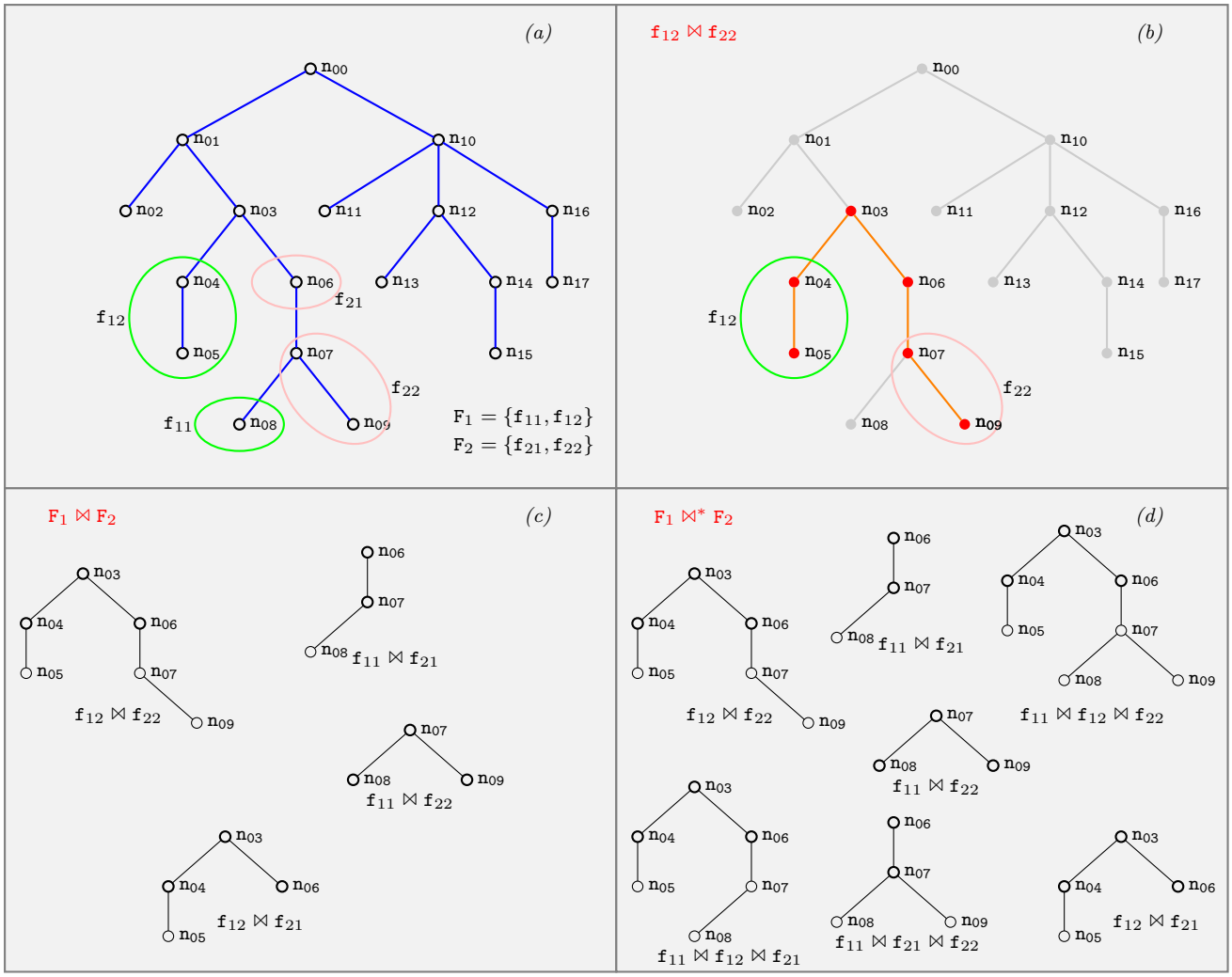

図 4　結 合 演 算

Fig. 4　(a) A Database Tree (b) Fragment Join (c) Pairwise Fragment Join and (d) Powerset Fragment Join Operations

の間で結合された断片が $\mathtt{f}$ であれば，以下が全て成り立つ．

（1）　$\mathtt{f}_1 \subseteqq \mathtt{f}$,

（2）　$\mathtt{f}_2 \subseteqq \mathtt{f}$,

（3）　$\mathtt{f}' \subseteqq \mathtt{f} \wedge \mathtt{f}_1 \subseteqq \mathtt{f}' \wedge \mathtt{f}_2 \subseteqq \mathtt{f}' \Rightarrow \not\exists \mathtt{f}'$

直感的に，任意の 2 つの断片 $\mathtt{f}_1$ と $\mathtt{f}_2$ の断片結合 とは，$\mathtt{f}_1$ と $\mathtt{f}_2$ が含まれる極小断片 (minimal fragment) である．ここで，任意の断片 $\mathtt{f}_1$，$\mathtt{f}_2$，$\mathtt{f}_3$ について断片結合は以下のような代数的性質を持つ．

等冪性　$\mathtt{f}_1 \bowtie \mathtt{f}_1 = \mathtt{f}_1$

可換性　$\mathtt{f}_1 \bowtie \mathtt{f}_2 = \mathtt{f}_2 \bowtie \mathtt{f}_1$

結合性　$(\mathtt{f}_1 \bowtie \mathtt{f}_2) \bowtie \mathtt{f}_3 = \mathtt{f}_1 \bowtie (\mathtt{f}_2 \bowtie \mathtt{f}_3)$

吸収性　$\mathtt{f}_1 \bowtie (\mathtt{f}_2 \subseteqq \mathtt{f}_1) = \mathtt{f}_1$

これらの性質は演算の容易な実装に役に立つだけではなく，5. 章に述べる代数操作による問い合わせ処理の最適化にも重要な役割を果たす．

［**定義** 5］（ペアワイズ断片結合）　データベース $\mathcal{D}$ の任意の断片集合 $\mathtt{F}_1$ と $\mathtt{F}_2$ があるとすると，$\mathtt{F}_1$ と $\mathtt{F}_2$ のペアワイズ断片結合は $\mathtt{F}_1 \bowtie \mathtt{F}_2$ により表され，それは $\mathtt{F}_1$ と $\mathtt{F}_2$ それぞれの要素同士のあらゆる組み合わせの断片結合を取ることにより誘導される断片集合であり，形式的には次の通りである．

$$\mathtt{F}_1 \bowtie \mathtt{F}_2 = \{\mathtt{f}_1 \bowtie \mathtt{f}_2 \mid \mathtt{f}_1 \in \mathtt{F}_1,\ \mathtt{f}_2 \in \mathtt{F}_2\}$$

ここで，任意の断片集合 $\mathtt{F}_1$, $\mathtt{F}_2$, $\mathtt{F}_3$ についてペアワイズ断片結合は以下のような代数的性質を持つ．

可換性　$\mathtt{F}_1 \bowtie \mathtt{F}_2 = \mathtt{F}_2 \bowtie \mathtt{F}_1$

結合性　$(\mathtt{F}_1 \bowtie \mathtt{F}_2) \bowtie \mathtt{F}_3 = \mathtt{F}_1 \bowtie (\mathtt{F}_2 \bowtie \mathtt{F}_3)$

単調性　$\mathtt{F}_1 \bowtie \mathtt{F}_1 \supseteqq \mathtt{F}_1$

分配性　$\mathtt{F}_1 \bowtie (\mathtt{F}_2 \cup \mathtt{F}_3) = (\mathtt{F}_1 \bowtie \mathtt{F}_2) \cup (\mathtt{F}_1 \bowtie \mathtt{F}_3)$

ただし，等冪性は成り立たないことに注意する．

［**定義** 6］（パワーセット断片結合）　データベース $\mathcal{D}$ の任意の断片集合 $\mathtt{F}_1$ と $\mathtt{F}_2$ があるとすると，$\mathtt{F}_1$ と $\mathtt{F}_2$ のパワーセット断片結合は $\mathtt{F}_1 \bowtie^* \mathtt{F}_2$ により表され，それは $\mathtt{F}_1$ 内および $\mathtt{F}_2$ 内に含まれる要素の任意数 (ただし，0 ではない) に対し断片結合を適用することにより誘導される断片集合である．形式的には次の通りである．

$$\mathtt{F}_1 \bowtie^* \mathtt{F}_2 = \{\bowtie (\mathtt{F}'_1 \cup \mathtt{F}'_2) \mid \mathtt{F}'_1 \subseteqq \mathtt{F}_1, \mathtt{F}'_2 \subseteqq \mathtt{F}_2, \mathtt{F}'_1 \neq \phi, \mathtt{F}'_2 \neq \phi\}$$

ただし, $\bowtie\{\mathtt{f}_1,\mathtt{f}_2,\ldots,\mathtt{f}_n\}=\mathtt{f}_1\bowtie\ldots\bowtie\mathtt{f}_n$

さらに，上記の定義は次のように展開できる．

$$\begin{aligned}\mathtt{F}_1\bowtie^*\mathtt{F}_2 \quad = \quad & (\mathtt{F}_1\bowtie\mathtt{F}_1)\cup\\ & (\mathtt{F}_1\bowtie\mathtt{F}_1\bowtie\mathtt{F}_2)\cup(\mathtt{F}_1\bowtie\mathtt{F}_2\bowtie\mathtt{F}_2)\cup\\ & (\mathtt{F}_1\bowtie\mathtt{F}_1\bowtie\mathtt{F}_1\bowtie\mathtt{F}_2)\cup\\ & (\mathtt{F}_1\bowtie\mathtt{F}_1\bowtie\mathtt{F}_2\bowtie\mathtt{F}_2)\cup\\ & (\mathtt{F}_1\bowtie\mathtt{F}_2\bowtie\mathtt{F}_2\bowtie\mathtt{F}_2)\cup\ldots\end{aligned}$$

図 4 に，本章中でこれまで示した結合演算を示す．(a) は演算の対象である，断片集合 $\mathtt{F}_1$ と $\mathtt{F}_2$ を持つ任意のデータベース木を示す．そして (b) は，(a) 中のデータベース木において，(a) 中の断片集合 $\mathtt{F}_1$ と $\mathtt{F}_2$ を入力として断片結合を行った際の解を示す．同様に，(c) はペアワイズ断片結合を行った際の解を示し，(d) はパワーセット断片結合を行った際の解を示す．

ここで，パワーセット断片結合演算は本研究で扱うキーワードによる問い合わせに対し，問い合わせ結果と思われる全ての断片を生成するために用いられる重要な演算であるが，コストの高い演算であることに注意してもらいたい．

4.3 問い合わせ処理

[**定義** 7]（問い合わせ） ある問い合わせは，$j=1,2,\ldots,m$ について $\mathtt{Q}_\mathtt{P}\{\mathtt{k}_1,\mathtt{k}_2,...,\mathtt{k}_m\}$ とする．このとき，$\mathtt{k}_j$ は問い合わせの各キーワードである．また，$\mathtt{P}$ は選択述語である．

ここでは，節点 $\mathtt{n}$ に対応づけられたデータベースの内容に問い合わせの任意のキーワード $\mathtt{k}$ が現れることを，$\mathtt{k}\in\mathtt{keywords}(\mathtt{n})$ と表記する．

[**定義** 8]（問い合わせの解） ある問い合わせ $\mathtt{Q}_\mathtt{P}\{\mathtt{k}_1,\mathtt{k}_2,...,\mathtt{k}_m\}$ を考えるとき，この問い合わせに対する解 $\mathtt{A}$ は次のように定義する．

$$\{\,\mathtt{f}\mid(\forall\mathtt{k}\in\mathtt{Q})\exists\mathtt{n}\in\mathtt{f}:\ \ \mathtt{n}\text{ は }\mathtt{f}\text{ における葉節点である}\ \wedge\ \mathtt{k}\in\mathtt{keywords}(\mathtt{n})\wedge\mathtt{P}(\mathtt{f})=\mathrm{true}\,\}$$

本章でこれまで説明を行った演算を用いることにより，データベース $\mathcal{D}$ に対し，例えば $\mathtt{Q}_\mathtt{P}\{\mathtt{k}_1,\mathtt{k}_2\}$ により表される問い合わせは次式により評価することができる．

$$\mathtt{Q}_\mathtt{P}\{\mathtt{k}_1,\mathtt{k}_2\}\ =\ \sigma_\mathtt{P}(\mathtt{F}_1\bowtie^*\mathtt{F}_2)$$

ただし $\mathtt{F}_1=\sigma_{\mathsf{keyword}=k_1}(\mathtt{F})$, $\mathtt{F}_2=\sigma_{\mathsf{keyword}=k_2}(\mathtt{F})$ および $\mathtt{F}=\mathtt{nodes}(\mathcal{D})$.

上記で述べたように，パワーセット断片結合演算は高コストの演算であり，本検索モデルの実用性を高めるためには，問い合わせ処理の最適化手法が必要である．次は，TALZBRAにおける最適化手法について述べる．

5. 最適化手法

代数操作（algebraic manipulation）による最適化手法は，従来のリレーショナルデータベース管理システムにおいて，古くから使われている論理的な最適化手法であり，問い合わせ処理の効率を高める最も重要な役割を果たしてきている [13]．この手法は，実装環境に依存しないため，独立に行えるという利点がある．この手法の目的とは，ある代数式が与えられたときに，その式と等価でありながら，より効率良く処理できる代数式の有無を調べることである．次は，TALZBRAにおける同様の論理最適化手法について述べる．

5.1 パワーセット断片結合演算の代数操作

上記のパワーセット断片結合演算の定義による処理法では，入力データのサイズが大きい場合，計算量が急速に増えることがあり得ると考えられ，問い合わせモデルの実用性に大きな疑問が問われる．以下で，このパワーセット断片結合演算の等価な定義への変形について詳解するが，まずは，その等価な代数式のために必要な演算を解説する．

[**定義** 9]（不動点） $\mathtt{F}$ は $\mathcal{D}$ における断片集合であるとする．このとき，$\mathtt{F}$ の不動点（$\mathtt{F}^+$）は次のように定義する．

$$\mathtt{F}^+=\{\bowtie\mathtt{F}_\mathtt{i}\mid\mathtt{F}_\mathtt{i}\subseteqq\mathtt{F}\wedge\mathtt{F}_\mathtt{i}\neq\emptyset\}.$$

直感的に，断片集合 $\mathtt{F}$ に対する不動点とは，$\mathtt{F}$ における任意の数 (0 でない) の部分集合へのペアワイズ断片結合演算の適用によって得られる全ての断片からなる断片集合である．また，この不動点は，$\mathtt{F}$ 自身へペアワイズ断片結合演算を $2^{|\,\mathtt{F}\,|}$ 回適用することにより求められるが，このような単純な計算法は，特に $|\,\mathtt{F}\,|$ の値が大きい場合，実用的であるとはいえない．この件については後ほど述べるが，不動点は問い合わせ結果を求めるために重要な役割を果たすため，不動点を効率良く計算することが重要である．

不動点を効率的に求めるために様々な手法が考えられる．例えば，不動点の定義はさらに次のように展開できるため，動的計画法（Dynamic Programming）を用いたアルゴリズムによって不動点を効率良く求めることが可能である．

$$\mathtt{F}^+=\mathtt{F}\cup(\mathtt{F}\bowtie\mathtt{F})\cup(\mathtt{F}\bowtie\mathtt{F}\bowtie\mathtt{F})\cup\ldots$$

さらに，次に解説するように論理的な手法も考えられる．

上述したように，この断片集合 $\mathtt{F}$ に対する不動点は，$\mathtt{F}$ 自身へペアワイズ断片結合演算を $2^{|\,\mathtt{F}\,|}$ 回適用することにより求められる．しかし，実際には $2^{|\,\mathtt{F}\,|}$ よりも少ない回数で $\mathtt{F}$ に対する不動点が求められる場合が多い．そこで，この論理的な手法の基本的な考え方とは，ペアワイズ断片結合演算を適用する必要な繰り返しの回数を予め引き出すことである．この回数は，与えられた断片集合の濃度に依存するのではなく，その断片集合の部分集合の濃度に依存することが証明できる．さらに，この部分集合が唯一であり，次の演算を適用することによって求められる．

[**定義** 10]（断片集合減少） $\mathtt{F}$ は $\mathcal{D}$ における断片集合であるとする．このとき，$\mathtt{F}$ への断片集合減少演算とは次のように定義できる．

$$\ominus(\mathtt{F})=\{\mathtt{f}\mid\ \nexists\mathtt{f}',\mathtt{f}''\in\mathtt{F}\text{ such that }\mathtt{f}\subseteqq\mathtt{f}'\bowtie\mathtt{f}''\}$$

ただし，$\mathtt{f},\mathtt{f}',\mathtt{f}''$ それぞれは $\mathtt{F}$ における異なる断片である．

直感的に断片集合減少演算とは，断片集合 $\mathtt{F}$ が与えられたときに，$\mathtt{F}$ から特定の要素を取り除くが，この特定の要素とは $\mathtt{F}$

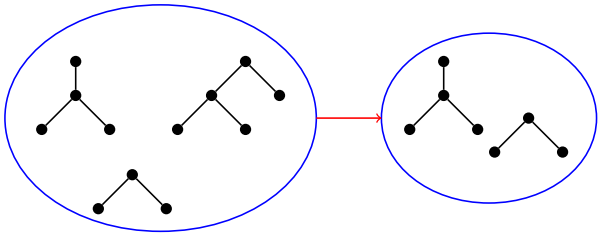

図 5　size(f) ≦ 4：サイズが 4 より大きい断片を除外する逆単調性を満たすフィルタ
Fig. 5　An antimonotonic filter that excludes fragments having sizes bigger than 4

における任意の異なる断片同士の断片結合からなる断片に含まれる断片群である．⊖(F) は，F に対する *reduced set* と呼ぶことにする．この *reduced set* の濃度が不動点を求めるためにペアワイズ断片結合演算を適用する必要な繰り返しの回数になることが証明されている．

これらによりパワーセット断片結合演算の定義は次のように変換することが可能である．

$$F_1 \bowtie^* F_2 = F_1^+ \bowtie F_2^+$$

ただし，F_1^+ と F_2^+ はそれぞれ F_1 と F_2 の不動点である．ここで重要なのは，たとえパワーセット断片結合演算の変換後の定義によって問い合わせ処理を行った場合でも，必ずしも多大な性能向上が得られるとはいえない．なぜなら，F_1^+ や F_2^+ を求めるのに，まず F_1 や F_2 それぞれへ断片集合減少演算を適用する必要があり，この断片集合減少演算を適用することによって元の集合の濃度が減少するという保証はないからである．従って，パワーセット断片結合演算の代数操作のみでは，問い合わせ処理の最適化が十分に効果的であるといえない．次に，より効果的な代数操作による最適化手法について述べる．

5.2　選択演算と結合演算との可換性

問い合わせ処理の初期段階では，なるべく選択演算を行うことが従来のデータベース・システムにおける代数操作による最適化の主要原理でもある [13]．TAIZBRAにおいても，選択演算と結合演算との実行順序が交換可能であり，この原則により最適化が実現できると述べられている．しかしながら，選択演算と結合演算との可換性はどんな場合においても適用できるわけではなく，これを実現させるためには，選択条件として指定されるフィルタが特定の性質を満たさなければならない．次に，この特定の性質を持つフィルタについて説明を行う．

5.3　逆単調性を満たすフィルタ

ある断片 f に対して，フィルタ P が逆単調性を満たすとは，$\forall f' \subseteq f : P(f) = true \Rightarrow P(f') = true$ が成り立つときをいう．すなわち，逆単調なフィルタ P が与えられ，また断片 f が与えられているとする．このとき，f が P を満たさなければ，f を含むどの断片も P を満たさない．逆単調なフィルタ同士の論理和と論理積両方とも逆単調性を満たす．つまり，P_1 と P_2 は異なる逆単調なフィルタとすると，$P_1 \wedge P_2$ と $P_1 \vee P_2$ も逆単調なフィルタである．以降，逆単調性を満たすフィルタは，他のフィルタとの区別が必要なとき P_a と記す．逆単調なフィルタの否定のフィルタは逆単調性を満たさないため，TAIZBRAにおいては，これらのフィルタが扱われていない．

TAIZBRAにおいて，選択条件に逆単調なフィルタが指定された場合，その選択演算とペアワイズ断片結合演算との実行順序が交換可能であることが証明されている．すなわち，

$$\sigma_{P_a}(F_1 \bowtie F_2) = \sigma_{P_a}(\sigma_{P_a}(F_1) \bowtie \sigma_{P_a}(F_2))$$

ただし，F_1 と F_2 は断片集合（F_1 と F_1 は異なる必要がない）であり，P_a は逆単調なフィルタである．

さらに，パワーセット断片結合演算は次のように評価することができる．

$$\begin{aligned}\sigma_{P_a}(F_1^+ \bowtie F_2^+) &= \sigma_{P_a}(\sigma_{P_a}(F_1^+) \bowtie \sigma_{P_a}(F_2^+)) \\ &= \sigma_{P_a}(\sigma_{P_a}(\sigma_{P_a}(F_1) \bowtie \sigma_{P_a}(F_1) \bowtie \ldots \\ &\qquad \bowtie \sigma_{P_a}(F_1)) \\ &\qquad \bowtie \sigma_{P_a}(\sigma_{P_a}(F_2) \bowtie \sigma_{P_a}(F_2) \bowtie \ldots \\ &\qquad \bowtie \sigma_{P_a}(F_2)))\end{aligned}$$

逆単調なフィルタ同士の論理和と論理積両方とも逆単調性を満たすため，逆単調性を満たすより複雑なフィルタを作ることが可能になり，より実用的な問い合わせを効率良く処理できる．

図 5 には，サイズが 4 より大きい断片が取り除かれる逆単調性を満たすフィルタ (size(f) ≦ 4) を示している．本研究において，問い合わせ結果は関連のある複数のデータ単位から構成される断片の集合である．生成される断片のサイズが大きければ大きいほど，その断片に無関係のノード（ノイズ）も多数含まれてしまう可能性も非常に高い．結合演算群のみでは，検索結果にこのような不要と思われる解が多く含まれる可能性が高く，検索の精度が落ちてしまう．しかし，逆単調性を満たすフィルタを導入することによって不要な解を問い合わせ処理の初期段階で除外することが可能になり，問い合わせ処理の効率が向上することがわかる．

6.　デスクトップサーチへ TAIZBRAの適用例

ここでは，デスクトップサーチへ TAIZBRAの適用することにより目的のデータが検索できることを例を挙げて示す．

図 6 は図 2 に示しているデスクトップ内のデータを表現する

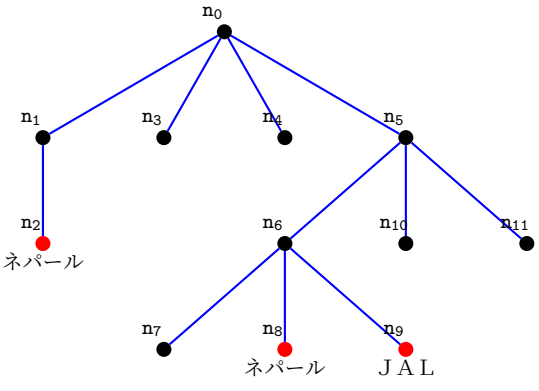

図 6　問い合わせ処理：例
Fig. 6　An Example of Query Evaluation

データベースを示す．このデータベースに対して次の問い合わせを考える．

$$\mathtt{Q}_{\mathsf{P_a}}\{\text{ネパール}, \text{JAL}\}$$

ただし，$\mathsf{P_a}$ は任意の逆単調性を満たすフィルタであり，それは $\mathsf{size} \leqq 4$ と設定する．この問い合わせは次式で評価できる．

$$\mathtt{Q}_{\mathsf{P_a}}\{\text{ネパール}, \text{JAL}\} = \sigma_{\mathsf{size} \leqq 4}(\mathtt{F}_1 \bowtie^* \mathtt{F}_2)$$

ただし，$\mathtt{F}_1 = \sigma_{\mathsf{keyword}=\text{ネパール}}(\mathtt{F})$, $\mathtt{F}_2 = \sigma_{\mathsf{keyword}=\text{JAL}}(\mathtt{F})$. また $\mathtt{F} = \mathtt{nodes}(\mathcal{D})$.

ここでは, $\mathtt{F}_1 = \{\langle \mathtt{n}_2 \rangle, \langle \mathtt{n}_8 \rangle\}$ 及び $\mathtt{F}_2 = \{\langle \mathtt{n}_9 \rangle\}$ である．また，$\langle \mathtt{n}_2 \rangle$ は $\mathtt{f}_2$, $\langle \mathtt{n}_8 \rangle$ は $\mathtt{f}_8$ などと書く．次は，$\mathtt{F}_1$ の不動点 $\mathtt{F}_1^+$ は $\{\mathtt{f}_2, \mathtt{f}_8, \mathtt{f}_2 \bowtie \mathtt{f}_8\}$ になり，$\mathtt{F}_1$ の不動点 $\mathtt{F}_2^+$ は $\{\mathtt{f}_9\}$ になる．

ここで，$\{\mathtt{f}_2 \bowtie \mathtt{f}_8\}$ からなる断片は $\mathsf{size} \leqq 4$ により初期段階で取り除かれる．また，$\{\mathtt{f}_2 \bowtie \mathtt{f}_9\}$ 及び $\{\mathtt{f}_8 \bowtie \mathtt{f}_9\}$ からなる断片のうち，前者はフィルタにより次の段階で除かれ，後者からなる断片，すなわち $\{\langle \mathtt{n}_6, \mathtt{n}_8, \mathtt{n}_9 \rangle\}$ が最終的な問い合わせ結果になる．

7. 関 連 研 究

近年，異種型データを統一させ管理，検索することと同様の問題を採り上げた研究は注目されている [7]．本研究と最も関連している研究としては，iMeMex プロジェクトである [5]．この研究では，データベースの SQL，XQuery のような問い合わせ言語での利用やグラフデータモデルでデータ表現されているという点で本研究とは異なる．表 1 には，このプロジェクトを本研究と比較対照したものを示す．表 1 により本研究は純粋なデータベース的なアプローチでなく，統合的なアプローチであることが明確にわかる. [3] では，近年のアプリケーションには，情報検索とデータベースの統合的なアプローチの重要性について述べられている．

また，キーワードによる検索において，様々なデータを対象とした研究がある．リレーショナルデータベースのキーワード検索をサポートするシステムは, [1] [8] [10] で行われた．さらに，XML データに対するキーワードに基づく検索は, [4] [9] [12] [15] で行われた．

また，デスクトップサーチにおける統一したデータベースとの研究は少なく，今後さまざまな効果が期待される研究課題である．

8. おわりに

本研究では，既存のデスクトップサーチツールにおける問題点を取り上げ，異種型データをシームレスに管理，検索を目的とし，統一的なデータモデルの提案を行い，デスクトップサーチへ TAIZBRA の適用する手法について述べた．これにより，ファイル内の部分的な検索問題や検索キーワードが「関連する複数のデータ単位」に跨って現れるとき起こりえる検索結果の問題点を解決できた．問い合わせの結果を関連度順にランキングして表示する方法の提案やシステムの実装を行うことが今後の課題である．

	iMeMex	本研究
扱えるデータ種	異種型データ	異種型データ
アプローチ	純粋なデータベース的なアプローチ	情報検索・データベース統合的なアプローチ
データモデル	グラフデータモデル	根付き順序木構造
検索手法	XPath に基づく問い合わせ言語	TAIZBRA によるデータベース的な問い合わせ処理法
長所	細かい問い合わせが可能	問い合わせが容易
短所	複雑な問い合わせ言語	キーワードと単純なフィルターによる問い合わせに限定

表 1　関連プロジェクト iMeMex との比較
Table 1　Comparison of our approach with iMeMex approach

謝　　辞

本論文に関して貴重な指摘を頂いた DEWS 学会の関係各位に深く感謝します．

文　　献

[1] Sanjay Agrawal, Surajit Chaudhuri, and Gautam Das. DBXplorer: A system for keyword-based search over relational databases. In *ICDE*, pages 5–16, 2002.

[2] Apple Mac OS X Spotlight. `http://www.apple.com/macosx/features/spotlight/`.

[3] Surajit Chaudhuri, Raghu Ramakrishnan, and Gerhard Weikum. Integrating DB and IR technologies: What is the sound of one hand clapping? In *CIDR*, pages 1–12, 2005.

[4] S. Cohen, J. Mamou, Y. Kanza, and Y. Sagiv. XSEarch: A semantic search engine for XML. In *Proc. of 29th VLDB*, pages 45–56, 2003.

[5] Jens-Peter Dittrich and Marcos Antonio Vaz Salles. iDM: a unified and versatile data model for personal dataspace management. In *VLDB*, pages 367–378, 2006.

[6] Google Desktop. `http://desktop.google.com/`.

[7] Alon Y. Halevy, Michael J. Franklin, and David Maier. Principles of dataspace systems. In *PODS*, pages 1–9, 2006.

[8] Vagelis Hristidis and Yannis Papakonstantinou. DISCOVER: Keyword search in relational databases. In *VLDB*, pages 670–681, 2002.

[9] Y. Li, C. Yu, and H. V. Jagadish. Schema-free XQuery. In *Proc. of 30th VLDB*, pages 72–83, 2004.

[10] Fang Liu, Clement T. Yu, Weiyi Meng, and Abdur Chowdhury. Effective keyword search in relational databases. In *SIGMOD Conference*, pages 563–574, 2006.

[11] Sujeet Pradhan. An algebraic query model for effective and efficient retrieval of XML fragments. In *VLDB*, pages 295–306, 2006.

[12] A. Schmidt, M. Kersten, and M. Windhouwer. Querying XML documents made easy: Nearest concept queries. In *ICDE*, pages 321–329, 2001.

[13] J. D. Ullman. *Principles of Database and Knowledge-Base Systems Vol. II.* Computer Science Press, 1989.

[14] Windows Desktop Search. `http://www.microsoft.com/windows/desktopsearch`.

[15] Y. Xu and Y. Papakonstantinou. Efficient keyword search for smallest LCAs in XML databases. In *SIGMOD*, pages 527–538. ACM, June 2005.